International Journal of Intelligent Transportation Systems Research (2022) 20:778–792
https://doi.org/10.1007/s13177-022-00325-1

Camera-Based System for the Automatic Detection of Vehicle Axle Count and Speed Using Convolutional Neural Networks

Victoria Miles[1] · **Francis Gurr**[1] · **Stefano Giani**[1]

Received: 19 October 2021 / Revised: 2 September 2022 / Accepted: 9 September 2022 / Published online: 17 September 2022

Abstract
This paper outlines the development of a non-intrusive alternative to current intelligent transportation systems using roadside video cameras. The use of video to determine the axle count and speed of vehicles traveling on major roads was investigated. Two instances of a convolutional neural network, YOLOv3, were trained to perform object detection for the purposes of axle detection and speed measurement, achieving accuracies of 95% and 98% mAP respectively. Outputs from the axle detection were processed to produce axle counts for each vehicle with 93% accuracy across all vehicles where all axles are visible. A simple Kalman filter was used to track the vehicles across the video frame, which worked well but struggled with longer periods of occlusion. The camera was calibrated for speed measurement using road markings in place of a reference object. The calibration method proved to be accurate, however, a constant error was introduced if the road markings were not consistent with the government specifications. The average vehicle speeds calculated were within the expected range. Both models achieved real-time speed performance.

Keywords Traffic monitoring · Road speed · Axle count · CNN

1 Introduction

Intelligent transportation systems (ITS) which monitor road use provide valuable data which can be used to inform many urban planning decisions. Traffic monitoring installations which continually track the number of passing vehicles can be used to build up a picture of how a particular road is used. Monitoring additional factors, such as the category or speed of vehicles, can help make this picture more detailed and improve its utility for various applications. Comparing the data produced from monitoring multiple roads to observe where traffic is typically heaviest or where heavy vehicles are more common allows for the efficient allocation of funds for road maintenance. Information into varying traffic density and speed at different times of day can inform the design of smart cities and motorways, and reliable data into where congestion frequently occurs can be used to highlight locations where such developments would be most beneficial. Additionally, road transport accounts for 22% of total UK emissions of carbon dioxide, and noise from road traffic affects 30% of people in the UK [1]. Traffic monitoring data can be used to identify problem areas and so help create and track initiatives that aim to address such issues.

Existing research into camera-based ITS solutions [6, 31, 34] tends to focus on simple shape-based classification of vehicles into general categories, such as car or van. These categories are much more general than those produced using traditional solutions in which the HGV (heavy goods vehicle) class in particular can be divided into many subclasses [27]. A key distinction between these subclasses is the number of axles the vehicle has, but little existing research has investigated the possibility of using a neural network to detect wheels, and so count axles, from video footage. Similarly, little research has been performed into the possibility of developing an integrated system capable of both classifying vehicles and estimating their speed.

This paper will detail the development of camera-based systems for the automatic detection of vehicle axle counts and vehicle speeds using a convolutional neural network

✉ Victoria Miles
victoria.s.miles@durham.ac.uk

Francis Gurr
francis.gurr@gmail.com

Stefano Giani
stefano.giani@durham.ac.uk

[1] Department of Engineering, Durham University, Stockton Road, Durham DH1 3LE, UK

(CNN). This research is carried out in collaboration with Q-Free, an ITS company with an interest in developing a sophisticated roadside traffic monitoring system with a focus on vehicle classification. A base system will be designed to perform simple categorisation of vehicles based on shape. The aim is for the axle count and vehicle speed systems to exist as additional modules to provide clients with the flexibility to choose the data they require.

Traditionally, the market for vehicle classification has been dominated by in-road systems [30] using sensors such as inductive loops to detect axles in order to classify vehicles. These solutions are capable of classifying vehicles into very specific categories with a high degree of accuracy. However, a major disadvantage is that they are intrusive; they require the road to be closed during installation and maintenance [5]. This is costly, time-consuming and endangers workers' lives.

Therefore, there is considerable demand for non-intrusive solutions that can be deployed out-of-road. Such systems have previously been proposed based on a mix of sensing technologies, such as radar, LiDAR and video footage, however improvements in machine learning and object detection offer the unique proposal of a simpler and cheaper system based only on video.

There are some major challenges which must be addressed when developing a camera-based system to compete with traditional in-road solutions. Compared to induction loops, cameras are expected to perform with less accuracy, as differing weather and lighting conditions add to the complexity of the system. Any roadside installation covering multiple lanes also needs to account for cases of occlusion, when one vehicle fully or partially obscures another. In-road sensors placed beneath each lane are affected by neither occlusion nor weather and so performance tends to be much more reliable.

However, non-intrusive systems benefit from vastly lower installation and maintenance costs. Another advantage of camera-based systems over other sensor technology is that existing traffic surveillance cameras can be utilised, further reducing costs of installation.

The first step for both systems proposed in this paper is the detection of vehicles, and wheels for the axle count system, from camera footage.

For this purpose, a CNN-based object detector will be used. A CNN is a deep neural network which contains a number of convolutional layers, in which a small filter is passed over an input matrix and dot products are taken at each filter location in order to produce the output of the layer [13, 17, 26]. This process maintains spatial structure of the input, making CNNs well suited to image recognition tasks as both simple and complex structures can be identified in the image at different layers of the network. Object detection refers to a subset of image classifiers in which, instead of predicting a class for the entire image, the classes and locations of objects within the image are identified by the model. These models should be capable of identifying any number of objects at various scales and in various poses.

In 2014 Girshick et al. presented R-CNN [10], a network in which region proposals are produced using the selective search algorithm [28]. The primary network computes features, leading to a classification for each region of interest. R-CNN was able to outperform other existing systems at the time of publication in terms of accuracy but it is slow to train and test and memory usage is high. Following the publication of R-CNN, considerable research has been carried out into the use of CNNs to detect and classify objects in images.

Fast R-CNN [9] attempted to improve on the efficiency of its predecessor by running the entire input image through several convolutional layers, then projecting the region proposals onto the convolutional feature map produced. Faster R-CNN [23] uses a Region Proposal Network (RPN) which shares convolutional features with the detection network in order to further improve speed performance of the network.

Whilst R-CNN was initially designed for accuracy and then optimised for speed, other networks were primarily designed for speed. One such network is YOLO (You Only Look Once) [22]. When YOLO performs detections, the input image is divided into a grid and each grid cell predicts boundary boxes, with confidence scores representing likelihood of the box containing an object and how accurate the box shape and position is expected to be. YOLO is capable of real-time performance but with poor accuracy when compared to Faster R-CNN. Versions of YOLO with improved accuracy were proposed in YOLOv2 [20] and YOLOv3 [21].

The industrial application of this work requires real-time performance. Therefore, YOLOv3 was used for the task of object detection. Two separate instances of the YOLOv3 network were trained for use with each of the two systems proposed.

2 Theory

2.1 Camera Classification

A key challenge for camera-based vehicle classification which does not affect in-road systems is the issue of partial or full occlusion of vehicles. This has an impact on any camera-based classification system but is especially problematic for axle detection as, in addition to the distortion of the shape of partially occluded vehicles, in many cases some or all wheels will be fully occluded. The nature of this issue means that a certain level of error is unavoidable. In the case of shape-based classification, vehicles which are entirely covered in the image by other vehicles cannot be classified

correctly even with a very high-performing network. Similarly, for the case of axle detection, when no axles of the vehicle can be seen due to occlusion it is impossible for any network to produce a meaningful estimate of axle count.

For the purposes of the axle detection solution presented in this paper, any vehicle on which no wheels are visible will be considered fully occluded. Any vehicle on which only some wheels are visible will be considered partially occluded, and any vehicle on which all wheels are visible will be considered to not be occluded.

In cases of full occlusion, error is unavoidable and so its effects can be ignored when evaluating the accuracy of an axle detection system and only considered when deciding whether video-based classification is appropriate for a particular application.

The effects of partial occlusion must be considered more carefully. For cases of partial occlusion there are two approaches available. The first approach would be to instruct the model to produce its best estimate of how many axles the vehicle has and report that number. Alternatively, the model could label that particular vehicle detection as occluded and therefore not output any count, removing the possibility of presenting incorrect data. Each approach should be carefully considered with reference to typical results of the network in order to choose the strategy with the likelihood of producing the most accurate output.

The issue of partial and full occlusion can also be addressed by implementing the system using a camera positioned at an appropriate height and angle to minimise the frequency of these cases. The significance of camera position and recommendations for optimal performance will be further discussed in Section 4.4.

Another potential issue is the presence of partially visible vehicles, those entering or leaving the frame. In some such cases not all axles are in frame, making a count of axles for these vehicles impossible. Similar considerations of how to process these cases could be made. However, when the models are run on video footage rather than still frames, vehicles should all only be partially visible for a small portion of the time they are present in the video. With an effective tracking algorithm it should be possible to ensure axle counts are predicted only when vehicles are fully visible, although this is outside the scope of this paper. Therefore, for the purposes of this paper, partially visible vehicles can be safely ignored when performing axle counts.

2.2 YOLO

YOLO is a an extremely fast object detection CNN as it does not use region proposals. Instead it predicts bounding box locations and class probabilities from just one 'look' at an image.

YOLOv3 uses a feature extractor called Darknet-53 [18] which consists of 53 convolutional layers, which extract a high-level feature map given an input image.

The features learned by the convolutional layers are passed to a detection network which predicts the location of objects at three different scales. At each scale the image is divided into an $S \times S$ grid and a set of bounding box predictions is obtained per grid cell, adapted from a predetermined set of anchor boxes. The first detection layer produces the largest bounding box predictions as the resolution at this stage is the lowest. The network then works at a finer resolution to predict medium sized bounding boxes in the second detection layer and the smallest bounding boxes are predicted in the final detection layer, where the resolution of the network is highest. This process is illustrated in Fig. 1.

A bounding box's position is defined by width, height and centre coordinates. For each bounding box an objectness score is calculated, representing the likeliness of an object being present in the box. Additionally, C class scores are predicted, representing the probability that the object present belongs to each of C classes.

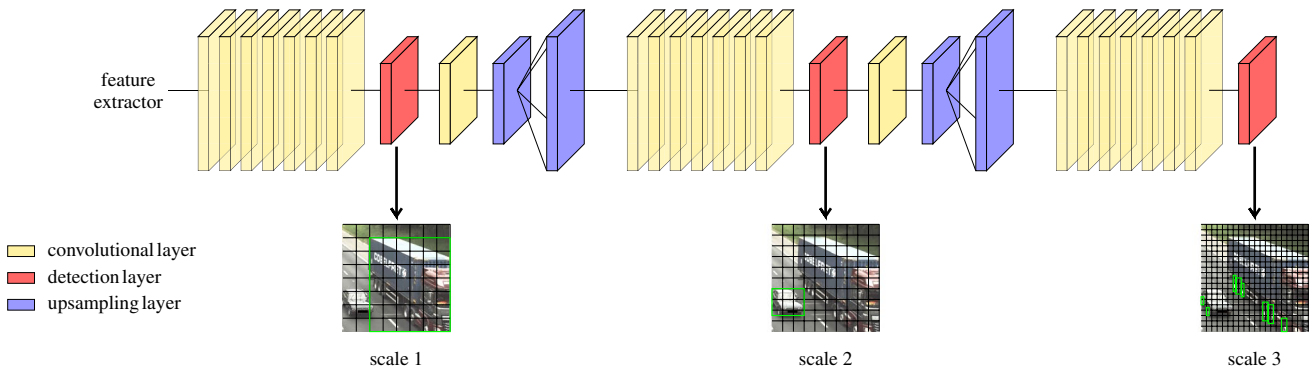


Fig. 1 A simplified diagram of the YOLOv3 detection network, example detections and network layers are displayed for illustration purposes only and not shown to scale

This system of anchors and predictions at different resolutions allows the network to accurately and efficiently locate objects of varying sizes in an image.

For an input image of size 416 × 416 pixels, YOLO predicts a total of 10647 bounding boxes, therefore the output must be processed first in order to get a reasonable set of results. Initially, the boxes are filtered based on their objectness and confidence scores and boxes below a threshold are ignored.

Non-maximal suppression is then used to remove multiple detections of the same object. This is implemented by selecting the box with the highest score, then removing any overlapping boxes. Overlap is determined using intersection over union (IOU), which is calculated as the area of overlap of two bounding boxes, divided by the total area covered by the two boxes.

YOLO is a network designed primarily for speed; bounding box locations and class predictions are all made from a single pass through the network and source code is written in c for maximum efficiency. This prioritisation of speed should be advantageous for meeting the real-time performance requirement, making accuracy of detections the primary consideration when training the networks.

2.3 Tracking

The Kalman filter [16] is a recursive algorithm used to estimate the state of a process using a form of feedback control. It is very useful for tracking applications where the aim is to predict where an object will next be found. The filter makes the assumptions that the model of the process is linear and that the noise is white Gaussian.

The filter can be broken down into two steps; predict and update. During the predict step the optimal prediction of the current state and its expected variance are calculated as follows:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \tag{1}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{2}$$

where $\hat{x}$ is the state prediction, P is the estimated uncertainty, k is the current step, A is the state transition matrix, B is the control matrix and Q is the noise uncertainty.

After a measurement, the update step combines the values and variances of the measurement and predicted state with a gain. Kalman gain is computed as:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{3}$$

where K is the gain, H is the observation matrix and R is the measurement uncertainty. Computed gain is used to update the estimates for state and variance as follows:

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{4}$$

$$P_k = (I - K_k H)P_k^- \tag{5}$$

where z_k is the output vector measured at step k.

If the measurement is noisy and has high variance, the gain is small and therefore reduces the weight of the measurement value. If the predicted variance is large, the gain will be large and so the measurement will be weighted heavily.

Here, the Kalman filter uses the bounding boxes predicted by the network as the measurements (z) in the update step and predicts the location of the corresponding vehicle at the next time step (x). By comparing the positions predicted by the filter and those predicted by the neural network for each time step, it is possible to track the movement of each vehicle across multiple frames of a video. Although tracking is a necessary component of any video-based classification system, a Kalman filter has been implemented only for the speed detection system for which tracking is a fundamental aspect of operation.

2.4 Camera Calibration

The process of camera calibration is necessary in order to determine vehicle speed in real-world units. It is used to find the parameters of the pinhole camera model that most closely approximate the camera the image was taken with. These parameters are represented by the camera projection matrix

$$P = K\begin{bmatrix} R & T \end{bmatrix} \tag{6}$$

where K consists of the intrinsic parameters and $\begin{bmatrix} R & T \end{bmatrix}$ corresponds to the extrinsic parameters, where R is the rotation matrix and T is the translation matrix [32].

The intrinsic parameters are described by the matrix

$$K = \begin{bmatrix} rf & \sigma & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where r is the aspect ratio, f is the focal length in pixels, σ is the skew factor and (u_0,v_0) are the coordinates of the principal point [33]. The skew factor describes the amount of shear distortion in the projected image, and the principal point is the intersection between the image plane and the line that passes through the pinhole of the camera. Often, the principal point is assumed to be (0,0), the skew factor is assumed to be 0 and the aspect ratio is set to 1, reducing the intrinsic parameters to the focal length.

The extrinsic parameters represent the position of the camera in the 3D world scene [7]. This is described by the

translation and rotation of the camera from the origin. Given the general camera model in Fig. 2, the horizontal rotation is given by the pan angle p, the vertical rotation by the tilt angle t, and the rotation along the camera's optical axis is given by the swing angle s. The location of the camera is given by the world coordinates $(X_{CAM},Y_{CAM},Z_{CAM})$.

Knowledge of the camera matrix allows for coordinates in the image plane to be projected onto the 3D world scene as shown.

$$\lambda\begin{bmatrix} x, & y, & 1 \end{bmatrix} = \begin{bmatrix} X, & Y, & Z, & 1 \end{bmatrix} P \tag{8}$$

where λ is the scale factor, (x,y) are the coordinates of a point on the image plane and (X,Y,Z) are coordinates of the corresponding point in the world scene.

The proposed calibration method is based on the methods presented by Fung and Yung [8] and He and Yung [14], and uses a rectangular pattern from road markings to calculate the camera parameters. The 3D world plane is defined with the X-axis along the direction of the road, the Y -axis perpendicular to the road and the Z-axis perpendicular to the surface of the road. It is assumed that the road is straight and flat and that all points on the road are on the Z = 0 plane in the 3D world scene. Four points on the road surface are selected using the road markings as a reference to form a rectangle $ABCD$ with the sides $\overrightarrow{AB}$ and $\overrightarrow{CD}$ parallel to the X-axis and the sides $\overrightarrow{AC}$ and $\overrightarrow{BD}$ parallel to the Y -axis in the world scene. The width and length of the rectangle are w and l respectively. This rectangle is shown in Fig. 3 as seen in both the image plane and the world scene.

The method has the advantage that it can be used to calculate all the necessary camera parameters without requiring any prior knowledge of the camera setup. The method presented by Fung and Yung (FY) calculates the parameters using the width of the rectangle and the two vanishing points which are found from the intersection of the parallel lines. However, FY suffers from ill-conditioning at pan angles close to 90 degrees. This is because as the pan angle tends to $90n$ degrees, where $n \in \mathbb{Z}$, the second vanishing point tends to infinity as one set of lines becomes parallel in the image scene. He and Yung (HY) propose an alternative method to calculate the parameters in these cases, using just one vanishing point and the width and length of the rectangle. This method cannot be used for every case as it suffers from large error in certain cases and therefore a combination of the methods must be used.

Table 1 presents the three different cases and the calibration method that is most suitable for each. The method used in Case 3 is the same as the standard HY method but with the labels of the rectangle vertices temporarily rotated by 90 degrees, effectively transforming the camera by a pan angle of 90 degrees in order to calculate the other parameters.

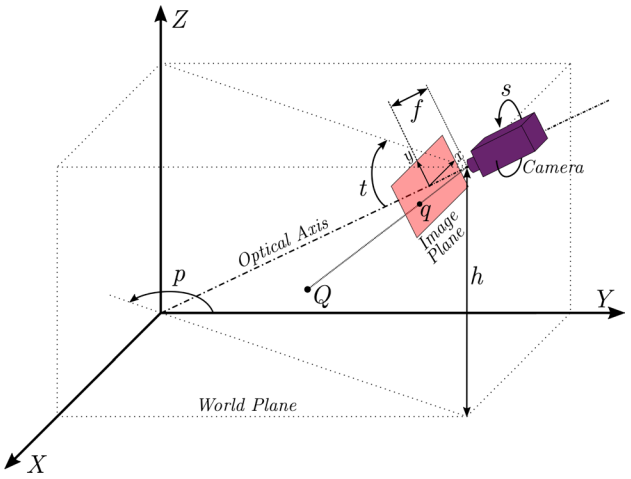
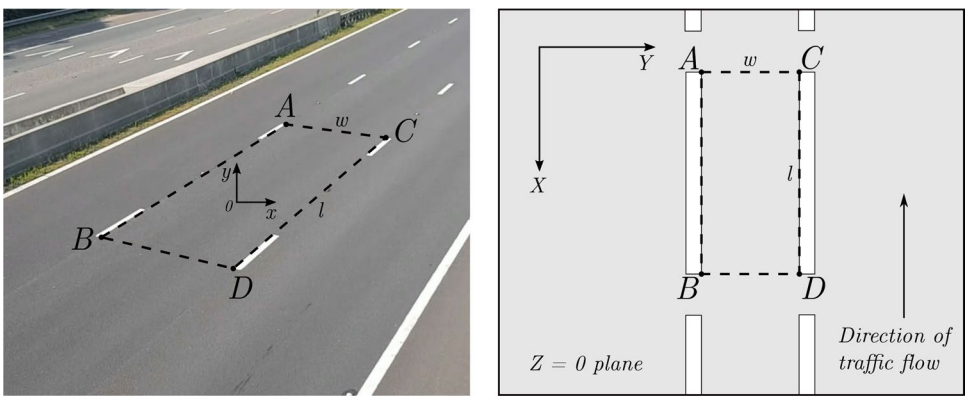


Fig. 2 General camera model (Based on [7])



(a) Viewed in the image scene

(b) Viewed top down in the 3D world scene

Fig. 3 Example of road markings used for calibration (Based on [8])

Table 1 Case selection

Case	Pan Angle	p	(degrees)	Calibration Method
1	$90n+30<$	p	$<90n+60$	Fung & Yung (FY)
2	$180n+60<$	p	$<180n+120$	He & Yung (HY)
3	$180n-30<$	p	$<180n+30$	HY Rotated (HY′)

where $n \in \mathbb{Z}$

3 Methods

3.1 Dataset and Evaluation of Cameras

The dataset used to train the speed calculation network contains around 15,000 images, randomly sorted into training, test and validation sets at a ratio of 7 : 2 : 1. All images are labelled with the coordinate positions of bounding boxes representing the locations of any vehicles within the image. The speed calculation network is trained to identify a single class of object, 'vehicle'.

The axle detection network is trained to identify two distinct classes of object in an image, 'vehicle' and 'wheel'. Due to the necessity of manually labelling wheel locations, a smaller dataset was used to train the axle detection network. This dataset consists of 3046 total labelled images, randomly sorted into a training dataset of 2440 images and test and validation datasets of 303 images each.

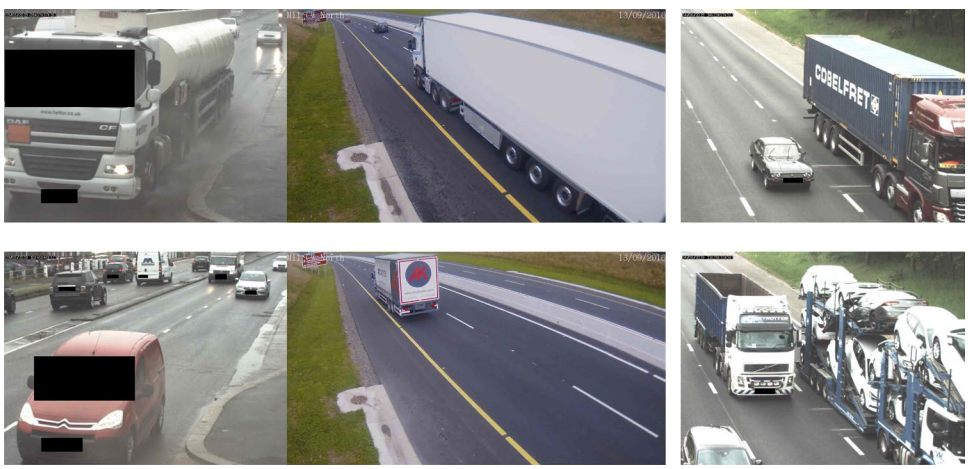
All images are frames taken from traffic surveillance footage of UK motorways and A-roads, with images from three cameras used for the axle detection network and a wider range of camera locations for the speed network. These images were provided by Q-Free and taken from a small selection of existing cameras. This means that the data is not entirely representative of the data the network is likely to see when in use. When installed at a roadside, the camera used will either be an existing camera carefully selected for this purpose or a camera specifically installed for use with this system.

Video footage was also necessary to evaluate the effectiveness of the speed calculation algorithm. Due to limited availability of suitable footage, testing was performed using one ten second video clip, as well as videos created from still images taken from the image dataset.

A key element of this research is the determination of important factors to consider when choosing a camera for operation. This is especially important for the axle detection model as the system must be capable of identifying small objects with a very high accuracy, a challenging task for a neural network. Therefore, an evaluation will be performed of the suitability of the three cameras selected for the purposes of axle detection. Examples of images from each of these three cameras are presented in Fig. 4.

The cameras were chosen to maximise the variety of the dataset whilst ensuring only images where wheels can be clearly distinguished were used. For example, cameras facing too directly down a road could not be used as vehicles were imaged from the front, with the side view necessary to detect axles not visible. Each camera used produced a dataset with unique challenges to be addressed by the network.

Dataset 1 is the largest of the three, containing 1536 total images. Adverse weather conditions are seen in many of the images from this dataset. This poses additional challenges for the axle detection network as wheels are often not clearly defined. This blurring of the undercarriage of vehicles is especially prevalent for lorries, such as the example seen in Fig. 4a (top). Individual wheels can only be seen in some images as indistinct patches of a different shade of grey, posing a significant challenge for the axle



(a) Dataset 1 (b) Dataset 2 (c) Dataset 3

Fig. 4 Example images taken from each of the three cameras used for axle detection

detection network. This dataset also features a high density of traffic, with partial and full occlusion common and is the only dataset taken from a dual carriageway in an urban environment.

Dataset 2 contains 1062 images. It is taken from a camera with a wide angle lens. This causes problems for several reasons. Firstly, for most of the range in which vehicles are visible the wheels are small and thus difficult to detect. CNN-based object detectors can struggle in cases where the objects to be detected are of substantially different scale and so the detection of very small wheels may be difficult to achieve. The second issue with the angle of this camera is that vehicles close to the camera, in the ideal position for wheel detection, are often only partially visible. Clearly this makes obtaining an accurate axle count impossible. For this dataset the window in which a vehicle is far enough from the camera to be fully visible but close enough for wheels to be clear and relatively large is very small. This window is especially small for longer vehicles which need to pass further from the camera before the entire vehicle is visible.

Furthermore, due to the wide angle, significant barrel distortion can be seen in the images of this dataset. This term refers to the effect where straight lines in an image appear curved, forming the shape of a barrel [11]. The effects of this distortion may impact the performance of the model when detecting objects. Figure 4b shows an example where the effects of barrel distortion can be clearly seen (top) and an example of a vehicle at a distance where wheels are too small to be reliably detected (bottom).

Dataset 3 is the only axle detection dataset taken from a three-lane road and the only dataset in which only one side of the road is visible. It is also the smallest dataset, with only 448 images. The images are taken from a camera located in the central reservation of a motorway and only show the leftmost two lanes on one side of the road. The axle detection model should perform well on this dataset as all vehicles are close enough to the camera for wheels to be clear and occlusion is less common than in Dataset 1. However, as this dataset will only allow the model to classify vehicles in two lanes, the use of footage from a similar camera may have somewhat limited use for real-world applications.

Another aspect of each dataset which may affect the model's success at axle detection is the type of vehicles present. Dataset 2 contains a high number of farm vehicles and heavy goods vehicles transporting farm equipment whilst Dataset 3 has a significant number of vehicle transporters carrying cars. These types of vehicles may skew the results as the network must be able to identify wheels which differ significantly in appearance as well as learn to ignore vehicles and wheels when being transported but identify them in all other situations. In contrast, Dataset 1 contains primarily cars, vans, lorries and buses and other vehicle types are uncommon.

The validation accuracy across the three datasets was used to make some preliminary conclusions as to the key factors in deciding whether a particular camera is suitable for axle detection. It was found that the effects of the distortion seen in Dataset 2 were significant enough to make this camera setup unsuitable. For this reason, model performance at axle detection will be evaluated using results across only Datasets 1 and 3 as these are taken from cameras found to be suitable for the application. The performance across all three datasets will be further discussed in Section 4.4 and conclusions made about some considerations for appropriate camera selection.

3.2 Training

The darknet framework from AlexeyAB [4] was used to configure and train the two networks.

The framework included data augmentation tools, which were used to randomly vary the saturation, exposure, hue, resolution and aspect ratio of the images, therefore artificially increasing the size of the dataset. The initial convolutional weights used were pre-trained on Imagenet [19]. The three anchor boxes were calculated using *k*-means clustering on the dataset.

Input sizes of 608×608 pixels to train the axle detection network and 416×416 pixels to train the speed measurement network were chosen. The higher resolution is necessary for the axle detection network as the inclusion of a wheel class means that smaller objects must be identified in each image and for reliable operation it is desirable that these objects are represented by as many pixels as possible in the input image.

The models were trained for a total of 4000 iterations for the axle detection network and 6000 iterations for the speed measurement network. After training, the sets of weights with the highest reported mean average precision for the test dataset were used to evaluate the accuracy of the networks. Mean average precision (mAP) is a standard metric used to evaluate the performance of object detection networks. It is determined by calculating an average precision (AP) value for each class and then taking the mean across all classes [15].

All training and evaluation of network performance was performed using an Nvidia GTX 1080 with 8GB of VRAM.

3.3 Detection of Axle Counts

The output detections of the axle detection network were processed using code written in Python. The script assigns a wheel to a vehicle if the centre point of the wheel bounding box is within the vehicle bounding box.

In order to ensure that wheels are correctly assigned in the case of vehicle bounding boxes overlapping, two measures are taken. Before assigning wheels, vehicles are ordered by

ascending size as in most cases the further vehicle, to which wheels inside both boxes should be assigned, will be smaller. In order to account for cases where the more distant vehicle is larger, wheels are not assigned to a vehicle if the centre point of the wheel is in the top 20% of the vehicle bounding box. The measures taken were found to consistently assign wheels to the correct vehicle across all three datasets.

A key consideration for processing the output detections is the identification of cases of full or partial occlusion. A vehicle is considered potentially occluded by the code when its bounding box overlaps with that of another vehicle. This means that both the occluded vehicle and the vehicle in the foreground occluding it will be flagged as occluded. Furthermore, in some cases where the bounding boxes overlap there is no actual occlusion taking place. As many vehicles are incorrectly flagged as occluded in this manner, there is no simple way to ascertain which vehicles are actually occluded. Therefore, the approach of simply ignoring any vehicles marked as occluded is likely to result in very high error rates and so the alternative approach of producing a reasonable estimate of axle count for partially occluded vehicles will be followed.

A possible improvement would be to introduce a third class when training the network which would identify occluded vehicles where not all axles are visible. If this model could achieve high accuracy it would be possible to ignore all vehicles where occlusion makes an axle count impossible or, alternatively, flag those vehicles for which the axle count is less reliable due to occlusion. While this is a potential avenue for future research, it was decided to focus on training the two-class model for the purposes of this paper.

A second flag was set in the code to identify vehicles which may only be partially visible in the frame. Again, knowing for sure whether a particular vehicle is partially visible or if it is merely close to the edge of the frame is difficult. However, this is a less important issue than that of occlusion. When the system runs on a video feed with a tracking algorithm in place, the axle count for each vehicle can simply be predicted at a point where there is a sufficient gap between the vehicle and the edge of the frame. Thus, vehicles which are marked as partially visible can be safely discounted on the assumption that a prediction with a higher confidence would be possible with the vehicle in a different position.

The output code should ideally also be able to handle a degree of error in the bounding box predictions provided by the network. Several features were implemented to aid in this. Vehicles with only one axle do not exist. Therefore in cases where only one wheel is assigned to a vehicle, that vehicle will be predicted to have two axles. This means that two-axle vehicles will return a correct axle count provided that at least one wheel is detected. This will allow for a degree of error in the detection of wheels for these vehicles, set two axles as the default return for any vehicle where some wheels are detected, and even account for cases of partial occlusion in two-axle vehicles.

In order to account for some inaccuracy in the positioning of vehicle bounding boxes, wheels which are not initially assigned to any vehicle will be assigned to the vehicle the shortest horizontal distance away. The risk of this approach is that background errors (when an element of the background is incorrectly identified as an object) have the potential to change a correct axle count to an incorrect one. However, one strength of YOLO as an object detector is that background errors are relatively rare and so their influence on overall accuracy is minimal.

3.4 Camera Calibration

A calibration tool was developed using Python and OpenCV [3] to enable the user to easily calibrate a camera for speed calculation [12]. It is assumed that the camera is in a fixed position and that distortion effects, such as barrel distortion, were negligible. It is also assumed that the road is flat and straight, and that the road markings used for calibration have parallel features and conform to the government road specifications. The dimensions of the road markings on UK roads can be found in the Traffic Signs Manual from the Department for Transport [2].

The user first selects the vertices of a rectangle *ABCD* by clicking on an image from the camera. The rectangle should use road markings for reference as shown in Fig. 3. To reduce the user error, the points can be inputted multiple times to form an average location for each. The width of the rectangle is specified and then the software calculates the camera parameters using Fung and Yung's method (FY). The pan angle is then used to check whether the parameters should be recalculated using He and Yung's (HY) method or HY′ with the *ABCD* labels rotated by 90 degrees as shown in Table 1. The camera parameters are then written to file to be used by the main program to calculate the speed of vehicles.

3.5 Speed Calculation

The tracking and speed calculations are implemented using AlexeyAB's C++ API [4] and a modified version of his C++ example file [12].

Tracking is used to identify each vehicle present in the video and associate a list of bounding box locations with it. The tracking algorithm used is the OpenCV Kalman filter. This standard implementation of the Kalman filter did result in some erroneous behavior as the originally detected bounding boxes would only be used to update the filter during the update step and were then discarded. Therefore, the location of the bounding box as predicted

by the network was lost and replaced by that predicted by the Kalman filter which was less accurate, hence the speed measurements were affected. To remedy this behavior, rather than changing the API, the Kalman filter predictions were matched to the bounding box predictions of the network. For each network predicted box, it was compared to every Kalman prediction and a score for each pairing was calculated. This score is given by the intersection area over the area of the network predicted box. For the pair with the highest score the tracking ID from the Kalman filter was applied to the corresponding network predicted bounding box.

Once the Kalman filter has identified which bounding boxes correspond to a particular vehicle, the average speed of each vehicle passing the view of the camera can be calculated. Given that each camera only covers a relatively short section of road, and that the function of this system is to monitor general traffic conditions rather than individual vehicles, it is deemed reasonable to assume that all vehicles are moving in a straight line and at a constant speed. This allows for speed to be simply calculated without the need for tracking specific movement on the $Z=0$ plane.

In order to calculate speed, the bounding boxes predicted must first be converted to a position on the $Z=0$ plane. To do this, the camera parameters are loaded from the file generated by the calibration tool. Then, for each detected vehicle, the corner of the bounding box closest to the ground plane is selected. The corner coordinates (x_q,y_q) in the image plane are then converted to a set of corresponding coordinates (X_Q,Y_Q) on the $Z=0$ plane in the real world scene using the following transformations:

$$X_Q = \frac{\mu_s h \cos p + \left\lfloor \frac{v_s h \sin p}{\sin t} \right\rfloor}{x_q \cos t \sin s + y_q \cos t \cos s + f \sin t} \tag{9}$$

$$Y_Q = \frac{\mu_s h \sin p - \left\lfloor \frac{v_s h \cos p}{\sin t} \right\rfloor}{x_q \cos t \sin s + y_q \cos t \cos s + f \sin t}, \tag{10}$$

where

$$\mu_s = x_q \cos s - y_q \sin s \tag{11}$$

$$v_s = x_q \sin s + y_q \cos s. \tag{12}$$

The first time a new vehicle is detected, its tracking ID, initial coordinates (X_Q,Y_Q) and the current frame number f_q are recorded. Each time the same vehicle is detected, its distance d from (X_Q,Y_Q) and the elapsed number of frames t_f since f_q are calculated. Using these, the average speed v of the vehicle from when it was first detected is

$$v = \frac{d}{t_f \cdot \frac{1}{FPS}} = d \cdot \frac{FPS}{t_f} \tag{13}$$

where FPS is the frames per second of the video feed.

4 Results and Discussion

4.1 Axle Detection

The neural network for axle detection achieved a mAP of 95.14% across Datasets 1 and 3, with a vehicle AP of 97.50% and a wheel AP of 92.77%, working at a speed of 20 FPS. Figure 5 contains examples of typical detections produced by the axle detection network.

(a) Dataset 1

(b) Dataset 3

Fig. 5 Typical output of the axle detection network for validation datasets 1 and 3, including both correct and false scenes

The performance of the model at the task of identifying vehicles is very good, especially considering that cases where vehicles are not correctly identified are generally due to occlusion or distance of the vehicle from the camera. In other words, the model rarely fails to correctly identify vehicles and almost always in cases where obtaining an accurate axle count would not be possible even with a correct vehicle detection.

The network's slightly lower accuracy on the task of wheel detection is not unexpected. Wheels are simpler, smaller and in many cases less distinct in images than vehicles. These factors all combine to make wheels more challenging for a neural network to reliably detect. However, the errors made by the network follow predictable patterns, many of which can be accounted for when processing the output detections for a final axle count.

An important requirement for the neural network is real time speed performance. One feature of the YOLO network is that, while training must take place at a fixed input image resolution, detections can be performed at a lower resolution if required. This allows for a fully trained model to be adjusted to alter speed and accuracy characteristics at test time. Figure 6 shows the speed and accuracy performance of the axle detection network for a range of test-time resolutions.

At the highest resolution of operation, the network operates at 19.7 FPS with an mAP of 95.14%. This may be sufficient speed for relatively low quality surveillance camera setups. However, the industry standard frame rate for high quality video footage is 30 FPS [24]. In the case of operation using a camera with the standard frame rate it would therefore be necessary to reduce the resolution of the network. As can be seen from Fig. 6, a reduction to input width and height resolution of 512 pixels is sufficient to increase the speed to 32.8 FPS without significantly reducing the accuracy of the network. This indicates that real-time performance is feasible for any realistic camera setup. For the purposes of this paper, general results are presented using the maximum width and height resolution of 608 pixels.

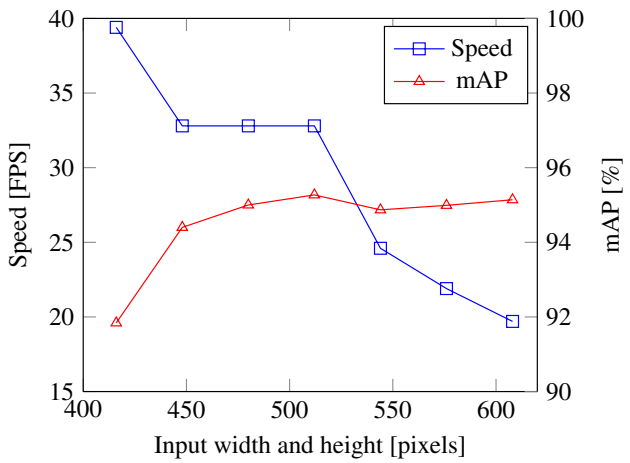


Fig. 6 Measurements of speed and mAP for detections performed by the axle detection network at varying resolution

Mean average precision is the standard metric for evaluating the performance of neural networks on object detection tasks and will be further discussed with relation to the different camera setups tested in Section 4.4. For the remainder of this section, however, the performance of the entire axle-count system will be evaluated using a measure of the percentage of vehicles present in the validation dataset for which a correct axle count was produced. This is more meaningful than mAP when considering the feasibility of the system as a whole for real-world implementation as the efficacy of the processing of network outputs is taken into account in addition to performance of the neural network.

In order to maximise the number of vehicles correctly counted, measures were taken to increase the likelihood of an accurate count, even when the network did not produce accurate bounding boxes.

False positives on background from the neural network are rare, and so the most common form of error on two-axle vehicles is only one wheel being correctly detected. As previously discussed, two-axle vehicles will be correctly counted whenever at least one wheel is detected thus neutralising this error entirely. Additionally, it was found that another frequent error made by the network was identifying one too many wheels in the rear section of a six-axle lorry, as seen in Fig. 5b (bottom-left). As lorries are the primary focus of this system and seven-axle vehicles are extremely rare, it was decided to return an axle count of six whenever seven axles are assigned to a vehicle.

In the future a more sophisticated system for identifying these false positive wheel detections will be introduced. However, at this stage a simple solution which eliminates the majority of error is sufficient for a preliminary investigation into the potential accuracy of the system. By disallowing both one-axle and seven-axle vehicles, the most common forms of error in both cars and vans, and lorries are eliminated.

It was also necessary to make decisions about which vehicles to include when presenting results in order to effectively measure the performance of the system. It was decided that the most meaningful metric of success would be the percentage of vehicles where a correct axle count is possible which were counted correctly by the system.

Vehicles which are partially visible in the frame to the extent that not all axles are visible are excluded, as well as vehicles which are judged to be too distant for wheel detection to be realistic. Thus, only vehicles in the region where axle counts would be expected to take place when working on a video feed are included. Vehicles which are fully occluded are also excluded from the results as these are

considered cases of unavoidable error. Results are given for both the case where partially occluded vehicles are included and the case where they are excluded. The comparison of results in these two cases will aid in understanding the significance of partial occlusion as an issue, and allow an informed decision to be made as to the importance of further work in improving the performance of the network in cases of occlusion.

The model achieved a correct axle count on 91.6% of vehicles across Datasets 1 and 3 when partially occluded vehicles were included and 93.3% when they were excluded. Table 2 shows the full breakdown of results, with results divided by true axle count of the vehicle included.

The significance of cases of partial occlusion can clearly be seen here. Across both datasets the inclusion of partially occluded vehicles corresponded to a roughly 2% decrease in overall accuracy.

Table 3 shows the number of vehicles with each axle count present in each dataset.

The distribution of vehicle types across Dataset 1 is a significant issue. Two-axle vehicles made up 98% of the vehicles present in the validation set. This is likely due to the location of the camera, in an urban area where vehicles with more than two axles are rare. This extremely uneven distribution also raises the possibility that, in a similar urban environment, the axle-count network may not add significant value to the base vehicle classification system.

The results for Dataset 3 are somewhat limited by the small sample size but performance across this dataset is extremely promising. Only three of the 45 vehicles present in the validation set were incorrectly identified. Two are presented in Fig. 5b: one six axle vehicle (top-right) is occluded to the extent that only three axles are visible and the only four-axle vehicle in the validation set (bottom-right) is a vehicle transporter on which a wheel of a vehicle being transported has been detected. Both of these are cases where some error is expected. The final incorrectly counted vehicle was a seven-axle vehicle. As explained above, it was decided that error when encountering rare seven-axle vehicles is acceptable if it allows for the reduction of error when identifying six-axle vehicles. Across this dataset, every non-occluded six-axle vehicle was correctly identified, including those such as the example shown in Figure 5b (bottom-left) where there is error in the neural network output.

Table 2 Percentage of vehicles of each axle count in validation sets 1 and 3 where predicted axle count is accurate

	Partially Occluded Vehicles	Number of Axles			
		2	3-5	6-7	All
Dataset 1	Included	91.8%	60.0%	100%	91.3%
	Excluded	93.5%	60.0%	100%	93.0%
Dataset 3	Included	100%	0.0%	85.7%	93.3%
	Excluded	100%	0.0%	92.3%	95.5%
Datasets 1 and 3	Included	92.6%	50.0%	92.9%	91.6%
	Excluded	94.2%	50.0%	100%	93.3%

4.2 Tracking

The Kalman filter successfully tracked vehicles across the video frame in most cases. However, periods of occlusion caused issues as the filter would not recognise the object again. This issue was most prevalent during periods of dense traffic. Measures can be taken to reduce the effects of occlusion by carefully selecting the camera position. The higher the camera is installed, the less occlusion will occur. The camera angle relative to the direction of the road will also have an effect. The closer the camera is to being perpendicular to the road, the more vehicles will obstruct those in other lanes. The closer the camera is to being parallel to the road, the more likely that vehicles are obstructing vehicles in the same lane. Having the camera close to parallel would also mean that the vehicles would rapidly reduce in size, because the vanishing point would be closer. Therefore the size of the region where cars are successfully detected is reduced. For these camera angles, the Kalman filter caused some vehicle identities to jump across the road nearer the vanishing point. However, this was easily remedied by cropping out the other side of the road.

In this work, the Kalman filter was only implemented for use with the speed calculation model, however there is great potential for improvements in the accuracy of the axle detection model through the implementation of a tracking algorithm.

Table 3 Number of vehicles of each axle count present in validation datasets 1 and 3

	Number of Axles						
	2	3	4	5	6	7	All
Dataset 1	293	3	1	1	1	0	299
Dataset 3	30	0	1	0	13	1	45
Datasets 1 and 3	323	3	2	1	14	1	344

4.3 Vehicle Speed

The evaluation of the accuracy of the calculated vehicle speeds is unfortunately limited by the lack of available ground truth speed measurements to use for calibration and evaluation of results. This work, therefore, represents a proof of concept; the development of a functional system which returns results typical of the types of roads being monitored but which would likely require some tuning using reliable speed measurements in order to guarantee high accuracy.

The calculated speeds were consistently in the region of expected speeds for the road and vehicle class. The speed measurements for vehicles in the overtaking lanes were faster than for the vehicles in the slow lane, as expected. This can be seen in the two frames from the video output in Fig. 7. Vehicle 9 is in the fast lane and traveling faster than vehicles 7 and 8. All three vehicles are also traveling close to 70mph which is expected for this type of road.

The neural network trained for the speed estimation model achieved an mAP of 98% at the single feature detection task. Combined with effective tracking of vehicles across multiple video frames using the Kalman filter, error due to incorrect output from the neural network could be effectively eliminated.

The camera calibration method was accurate, however, it relied on knowing the accurate dimensions of the road markings. Occasionally the road markings deviated from the government road specifications and in these cases the speed measurements were all uniformly affected due to the constant calibration error. The camera calibration was also only valid for short periods of time, as the video was not taken with a tripod and therefore the camera was not in a truly fixed position.

Using the corner of the bounding box closest to the road surface did not result in the most accurate solution. This is because YOLOv3 best predicts the bounding box center and only fits the box shape using an anchor box, resulting in loosely fitting bounding boxes. Therefore, the corner of the box was never perfectly on the $Z = 0$ plane in the world scene. The corner of the bounding box would also remain relatively stationary while the object was coming into view at the edge of the frame. This would result in errors in the average speed.

For a network size of 416×]416, using a HD 1920 × 1080px video as input, the model reached a peak speed of 36 FPS whilst both displaying the output detection as a video stream and performing the tracking and speed calculations. Using the road markings, the length of the road visible in Fig. 7 can be estimated to be 60m. Therefore, a vehicle traveling at 70mph will cross the video frame in just under two seconds and will be captured by the camera in approximately 60 frames.

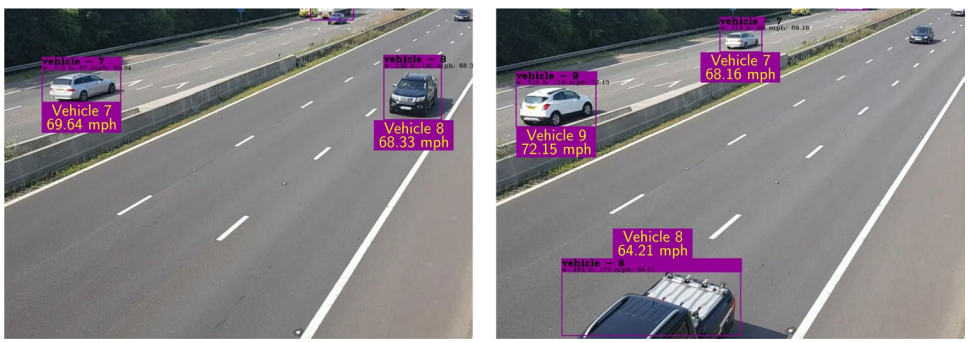
As the model's speed is faster than the video's 30 FPS, all occurrences of the vehicle will be used. For calculating the vehicle speed, an absolute minimum of two occurrences, or just over 1 FPS, is necessary. However, this would not be sufficient to track the vehicle, which is necessary to calculate the speed. Therefore, the more occurrences the better and currently the model is using the maximum number of captures as it is running faster than the camera.

For real-time application, the network would be running on less powerful hardware. However, the model would not be required to display the output as a video stream, which would further increase the computational performance. Therefore, the network shows very strong potential to perform at the speeds required for real-time application.

4.4 Investigation into Appropriate Camera Selection for Axle Detection

Average precision values for the axle detection network across each of the three validation datasets are presented in Table 4.

It can be seen that vehicle AP is fairly consistent across all three datasets whereas wheel AP varies considerably. Whilst Datasets 1 and 3 return high values for wheel AP of 92.04% and 95.05% respectively, model performance on Dataset 2 is considerably poorer, with only 73.51% wheel AP achieved.



(a) Frame 50 (b) Frame 77, approximately one second later

Fig. 7 Two successive captures from the video output

Table 4 Measures of bounding box prediction accuracy for the axle detection network across each validation dataset

	Vehicle AP	Wheel AP	mAP
Dataset 1	97.25%	92.04%	94.65%
Dataset 2	98.53%	73.51%	86.02%
Dataset 3	97.80%	95.05%	96.42%
Datasets 1 and 3	97.50%	92.77%	95.14%

As outlined previously, images taken from some cameras could be immediately discounted due to camera angles at which wheels are not clearly visible. A more thorough and systematic investigation of the significance of camera positioning and angles could be pursued to identify the ideal setup for axle detection. However, this would require the compilation of a labelled dataset containing images from a high number of cameras, with factors such as camera position relative to the road and camera yaw and pitch angles, varied in order to provide meaningful results related to the specific setup. Such an investigation is outside the scope of this paper but is a potential avenue for future research.

In this study, evaluation of model performance across the three datasets has revealed some more subtle factors which should be considered and a general approach is outlined for the selection of an appropriate camera for use with the axle-detection system.

The poor performance of the model on Dataset 2 suggests that the effects of barrel distortion are more significant than initially expected. Detections run on this dataset showed that in many cases wheels which were perfectly clear to the human observer were not identified by the model. In contrast, across Datasets 1 and 3 wheel detections were inaccurate primarily in cases where wheels were small, unclear, blurred or close together and so detection could be reasonably expected to be more challenging.

Further work would be necessary in order to investigate the possibility of limiting the impact of barrel distortion, through the means of additional data augmentation techniques or further training with more distorted data. However, at this stage in the research it is simply recommended to choose cameras which do not display barrel distortion in order to ensure optimal performance of the axle detection network.

It was decided that both Dataset 1 and Dataset 3 were suitable for this application. However, slight differences in performance can be observed between the two sets. One key reason for this is weather conditions. Dataset 3 contains only images taken in clear weather, whereas Dataset 1 contains some images taken in sunny conditions and others where rain severely impacts visibility. Table 5 shows the average precision values for Dataset 1 when data is segregated by weather conditions.

Table 5 Measures of bounding box prediction accuracy of the axle detection network for dataset 1 under different weather conditions

	Vehicle AP	Wheel AP	mAP
Clear	97.26%	97.42%	97.34%
Rain	97.04%	91.38%	94.21%

As can be seen, the impact of weather on the accurate detections of wheels is significant. In wet conditions spray from the road surface can partially obscure the underside of vehicles, making distinction of individual wheels more difficult. Figure 5a (bottom) shows examples of similar three-axle vehicles in similar positions but different weather conditions, where the image taken in wet conditions (left) was not correctly labelled and the image taken in sunny conditions (right) was. From these images it can be clearly seen that even slight changes in weather conditions can have a significant impact on model performance.

Although there is no data available with other forms of adverse weather present, it can be assumed that other conditions which affect visibility, such as fog or snow, would cause similar issues. This is one of the major disadvantages of video-based classification as the effects of reduced visibility can be extremely difficult to overcome. Any potential user of the axle detection system should be informed that performance is likely to suffer slightly in adverse weather conditions. However, the output processing of the wheel detections produced by the network has been shown to allow for a degree of inaccuracy from the network, demonstrated by the consistently high percentage of vehicles across Datasets 1 and 3 for which an accurate axle count was returned. Additionally, the impact of rain on the task of vehicle detection was found to be minimal and so it can be expected that the performance of the speed detection system will not be severely impacted by adverse weather conditions.

Another key difference between Dataset 1 and Dataset 3 is the number of partially or fully occluded vehicles. In Dataset 1, 18.4% of vehicles visible in the validation set are occluded heavily enough that at least one wheel is not visible, compared to only 2.2% for Dataset 3. This difference is caused by two factors.

Firstly, Dataset 1 is taken from an urban dual carriageway on which there is relatively heavy traffic. Dataset 3, in contrast, is taken from a motorway camera and traffic is lighter. Road type is not a factor which can necessarily be accounted for when selecting a location for this system as town councils are likely to desire data from specific roads and traffic flow clearly cannot be controlled.

The second factor causing more significant occlusion in Dataset 1 is the height at which the camera is installed. The camera used for Dataset 1 is relatively low down. While this does provide some advantages as vehicles are seen from a

view which is fairly close to side-on, giving a good view of wheels, it also means that many vehicles are directly between the camera and other vehicles.

The higher positioning of the camera for Dataset 3 means that more of the top of vehicles is visible and less of the sides. However, it means that vehicles must be closer together and large for occlusion to occur. The height at which cameras are installed is a factor which can be controlled to an extent. When choosing a camera for use with this system, increased elevation is desirable provided that wheels are still clearly visible.

However, it is important to note that the inclusion of partially occluded vehicles had similar impact on overall model accuracy for both datasets despite the discrepancy in number of occluded vehicles present. This is likely due to the fact that the majority of occluded vehicles in Dataset 1 have two axles and two-axle vehicles can be correctly identified even when partially occluded. In contrast, a higher proportion of partially occluded vehicles in Dataset 3 have more than two axles and so cannot be correctly identified unless all axles are visible. Thus, it is reasonable to conclude that partial occlusion is a less severe issue on roads where two-axle vehicles are more common and so a lower mounted camera may indeed be appropriate in urban areas similar to that of camera 1 where many-axled vehicles are uncommon.

5 Conclusions

In summary, the axle detection system proposed in this paper produced an accurate axle count for 93.3% of vehicles in which all axles are visible, with superior performance demonstrated based on well chosen camera placement. The system is capable of real-time operation, with a minimum speed of 19.7 FPS and speeds exceeding 30 FPS achieved without significant impact on accuracy when operating at a lower resolution.

The speed detection model achieved very high accuracy for the task of object detection with 98% mAP. The system is capable of real-time operation, with the model being able to keep up with a 30 FPS video and calculated speeds were in the region of expected speeds for the road and vehicle type.

The systems presented in this paper represent early prototypes of the final solutions envisaged, with significant further work required to produce fully operational modules.

Operating the axle detection system on video footage using a tracking algorithm will allow for several potential improvements in terms of accuracy. If the section of a frame in which wheels are most clearly visible can be identified for each camera, detections can take place only for vehicles in this section. This will reduce error as axles are counted only when the vehicle is in the ideal position for detection. Further improvements could be made by averaging the axle count for every frame in which the vehicle is in the detection zone in order to account for random error in detection.

To improve the tracking for the speed calculation system, the Deep SORT algorithm [29] should be used. By including the object features, occluded objects would be tracked much more successfully.

The camera calibration method is very good in principle. However, its accuracy relies on the road markings and this is not reliable in practice as they do not always conform to the government road specifications. The calibration method presented by Sochor et al. [25] shows promise, as it is fully automatic, requires no user input and does not rely on any reference markers.

To improve the accuracy of the speed measurements, 3D bounding boxes should be used. The center of the ground plane of the 3D bounding box would be a much more accurate marker than the corner of the 2D bounding box that is closest to the $Z = 0$ plane. Using 3D boxes would also open up the possibility of being able to output the size of the object without much additional effort. The accuracy of the model should also be investigated using ground truth speed data.

Acknowledgements The authors would like to thank Q-Free for access to resources and additional support without which this research would not be possible.

Declarations

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. (2013) Environmental protection uk: Impacts of car pollution. https://www.environmental-protection.org.uk/policy-areas/air-quality/air-pollution-and-transport/car-pollution/, Accessed: 22/10/2019
2. (2019) Traffic Signs Manual, Chapter 5: Road markings. Department for Transport
3. (2021) Opencv. https://opencv.org/, [Accessed February 2021]
4. Alexey, A.B.: Darknet. https://github.com/AlexeyAB/darknet, Accessed: 15/3/2020 (2016)

5. Avery, R., Wang, Y., Rutherford, G.: Length-based vehicle classification using images from uncalibrated video cameras. In: Proc. IEEE International Conference on Intelligent Transportation Systems (ITSC'04), Washington, WA, USA, pp 737-742 (2004)
6. Bautista, C.M., Dy, C.A., Mañalac, M.I., Orbe, R.A., Cordel, M.: Convolutional neural network for vehicle detection in low resolution traffic videos. In: 2016 IEEE Region 10 Symposium (TENSYMP), pp 277–281 https://doi.org/10.1109/TENCONSpring.2016.7519418 (2016)
7. Fung, G., Yung, N., Pang, G.: Camera calibration from road lane markings. Opt. Eng. **42**, 2967–2977 (2003a)
8. Fung, G., Yung, N., Pang, G.K.H.: Camera calibration from road lane markings. Opt. Eng. **42**, 2967–2977 (2003b)
9. Girshick, R.: Fast R-Cnn. In: The IEEE International Conference on Computer Vision (ICCV) (2015)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
11. Grigonis, H.: Understanding lens distortion in photography (and how to fix it). https://expertphotography.com/what-is-lens-distortion/, [Accessed April 2020] (2019)
12. Gurr, F.B.: Final year project code. https://github.com/Francis-Gurr/Vehicle-Speed-YOLO, Accessed: 25/4/2020 (2019)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
14. He, X.C., Yung, N.: New method for overcoming ill-conditioning in vanishing-point-based camera calibration. Opt. Eng., 46 (2007)
15. Hui, J.: map (mean average precision) for object detection. https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173, [Accessed February 2021] (2018)
16. Kalman, R.: A new approach to linear filtering and prediction problems. J. Basic Eng. **82**, 35–45 (1960)
17. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. NIPS, 25 (2012)
18. Redmon, J.: Darknet: Open source neural networks in c. http://pjreddie.com/darknet/ (2016a)
19. Redmon, J.: Darknet pre-trained weights. https://pjreddie.com/media/files/darknet53.conv.74, Accessed: 15/3/2020 (2016b)
20. Redmon, J., Farhadi, A.: Yolo9000: Better, Faster Stronger. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
21. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv:1804.02767 (2018)
22. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 779–788 (2016)
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**, 1137–1149 (2017)
24. Security B: Fps. https://www.videosurveillance.com/tech/frame-rate.asp, [Accessed April 2020] (2015)
25. Sochor, J., Juranek, R., Herout, A.: Traffic surveillance camera calibration by 3d model bounding box alignment for accurate vehicle speed measurement. Comput. Vis. Image Underst. **161**, 87–98 (2017)
26. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going Deeper with Convolutions. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015)
27. for Transport D: A simplified guide to lorry types and weights. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/211948/simplified-guide-to-lorry-types-and-weights.pdf, [Accessed November 2019] (2003)
28. Uijlings, J., van de Sande, K., Gevers, T., Smeulders, A.: Selective search for object recognition. Int. J. Comput. Vis. **104**, 154–171 (2013)
29. Wojke, N., Bewley, A., Paulus, D.: Simple Online and Realtime Tracking with a Deep Association Metric. In: Proc. IEEE International Conference on Image Processing (ICIP'17), Beijing, China, pp 3645-3649 (2017)
30. Won, M.: Intelligent traffic monitoring systems for vehicle classification: a survey. arXiv:1910.04656 (2019)
31. Yu, S., Wu, Y., Li, W., Song, Z., Zeng, W.: A model for fine-grained vehicle classification based on deep learning. Neurocomputing **257**, 97–103 (2017)
32. Zhang, Z., Tan, T., Huang, K., Wang, Y.: Practical camera calibration from moving objects for traffic scene surveillance. IEEE Trans. Circuits Syst. Video Technol **23**, 518–533 (2013)
33. Zheng, Y., Peng, S.: A practical roadside camera calibration method based on least squares optimization. IEEE Trans. Circuits Syst. Video Technol. **15**, 813–843 (2014)
34. Zhuo, L., Jiang, L., Zhu, Z., Li, J., Zhang, J., Long, H.: Vehicle classification for large-scale traffic surveillance videos using convolutional neural networks. Mach. Vis. Appl. **28**, 793–802 (2017)