Traffic Image Analysis using Deep Learning for Safe Vehicle Navigation in Roads Controlled by Police

Samitha P. Randeniya
Postgraduate Institute of Science,
University of Peradeniya,
Peradeniya 20400, Sri Lanka

Ruwan D. Nawarathna
Department of Statistics and Computer Science,
University of Peradeniya,
Peradeniya 20400, Sri Lanka

ABSTRACT
Driving in urban environments where traffic is controlled by police presents significant challenges for both human drivers and autonomous vehicles (AVs). Interacting with pedestrians and traffic police officers in such settings requires sophisticated communication and understanding of their intentions. Such interactions are critical because pedestrians are the most vulnerable road users. Traffic conditions, driving scenarios, police signals, and pedestrian behaviours can vary widely between countries. Understanding these behaviours and signals is not straightforward and depends on numerous factors such as pedestrian demographics, traffic dynamics, and environmental conditions. In different countries, pedestrians may use hand signals to stop traffic when crossing the road, and traffic police officers may control vehicles during traffic jams, traffic light malfunctions, and at zebra crossings. The common signals used are STOP and GO. Convolutional Neural Networks (CNNs), a deep learning technology, are widely applied in areas such as computer vision and object recognition. This study explores how an AV can identify the STOP signal from a pedestrian or traffic police officer amidst other pedestrians and officers on the road. A model is proposed using a custom dataset and a CNN-based multi-class object detection framework. Additionally, the model can identify pedestrians crossing at zebra crossings. To test the proposed model in real-time, a compact autonomous vehicle was designed using a Raspberry Pi, a popular microcontroller for small-scale projects. This prototype AV can detect five classes of objects and respond by moving forward or stopping based on the relevant signals. The study focused on the traffic conditions in Sri Lanka, where the case study was conducted.

General Terms
Artificial Intelligence, Self Driving Cars, Computer Vision.

Keywords
Deep Learning, Traffic Image Analysis, Object Detection, Safe Vehicle Navigation, Single-stage model

1. INTRODUCTION
In the past decade, as the concept of self-driving vehicles became a hot topic, engineers and scientists have been striving to achieve vehicle autonomy by eliminating the need for human control. This technology can assist drivers or even replace them entirely. Autonomous vehicles (AVs) can directly impact society at both micro and macro levels [1]. However, implementing AVs presents immense challenges for researchers and scientists, as it requires the vehicles to perform with the accuracy and efficiency of human drivers [2]. Driving is not only a dynamic control task but also a social activity that necessitates mutual understanding among road users to ensure smooth traffic flow and safety for all [3]. Interaction between AVs and pedestrians is crucial, as it helps pedestrians navigate crossings safely while enabling AVs to avoid accidents. When developing a self-driving car, various factors must be considered, such as pedestrians randomly crossing the road and traffic police officers managing traffic, as illustrated in Figure 1.

a) b)

Fig 1: Traffic Police STOP sign (a) and some pedestrians randomly crossing the road (b)

In this study, the self-driving car must be capable of detecting traffic police officers' signals and responding accordingly. Additionally, some pedestrians use hand signals to indicate to the vehicle to stop so they can cross the road, as shown in Figure 2. Moreover, when an AV approaches a zebra crossing, it must be particularly attentive, as there is a possibility of pedestrians crossing. In such cases, the AV must stop before the zebra crossing line and wait until pedestrians have safely crossed the road.

Fig 2: A pedestrian signaling to stop

The study focuses on detecting hand signals from traffic police officers and pedestrians, as well as recognizing zebra crossing lines. The primary objective is to develop a deep learning-based model capable of identifying pedestrians and police officers, including their hand signals, to aid navigation for self-driving cars on roads where people cross randomly, and traffic is controlled by officers. To achieve this, two advanced deep learning object detection models, Faster R-CNN and SSD, are trained. Data collection occurs in three distinct scenarios: indoor, outdoor, and through images extracted from the Joint Attention in Autonomous Driving (JAAD) dataset [23],

offering diverse backgrounds, lighting conditions, and global road conditions. Additionally, a prototype car controlled by a Raspberry Pi is constructed to deploy and test the models in real-time operations, fulfilling the second objective of the study.

1.1 Understanding pedestrian behaviour

Some modern studies still rely on questionnaires, especially when measuring general attitudes towards various aspects of driving, such as crossing in front of autonomous vehicles [4]. The effects of vehicle speed and distance are also examined separately in the literature. It has been shown that an increase in vehicle speed impairs pedestrians' ability to accurately estimate speed and distance [5]. Social norms influence the extent to which pedestrians obey the law, take risks, and communicate with one another [6]. Behavioral analysts have identified factors such as demographics [7], pedestrian state, and traffic characteristics as significant in pedestrian decision-making.

1.2 Study of characteristics among different pedestrians

One of the key dynamic factors is gap acceptance or how much gap in traffic (typically in time) that pedestrians consider safe to cross. Gap acceptance depends on two dynamic factors, speed of the vehicle and vehicle distance from the pedestrian. The combination of these two factors defines Time To Collision (or Contact) (TTC), or how far the approaching vehicle is from the point of impact [8]. As for pedestrians, it is shown that the majority of pedestrians tend to pay attention to the frequency of which may vary depending on the crosswalk delineation such as the presence of traffic signals or zebra crossing lines [7] before crossing. Some findings suggest that when pedestrians make eye contact with drivers, the drivers are more likely to slow down and yield to the pedestrians [9].

Signals significantly influence pedestrians' level of caution [7]. In a study by Tom and Granie [10], it was shown that pedestrians look at vehicles 69.5% of the time at signalized intersections and 86% of the time at unsignalized intersections. The authors also note that pedestrians' trajectories differ at unsignalized crossings; specifically, they tend to cross diagonally when no signal is present. Furthermore, pedestrians are found to rely more on the distance of the approaching vehicle when crossing; within the same TTC, they cross more frequently when the vehicle's speed is higher. Some scholars investigate the relationship between pedestrian waiting time before crossing and gap acceptance.

1.3 Pedestrian intention estimation

The task of estimating pedestrian intentions is typically approached as a tracking problem in intelligent transportation research. Different methods are used by traffic participants to communicate with each other. For instance, pedestrians may employ eye contact (gazing/staring), subtle movements toward the road, hand gestures, smiles, or head nods. Conversely, drivers may use flashing headlights, hand waves, or eye contact [11].

Additionally, some researchers suggest that changes in vehicle speed can indicate the driver's intention [7]. The stopping behavior of vehicles may also serve as a communicative cue. Studies indicate that when drivers stop their cars significantly before the legally required stopping point, they are signaling their intention to yield the right of way to others [12]. An innovative dataset addresses a critical aspect of autonomous driving: the joint attention required between drivers and pedestrians, cyclists, or other drivers. This dataset was created to illustrate the varied behaviors of traffic participants.

1.4 Detection of police hand signals

Limited research has been undertaken on the detection of police hand signals, with only two notable works discussed here. One study delves into vision-based traffic police hand signal recognition in surveillance videos [13]. The other study focuses on instances where heavy traffic prevents the use of automatic traffic light systems, traffic control depends on traffic police gestures [14].

1.5 Detection of zebra crossings

The zebra crossing, a common road marking used instead of a pedestrian crossing in many countries, consists of multiple parallel white stripes in a 3D space. The study in [15] proposes a zebra crossing detection method for intelligent vehicles. The method is applied to a bird's-eye-view image known as an inverse perspective mapping image.

Pedestrian crossings, integral to transportation infrastructures, play a crucial role in safeguarding pedestrians' lives and property while maintaining orderly traffic flow. As a prominent feature in street scenes, detecting pedestrian crossings contributes to the reconstruction of 3D road markings and reduces the adverse effects of outliers in 3D street scene reconstruction. Monitoring the condition of pedestrian crossings is imperative due to wear and tear from heavy traffic flow. In this regard, an approach to automatic pedestrian crossing detection using images from a vehicle-based Mobile Mapping System is proposed in [16], along with an analysis of its deterioration and impairment.

1.6 Computer vision in self-driving cars

In [17], a vision-based detection system for self-driving cars, encompassing traffic signs, traffic lights, and pedestrians, has been proposed. Furthermore, an autonomous lane-keeping system utilizing end-to-end learning is introduced. The study presented in [18] elaborates on devices such as Lidar and Radar, which are based on computer vision and are integral to self-driving cars, providing detailed descriptions and definitions. Additionally, the paper discusses the workings of pathfinding in self-driving cars using computer vision.

1.7 Deep neural networks and machine learning algorithms used in self-driving cars

A discussion on machine learning algorithms used in self-driving cars and proposals for enhancements is presented in [19]. In [20], a network four times smaller than the PilotNet model and approximately 250 times smaller than the AlexNet model is proposed, aiming to achieve a significantly faster frame rate. This proposal focuses on developing a lightweight model for autonomous vehicles that operates more swiftly than current algorithms, with a comparison drawn between the suggested model and existing networks. Furthermore, [21] explores a comparison of AI-based self-driving architectures, including convolutional and recurrent neural networks, as well as deep reinforcement learning paradigms. These methodologies serve as the foundation for surveyed driving scene perception, path determination, behavior arbitration, and motion control algorithms. Additionally, the paper delves into current challenges encountered in designing AI architectures for autonomous driving, such as safety concerns, computational hardware limitations, and training data sources.

2. MATERIALS AND METHODS

The proposed methodology comprises five steps: data collection, data preprocessing, object detection model selection, and finally, designing a prototype car to implement the results, as illustrated in Figure 3. Each step is elaborated upon in detail below.

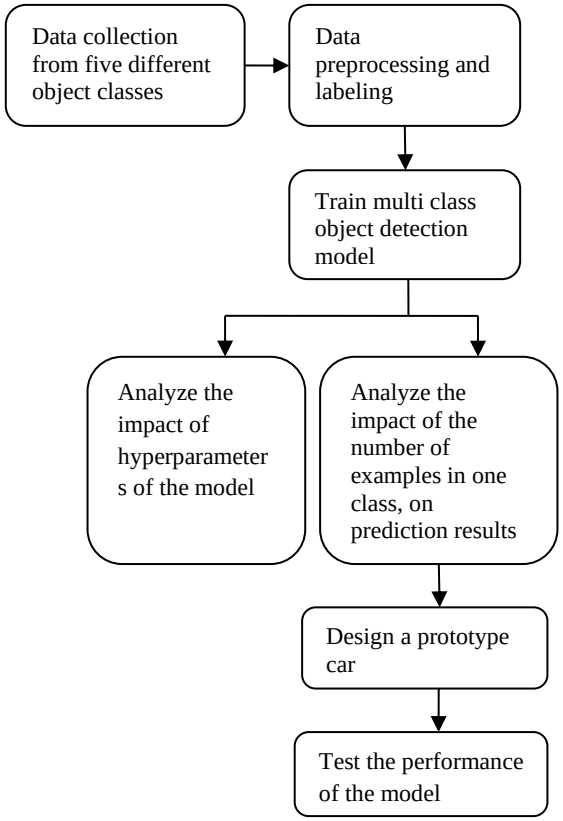


Fig 3: Steps of the proposed methodology

2.1 Data collection

The study considers images from five different object classes which are common in road and driving conditions. The names and descriptions of the five object classes are listed in Table 1. Datasets need to encompass various background and lighting conditions, as well as different environments, including local and international settings.

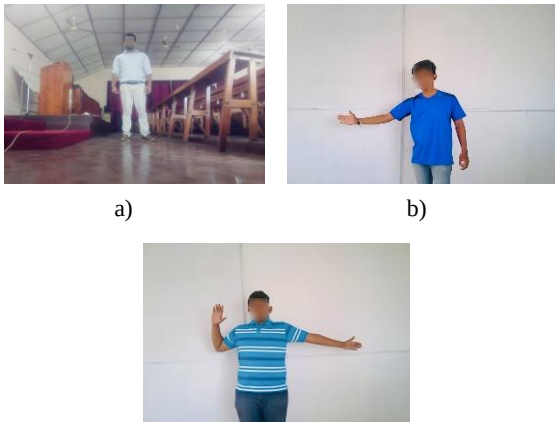
Table 1: Object classes in the data sets

Class	Description
Person	Person class represents pedestrians on the roads
Person Stop	Person Stop class includes images of pedestrians signaling to stop a moving vehicle
Police	Police class includes images of traffic police officers on the roads
Police Stop	Police Stop class contains images of traffic police officers gesturing to stop traffic
Zebra Crossing	Zebra crossing class includes images of pedestrian crossing lines on the roads

Therefore, data were collected in three different scenarios: indoor data driving, outdoor driving, and Joint Attention in Autonomous Driving (JAAD) data [23]. The collected images from each approach are then labeled as Data Set 01, Data Set 02, and Data Set 03.

2.1.1 Data set 01

Data Set 01 consists of images collected in indoor environments. Six members participated in the data collection, acting according to the desired gestures and signals. The data were gathered using a Windows 640 XL mobile phone camera and a 0.9MP laptop camera. This dataset includes images from three classes: Person, Person Stop, and Police Stop, as shown in Figure 4. In each recording session, only one camera was used at a time, and it was placed in various halls on different days, with varied backgrounds and lighting conditions. All images were captured from distances of 2m, 4m, and 7m from the cameras.

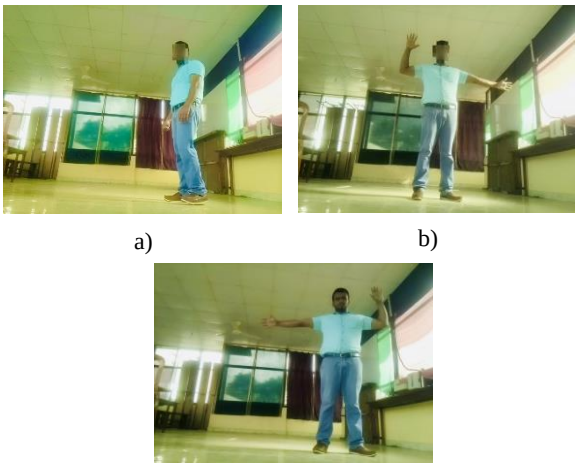


a) b)

c)

Fig 4: Sample images from Data Set 01 collected in indoor environments: (a) Person, (b) Person Stop and (c) Police Stop

Although the models were designed for actual self-driving cars, the implementation was carried out using a prototype car. Consequently, images from the perspective of the prototype car were included in the datasets. A Picamera was used to capture these images, as shown in Figure 5.



a) b)

c)

Fig 5: Sample images of the Data Set 01 captured using a Picamera

from two angles: one covering the upper body and the other covering the full body. Additionally, images were captured from three different distances: 2m, 4m, and 7m. A Python script

was developed to automate the continuous image capture process.

The data were collected in daylight conditions and inside a hall. Data Set 01 contains a total of 2,855 images, with 2,168 images distributed across three different classes. The number of images for each class in Data Set 01 is listed in Table 2.

Table 2: Number of images in each object class of Data Set 01

Class	Number of Images
Person	891
Person Stop	973
Police Stop	991

2.1.2 Data set 02

This data was collected as video recordings during real-time driving in road conditions, including scenarios such as pedestrians randomly crossing and congested roads controlled by traffic police officers. An Apple iPhone 5S camera was used to capture these videos over a period of five months, considering various climate and lighting conditions. Using FFmpeg [22], 3,383 images were extracted from the video files. These images cover five different classes: Person, Person Stop, Police Stop, Police, and Zebra Crossing, as shown in Figure 6. The number of images for each class in Data Set 02 is presented in Table 3.

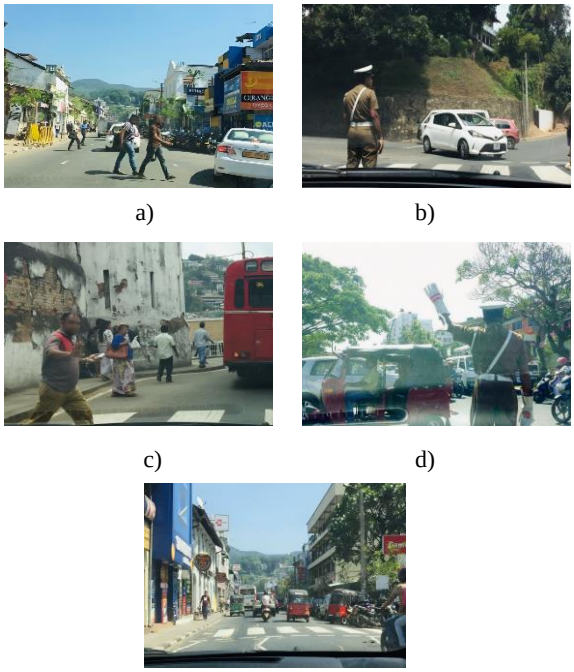


Fig 6: Sample images from real road conditions outdoor collected Data Set 02 (a) Person (b) Police (c) Person Stop, d) Police Stop and (e) Zebra crossing

Table 3: Number of examples in the outdoor collected data set

Class	Number of Images
Person	1567
Person Stop	359
Police Stop	258
Police	649
Zebra crossing	550

2.1.3 Data set 03

To incorporate international road conditions and analyze the impact of the number of images in each class on the model's results, images from the Joint Attention in Autonomous Driving (JAAD) dataset were selected [23]. The JAAD dataset, created for developing self-driving vehicle models, includes videos filmed in various locations across North America and Eastern Europe. For our study, only images from the Person class were considered, as shown in Figure 7. A total of 582 images were extracted for this class.

Fig 7: Sample images from Data Set 03 which were extracted from JAAD data set

2.2 The proposed model

2.2.1 Faster R-CNN

There are various multiclass object detection models, including Fast R-CNN, Faster R-CNN, YOLO, and Mask R-CNN. Among these, Faster R-CNN was identified as the best model for this task. Details of the Faster R-CNN model are presented below.

Faster R-CNN is one of the most widely used object detection models. Similar to other popular models, it relies on deep convolutional neural networks (CNNs). It has become an authoritative model for deep learning-based object detection and has influenced many subsequent detection and segmentation models. The main advancement of Faster R-CNN over its predecessor, R-CNN, is the replacement of the selective search algorithm with a faster neural network, the Region Proposal Network (RPN). In the final layer of an initial CNN, a sliding window of size 3 × 3 moves across the feature map

and reduces it to a lower dimension, as shown in Figure 8. For each location of the sliding window, the RPN generates multiple possible regions based on k fixed-ratio anchor boxes, also known as default bounding boxes. Each region proposal includes a score and four coordinates representing the bounding box of the region.

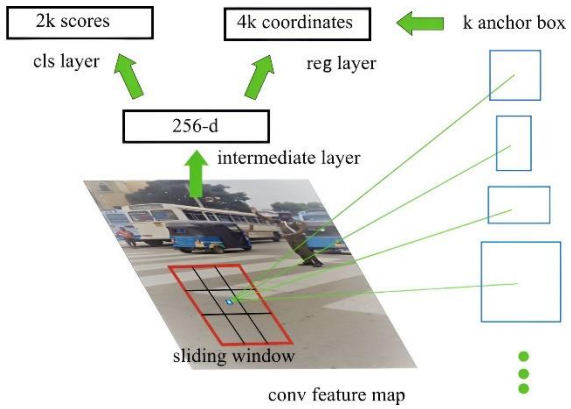


Fig 8: The anchor box approach of Faster R-CNN

The 2k scores represent the softmax probability of each of the k bounding boxes containing an object. Although the RPN outputs bounding box coordinates, it does not classify any potential objects; it only proposes object regions. When an anchor box has an objectness score exceeding a certain threshold, that box's coordinates are passed forward as a region proposal. The architecture includes fully connected layers, a pooling layer, a softmax classification layer, and a bounding box regressor. Faster R-CNN offers superior speed and high performance. The architecture of Faster R-CNN is illustrated in Figure 9. Several CNNs can serve as the backbone for Faster R-CNN, such as Inception, AlexNet, VGG-16, and LeNet. In this study, VGG-16 was used as the backbone for Faster R-CNN.

2.3 Model training

Data preprocessing and training of Faster R-CNN were performed as follows. Faster R-CNN requires only an input image and ground truth boxes for each object. The training dataset was generated manually by tagging the classes in the images using LabelImg [24], as shown in Figure 10. LabelImg saves the annotations as XML files in PASCAL VOC format.

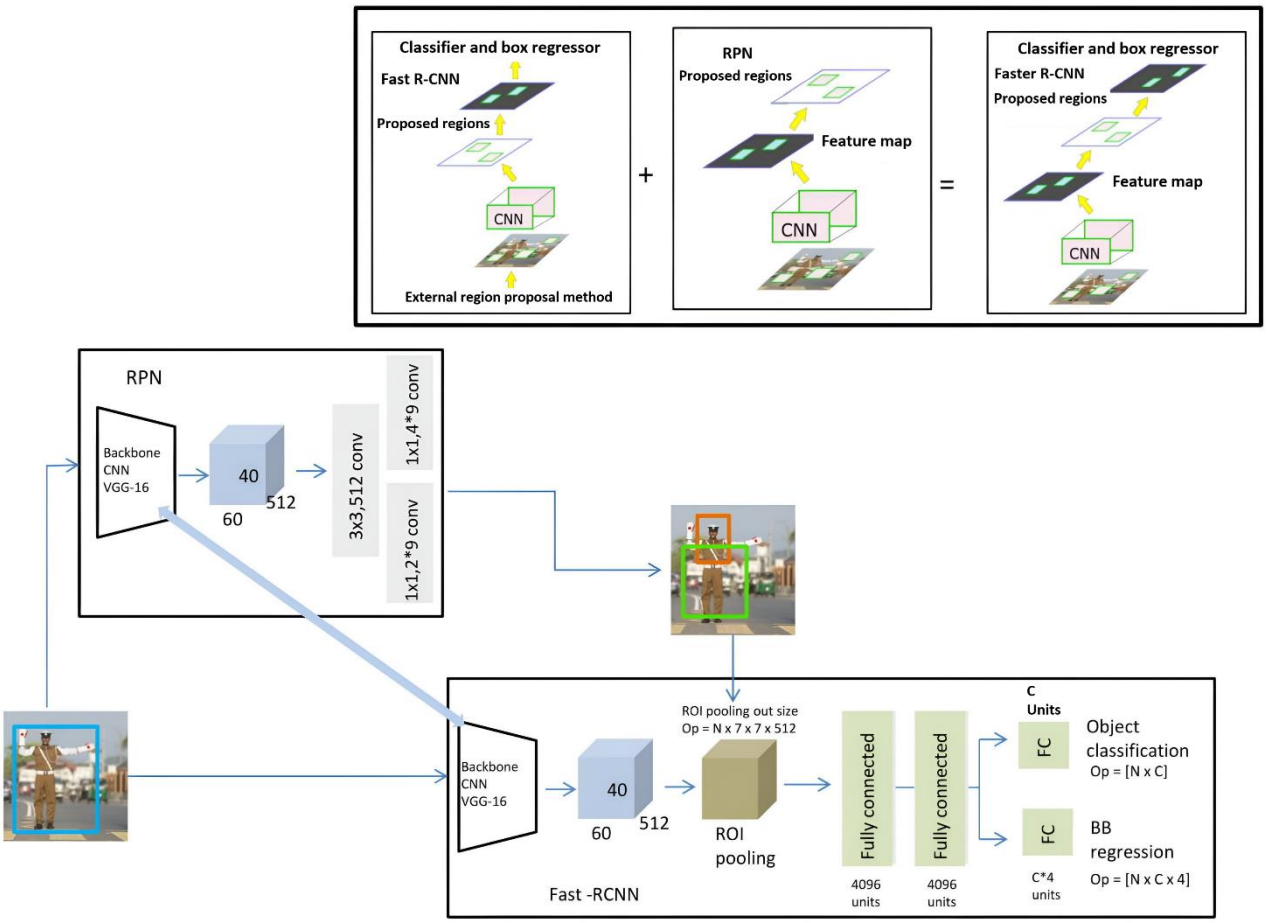


Fig 9: The complete Faster R-CNN architecture

From the selected images, 80% were allocated for training and 20% for testing. During the training process, the frozen graph weights are integrated into the graph as constants to be used for out-of-the-box inference. At the core of Faster R-CNN is the prediction of category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps. Once the training process is complete, the object detection classifier performs object detection, localization, and classification. The pre-trained Faster R-CNN models were fine-tuned for our collected datasets using manually labeled images. The training was halted after 200,000 timesteps.

Fig 10: Labelling of the classes (Passenger and zebra crossing) using bounding boxes.

2.4 Model evaluation

Precision, recall, F1-score, and mAP (Mean Average Precision) are the most relevant evaluation metrics for this application. The relevance of the detection results are described by precision (Eq. 1):

$$\text{precision} = \frac{TP}{TP + FP} \tag{1}$$

where TP = true positives, FP = false positives

Recall describes the percentage of relevant objects that are detected with the detector (Eq. 2):

$$\text{recall} = \frac{TP}{TP + FN} \tag{2}$$

where FN = false negatives.

In general, as precision increases, recall decreases, and vice versa. A highly sensitive model may detect a large percentage of objects in an image but also generate a high number of false positives. Conversely, a model with a high detection threshold may produce fewer false positives but also leave a higher percentage of objects undetected. The application must strike the right balance between these two factors. The F1-score serves as a single metric for evaluating these two viewpoints (Eq. 3).

$$\text{F1\_score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{3}$$

The definition of true positives, false positives, and false negatives can vary between different object detection applications. The TensorFlow Object Detection API adopts the "PASCAL VOC 2007 metrics" [25], where a predicted instance is considered a true positive when the Intersection over Union (IoU) exceeds 50% [26] (Eq. 4).

$$\text{IoU} = \frac{area(\text{true bbox} \cap \text{predicted bbox})}{area(\text{true bbox} \cup \text{predicted bbox})} > 0.5 \tag{4}$$

An object can only have one bounding box associated with it. Therefore, if multiple bounding boxes are predicted for an object, only one is considered a true positive (TP), while the others are considered false positives (FP). An object without any predicted bounding boxes associated with it is classified as a false negative (FN). Objects labeled as "difficult" are excluded from the evaluation protocol.

Mean Average Precision (mAP) serves as a widely used metric for comparing model performance in object detection. For each prediction, both recall and precision values are calculated based on their confidence ratings. Recall is the proportion of true positives (TPs) above a given rank out of all user-tagged objects, while precision is the proportion of TPs in predictions above the given rank. A precision-recall curve is formed from these precision-recall pairs, and Average Precision (AP) is calculated as the area under this curve. AP approximates precision averaged across all values of recall between 0 and 1 by summing precision values multiplied by the corresponding change in recall from one point in the curve to the next. However, what VOC metrics refer to as "AP" is the interpolated average precision, defined as "the mean precision at a set of eleven equally spaced recall levels". The precision value corresponding to each recall interval is the maximum precision value observed in the curve within the interval. The mAP is then the mean of AP values for all object classes. It is important to note that the basic AP is typically lower than the interpolated AP, and this distinction should be considered when comparing mAP values.

2.5 Model implementation

The models were trained using the TensorFlow object detection API, which performs optimally with TensorFlow-GPU. This API is an open-source framework developed on top of TensorFlow, designed to simplify the construction, training, and deployment of object detection models. It offers users access to various pre-trained object detection models along with instructions and example codes for fine-tuning and deploying these models for object detection tasks. The TensorFlow object detection API supports the use of different pre-trained models and allows for fine-tuning according to specific datasets. In this study, the Faster R-CNN and SSD models with MobileNet were selected. Training was conducted on a Dell PowerEdge R740xd server equipped with a P40 GPU. The purpose of testing with two models was to identify a more suitable solution for devices with limited processing power, such as the Raspberry Pi.

2.6 Prototype car

A prototype car was developed to validate the proposed solution, employing a Raspberry Pi 3 Model B as the microcontroller. The Raspberry Pi processed the live video feed from the Picamera, while the object detection model predicted the classes. Control of the front and rear motors of the car was facilitated by the L298N motor controller. The complete circuit diagram of the prototype car is depicted in Figure 11. The Raspberry Pi is renowned for its robust performance and is widely utilized in low-power processing projects. Despite the availability of other devices with greater processing power and performance, the Raspberry Pi was chosen for its cost-effectiveness and suitability for the task at hand.

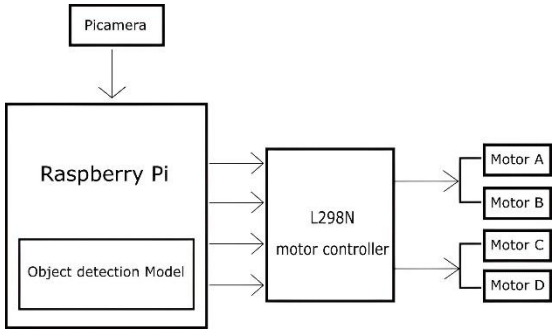


Fig 11: Circuit diagram of the prototype car

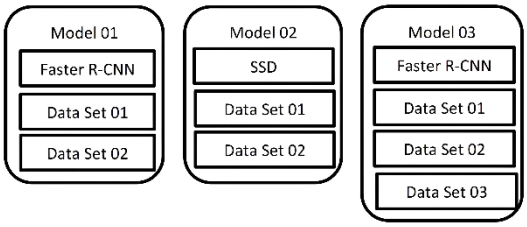
3. RESULTS AND DISCUSSION

3.1 Experiments

Three experiments were conducted using Faster R-CNN, along with the single-shot detector (SSD) for comparison purposes. SSD is another multi-class object detection model that predicts both the bounding box and the class simultaneously as it processes the image in a single shot.

In SSD, an input image and a set of ground truth labels are fed into the model, which then passes the image through a sequence of convolutional layers, cropping several sets of feature maps at different scales. VGG-16 serves as the backbone of SSD.

The object detection models were trained using different datasets. Faster R-CNN and SSD with Data Set 01 and Data Set 02 were designated as Model 01 and Model 02, respectively. To assess the impact of the number of images in each class on the final performance of the model, Faster R-CNN was trained with Data Set 01, Data Set 02, and Data Set 03. This approach was labeled as Model 03. The models and selected datasets are illustrated in Figure 12.



a) **b)** **c)**

Fig 12: Models with their trained data sets, Model 01 (a), Model 02 (b), and Model 03 (c)

3.2 Results of model training

All models completed training for 200,000 epochs. The total loss graphs for Model 01, 02, and 03 are depicted below in Figure 13. Notably, Model 02 failed to predict any classes after 200,000 epochs of training. As a result, Model 02 was subjected to retraining for an additional 100,000 epochs. Examination of the loss graphs in Figure 13 indicates that all models have been adequately trained.

3.3 Results of model testing/verification

A specially curated dataset was employed to assess the models' performance. This dataset consists of 238 images, each encompassing all five classes. Importantly, these images were not utilized during the model training phase. The trained models were then tested using this dataset of 238 images.

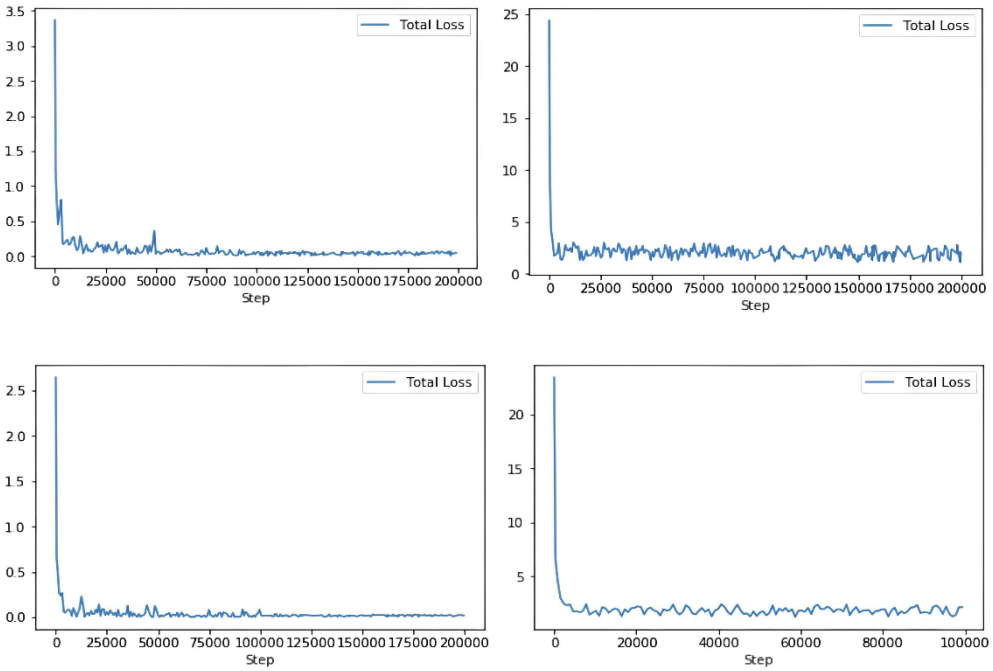


Fig 13: Total loss graphs of Model 02 (a), Model 03 (b) and Model 04 (c) with 200,000 time steps, and Model 03 with 100,000 time steps (d)

The performance of Model 01 is presented in Table 4, showcasing Precision, Recall, and F1-score values. As observed in Table 4, the precision and recall values for Person Stop class are lower compared to all the other classes. Specifically, Faster R-CNN falsely identifies a person as a Person Stop sign, even when the person merely raises one hand more than 45 degrees. Hence, the accuracy of predicting the Person Stop sign is lower compared to the other classes.

The performance of Model 02 is outlined in Table 5. A notable decrease in precision value is observed for the Police class. This decline in precision can be attributed to a higher number of false-positive predictions. Specifically, certain images belonging to the Person class were erroneously identified as Police, possibly due to the similarity in color between the clothing worn by individuals and police uniforms. Despite this, Table 5 indicates a high recall value for the Police class.

Table 6 illustrates the performance of Model 03, which comprises Faster R-CNN trained with all three datasets. The objective of training this model was to assess the influence of the number of images in a single class on the final predictions. Notably, the number of images in the Person class was increased for this purpose.

As depicted in Table 6, the precision, recall, and F1-score for the Person class have increased compared to Model 01. However, the average values for Model 03 are lower than those of Model 01. Consequently, while the performance in predicting the Person class has improved, the performance in predicting other classes has declined.

3.4 Sample predictions of the best model (Model 01-Faster R-CNN)

Some prediction results of the best model (Model 1) are shown in Figure 14. These sample predictions include both indoor and outdoor images.

3.5 Performance of the models with prototype car

Figure 15 displays the prototype car, which was created as a cost-effective solution since designing a full-scale vehicle would be impractical. To determine the optimal model for the prototype car, each model underwent testing with the Raspberry Pi. While Model 01 exhibited superior performance, its prediction results were excessively slow. Consequently, Model 01 proved unsuitable for the prototype car, which relies on live video feeds. Conversely, Model 02 demonstrated faster processing with the Raspberry Pi while maintaining commendable prediction accuracy. Hence, Model 02 was chosen for deployment with the prototype car. Notably, the performance of the prototype car was sluggish with Faster R-CNN but improved significantly with SSD. Consequently, the final implementation utilized SSD for enhanced performance.

4. Discussion

As outlined in the results section, Faster R-CNN emerged as the top-performing model compared to SSD. Notably, within both Model 01 and Model 02, Faster R-CNN exhibited greater sensitivity than SSD. This heightened performance can largely be attributed to Faster R-CNN's utilization of region proposal networks.

However, Faster R-CNN incurred higher computational costs compared to SSD. Both Faster R-CNN and SSD models underwent training for 200,000 epochs initially. However, SSD did not perform as expected in this instance due to overtraining, leading to suboptimal results. This underscores the potential ramifications of overtraining on model efficacy. Subsequently, SSD was retrained for 100,000 epochs, resulting in improved outcomes. Hence, 100,000 timesteps emerged as the optimal training duration for SSD.

Table 4: Precision and recall of the Model 01 detections

	Person	Person Stop	Police	Police Stop	Zebra crossing	Average
Precision	0.92	0.67	0.90	0.93	0.94	0.87
Recall	0.93	0.75	0.78	0.82	0.92	0.84
F1-score	0.92	0.85	0.80	0.87	0.93	0.87

Table 5: Precision and recall of the Model 02 detections

	Person	Person Stop	Police	Police Stop	Zebra crossing	Average
Precision	0.89	0.81	0.48	0.88	0.95	0.80
Recall	0.86	0.80	0.74	0.82	0.83	0.81
F1-score	0.87	0.80	0.58	0.84	0.88	0.79

Table 6: Precision and recall of the Model 03 detections

	Person	Person Stop	Police	Police Stop	Zebra crossing	Average
Precision	0.93	0.57	0.66	0.91	0.93	0.80
Recall	0.97	0.65	0.78	0.95	0.89	0.84
F1-score	0.94	0.60	0.71	0.92	0.90	0.81

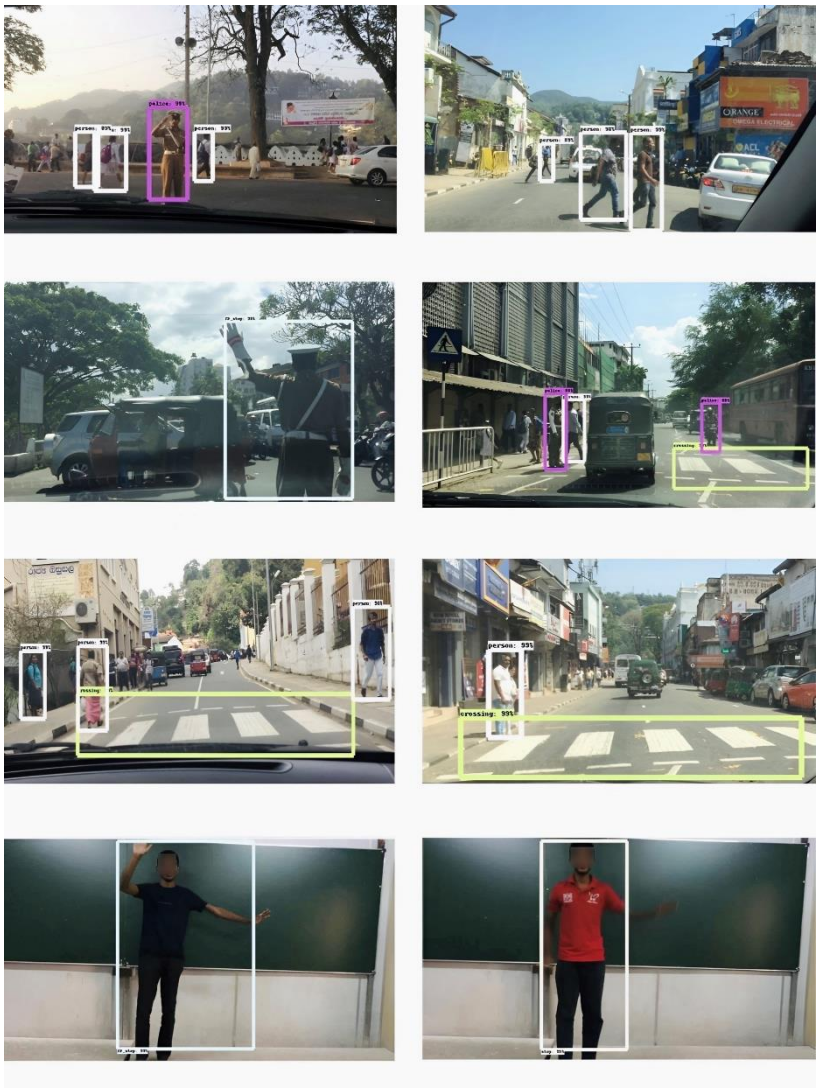


Fig 14: Sample predictions of the best model (Model 01) with bounding boxes.

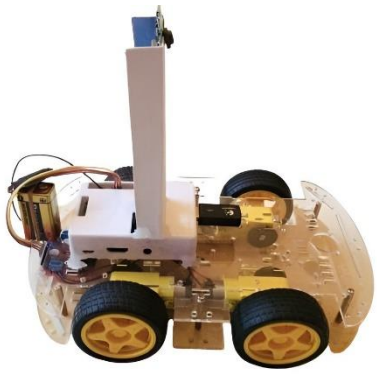


Fig 15: The final design of the prototype car

The number of images in each class within a dataset can be adjusted. Given that data collection during real-time driving conditions naturally yields more images in the Person class than in other classes, an investigation into the impact of image quantity on model performance was conducted by augmenting the number of Person class images. Training the datasets with Faster R-CNN revealed that while the accuracy of the Person class improved, the accuracy of other classes slightly decreased compared to the initial model results. This suggests that augmenting the number of examples in a single class can enhance the model's ability to detect that class. Consequently, the number of examples in a dataset emerges as a critical factor influencing model performance, with higher volumes of data resulting in improved performance.

There is a notable imbalance in the results across various classes. Particularly, the accuracy of the Person Stop sign and Police classes significantly lagged behind the others. The primary issue with the Person Stop class stemmed from models incorrectly identifying individuals as Person Stop signs when their hands were raised more than 45 degrees, even though this gesture did not signify a stop signal. Additionally, instances arose where models misclassified individuals as Police when their clothing colors resembled those of a police uniform. Moreover, in certain images, the models struggled to distinguish between the Police and Person classes.

The quality and quantity of images in the dataset significantly influence the model's sensitivity and accuracy. It is advisable

to include high-resolution images with sharp details in the dataset. In this study, images ranging from 640x480 to 1920x1080 resolution were utilized, with file sizes below 521 Kilobytes. Larger image sizes may prolong the model training process. Training the models on a GPU-accelerated platform is recommended to expedite the training process.

When employing both models with Raspberry Pi, Faster R-CNN exhibited slower performance, achieving as low as 1.05 frames per second. Consequently, utilizing Faster R-CNN with a low-power device can result in diminished speed performance. As SSD operates more swiftly than Faster R-CNN with Raspberry Pi, SSD was employed for the prototype car. However, in terms of performance, Faster R-CNN outperformed SSD. Therefore, the prototype car's real-time driving implementation was executed using SSD, prioritizing speed over performance.

5. CONCLUSIONS

When implementing a self-driving car's navigation system, various factors must be taken into account, especially in diverse traffic scenarios where pedestrians may cross roads unpredictably or traffic officers control traffic flow. To address this challenge, multiclass object detection models were developed to identify pedestrians, traffic officers, and their hand signals. For instance, when a person is detected on a Zebra crossing, the self-driving car must halt, whether or not a hand signal is present. Consequently, images were gathered for training these object detection models across five classes: Person, Person stop, Police stop, Police, and Zebra crossing. To encompass a wide range of scenarios, data were collected indoors, outdoors, and from the Joint Attention in Autonomous Driving (JAAD) dataset. Subsequently, each model was subjected to training using various datasets, while the impact of epoch numbers on model performance was also assessed. Additionally, the effect of increasing the number of images per class on the final model predictions was evaluated.

After reviewing all the test results and performance metrics, it was determined that Faster R-CNN emerged as the most accurate model among the three. Furthermore, this model exhibited superior speed compared to the others and excelled in object localization and detection. While Faster R-CNN delivers exceptional performance in these aspects, its efficiency diminishes when deployed on low-power computational devices. However, optimizing the resources allocated to model execution can enhance its speed. Notably, Faster R-CNN requires more training time compared to the SSD model, capable of training up to 200,000 epochs without encountering issues. Conversely, the SSD model's performance deteriorates after surpassing 100,000 epochs. Consequently, selecting the appropriate model and training parameters is crucial to achieving optimal performance. Finally, a Raspberry Pi-powered prototype car was developed for model deployment and real-time driving tests. However, implementing these models demands considerable computational resources beyond the capabilities of the Raspberry Pi. Therefore, utilizing a higher-powered computational platform is essential for seamless implementation. Additionally, the Picamera, renowned for its superior image quality and lightweight design, is recommended for projects requiring a compact camera solution, particularly when integrated with devices like the Raspberry Pi.

To enhance performance, it is imperative to gather additional data using high-quality cameras, encompassing various lighting conditions and climates, including day and night scenarios. While the Raspberry Pi employed in the prototype car exhibits sluggish performance in real-world settings, transitioning to more powerful processing devices is essential to achieve optimal performance. Expanding the dataset allows for utilizing resources like Nvidia Cloud GPU or Google Cloud GPU for efficient model training. Additionally, robust networking capabilities enable the utilization of cloud or edge computing for vehicle operation.

6. REFERENCES

[1] T. Winkle, “Safety benefits of automated vehicles: Extended findings from accident research for development, validation and testing,” in Autonomous Driving: Technical, Legal and Social Aspects, 2016, pp. 335–364. doi: 10.1007/978-3-662-48847-8_17.

[2] T. Litman, “Autonomous Vehicle Implementation Predictions: Implications for Transport Planning,” Transportation Research Board Annual Meeting, vol. 28, 2014, doi: 10.1613/jair.301.

[3] A. Rasouli, I. Kotseruba, and J. K. Tsotsos,“Understandi ng Pedestrian Behavior in Complex Traffic Scenes,” IEEE Transactions on Intelligent Vehicles, vol. 3, no. 1, pp. 61–70, 2018, doi: 10.1109/tiv.2017.2788193.

[4] S. Deb, L. Strawderman, D. W. Carruth, J. DuBien, B. Smith, and T. M. Garrison, “Development and validation of a questionnaire to assess pedestrian receptivity toward fully autonomous vehicles,” Transportation Research Part C: Emerging Technologies, vol. 84, pp. 178–195, 2017, doi: 10.1016/j.trc.2017.08.029.

[5] R. Sun, X. Zhuang, C. Wu, G. Zhao, and K. Zhang, “The estimation of vehicle speed and stopping distance by pedestrians crossing streets in a naturalistic traffic environment,” Transportation Research Part F: Traffic Psychology and Behaviour, vol. 30, pp. 97–106, 2015, doi: 10.1016/j.trf.2015.02.002.

[6] A. Hussein, F. García, J. M. Armingol, and C. Olaverri-Monreal, “P2V and V2P communication for pedestrian warning on the basis of autonomous vehicles,” in IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, 2016, pp. 2034–2039. doi: 10.1109/ITSC.2016.7795885.

[7] A. Rasouli, I. Kotseruba, and J. K. Tsotsos, “Agreeing to cross: How drivers and pedestrians communicate,” in IEEE Intelligent Vehicles Symposium, Proceedings, 2017, pp. 264–269. doi: 10.1109/IVS.2017.7995730.

[8] E. Y. Du et al., “Pedestrian behavior analysis using 110-Car Naturalistic Driving data in USA,” 2013. [Online]. Available: https://www.esv.nhtsa.dot.gov/Proceedings/23/isv7/main.htm

[9] Z. Ren, X. Jiang, and W. Wang, “Analysis of the Influence of Pedestrians’ eye Contact on Drivers’ Comfort Boundary during the Crossing Conflict,” in Procedia Engineering, 2016, vol. 137, pp. 399–406. doi: 10.1016/j.proeng.2016.01.274.

[10] A. Tom and M. A. Granié, “Gender differences in pedestrian rule compliance and visual search at signalized and unsignalized crossroads,” Accident Analysis and Prevention, vol. 43, no. 5, pp. 1794–1801, 2011, doi: 10.1016/j.aap.2011.04.012.

[11] M. Sucha, D. Dostal, and R. Risser, “Pedestrian-driver communication and decision strategies at marked

crossings," Accident Analysis and Prevention, vol. 102, pp. 41–50, 2017, doi: 10.1016/j.aap.2017.02.018.

[12] D. Dey and J. Terken, "Pedestrian interaction with vehicles: Roles of explicit and implicit communication," in AutomotiveUI 2017 - 9th International ACM Conference on Automotive User Interfaces and Interactive Vehicular Applications, Proceedings, 2017, pp. 109–113. doi: 10.1145/3122986.3123009.

[13] R. Sathya and M. Kalaiselvi Geetha, "Vision based Traffic Police Hand Signal Recognition in Surveillance Video - A Survey," International Journal of Computer Applications, vol. 81, no. 9, pp. 1–10, 2013, doi:10.5120 / 14037-2192.

[14] G. Wang and X. Ma, "Traffic Police Gesture Recognition using RGB-D and Faster R-CNN," in 2018 International Conference on Intelligent Informatics and Biomedical Sciences, ICIIBMS 2018, 2018, pp. 78–81. doi: 10.1109/ICIIBMS.2018.8549975.

[15] C. Wang, C. Zhao, and H. Wang, "Self-similarity based zebra-crossing detection for intelligent vehicle," Open Automation and Control Systems Journal, vol. 7, no. 1, pp. 974–986, 2015, doi:10.2174/1874444301507010974.

[16] X. Liu, Y. Zhang, and Q. Li, "Automatic pedestrian crossing detection and impairment analysis based on mobile mapping system," in ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2017, vol. 4, no. 2W4, pp. 251–258. doi: 10.5194/isprs-annals-IV-2-W4-251-2017.

[17] K. Behrendt, L. Novak, and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," in Proceedings - IEEE International Conference on Robotics and Automation, 2017, pp. 1370–1377. doi: 10.1109/ICRA.2017.7989163.

[18] R. W. Wolcott and R. M. Eustice, "Visual localization within LIDAR maps for automated urban driving," in IEEE International Conference on Intelligent Robots and Systems, 2014, pp. 176–183. doi:10.1109/IROS.2014.69 42558.

[19] Q. Rao and J. Frtunikj, "Deep learning for self-driving cars: Chances and challenges: Extended Abstract," in Proceedings - International Conference on Software Engineering, 2018, pp. 35–38. doi: 10.1145/ 3194085319 4087.

[20] J. Kocić, N. Jovičić, and V. Drndarević, "An end-to-end deep neural network for autonomous driving designed for embedded automotive platforms," Sensors (Switzerland), vol. 19, no. 9, 2019, doi:10.3390/s190920 64.

[21] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," Journal of Field Robotics, 2019, doi: 10.1002/rob.21918.

[22] Tomar and Suramya, "Converting video formats with FFmpeg," Linux Journal, 2006.

[23] Z. Fang and A. M. López, "Is the Pedestrian going to Cross? Answering by 2D Pose Estimation," in IEEE Intelligent Vehicles Symposium, Proceedings, 2018, vol. 2018-June, pp. 1271–1276. doi:10.1109/IVS.2018.85004 13.

[24] Tzutalin, "LabelImg," LabelImg, 2015.

[25] A. Lindgren, F. Chen, P. W. Jordan, and H. Zhang, "Requirements for the design of advanced driver assistance systems - The differences between Swedish and Chinese drivers," International Journal of Design, vol. 2, no. 2, pp. 41–54, 2008.

[26] G. M. Björklund and L. Åberg, "Driver behaviour in intersections: Formal and informal traffic rules," Transportation Research Part F: Traffic Psychology and Behaviour, vol. 8, no. 3, pp. 239–253, 2005, doi: 10.1016/j.trf.2005.04.006.