Vehicle Classifcation and Speed Estimation system using Raspberry pi 3

Shavin Tharinda Kaluthantri
Mechatronics Engineering
Monash University Malaysia campus
Bandar Sunway, Malaysia
stkal1@student.monash.edu

Madhavan Shanmugavel
Mechatronics Engineering
Monash University Malaysia campus
Bandar Sunway, Malaysia
madhavan.shanmugavel@monash.edu

Raghav Menon
Electrical and Electronic Engineering
Stellenbosch University
Stellenbosch, South Africa
rmenon@sun.ac.za

***Abstract*—This paper focusses on the design and development of a low-cost vehicle classification system using raspberry pi along with a speed estimation unit. This system, which is called augmented sensor-based classification system, uses a k-nearest neighbor (k-NN) method for the classification of vehicle classes based on vehicle dimensions. The vehicle class is strictly regulated by the law and is based on its dimensions. Hence with the aid of a camera and a few low-cost ultrasonic sensors we determine the speed and the dimensions of the vehicle. Experiments were conducted in laboratory with scaled models of different vehicle classes. The augmented sensor-based classification system has shown to classify vehicles with an accuracy of 0.842 with the speed estimation unit having an accuracy of 0.861. The system was tested for various camera angles and lighting conditions. We compared the performance of this system with a vision-based system using convolution neural network (CNN) trained directly on the images of vehicle belonging to different classes. The vision-based system had an accuracy of 0.858 which is highly dependent on the amount of training data, camera angle and lighting conditions. We see that performance of our low-cost system is comparable to that of the vision-based system.**

Keywords—vehicle classification, speed estimation, perspective transformation, convolutional neural networks k-nearest neighbors

I. INTRODUCTION

Vehicle identification and classification is a widely used technique by transportation systems used to classify vehicles to different classes based on the outer dimension or by counting number of axels in the vehicle. Vehicle classes is an important parameter in intelligent transportation systems as it is used for access control, obtaining statistical information, vehicle tracking for security purposes and automatic payment receipt creation at toll-collection points [2].

Speed estimation techniques can be used alongside vehicle classification and identification techniques to detect any problems in traffic automatically, such as speed rule violation and traffic congestion [3]. From estimates of speed, signs of accidents and traffic jams can be detected by low speeds of vehicle, followed by a high volume of detected vehicles. Therefore, adding speed information to vehicle classification and identification contributes greatly to the development of an intelligent transportation system.

From the wide range of the vehicle classification and speed estimation approaches [1]–[15] proposed, the techniques can be categorized in two main types:

1. Hardware-based approaches
2. Software-based approaches

Hardware based classification and speed estimation systems [2], [3], [7], [8]-[12] are the primitive and the most commonly used techniques used by transportation systems. Well known hardware-based systems include loop detectors, infrared, ultrasonic and radar sensors [2], the use of cheap inductive loops for vehicle classification was proven to be possible in [7] as the vehicles detected by loops for different classes of vehicles was found to have unique magnetic profile which was used as the basis for classification data processing algorithms. Different approaches aimed at enhancing single - loop detector speed estimation for length - based classification were investigated [8], [9]. However, classification errors occurred due to fact that these vehicles have similar vehicle lengths and axle lengths [8]. The inductive sensors were fused with piezo-sensors to obtain the vehicle weight and vehicle length using speed estimation [9]. However, these systems require sensors and controllers to be embedded in roadway surfaces [3], [7], [8], [9], [10], which require roads to be closed while the construction and maintenance is being done. Since sensors embedded on the roads pose a threat of damaging roads and obtaining false readings due to vehicles adjacent roads, A system has been proposed [12] which avoids these problems by the development of a portable roadside magnetic sensor. This device is placed next to the roadway and it measures the traffic in the immediate adjacent lane. The errors prone due to large vehicles in the opposite lane is removed by applying a threshold on vehicles classified based the magnetic intensity of the vehicles. This algorithm improves the overall accuracy by 8%. Speed measurement is obtained based on the calculations of the cross correlation between longitudinally spaced sensors and vehicle classification is done based on the magnetic length and an estimate of the average vertical magnetic height of the vehicle. This system can obtain speed measurements with an error of less than 2.5 %

The improvement in computational power of computers over the years lead to the use of novel computational intelligence-based techniques such as neural networks and genetic algorithms [1], [3], [6], [10], [13]. for the classification of vehicle patterns. Networks can learn and adapt non-linear patterns while the optimization of complex algorithm are done using genetic algorithms [10]. System proposed in [3] uses back Propagation neural network (BPNN) to classify time variable signals of vehicles generated by a single inductive sensor into 5 group, achieving a classification rate of 94.21%.

Software-based vehicle classification and speed estimation mainly uses computer vision to automatically process images of the field of view of the camera [6]. Advanced video monitoring systems and research techniques [4]-[6], [13]-[15] are being developed to enable monitoring and analysis of traffic in real time. The proposed algorithm in [14] classifies vehicles using size and linearity features of vehicles obtained from the video sequences of one camera, the size classification technique is used to differentiate different types of vehicles while the linearity features recognizes the difference between trucks and busses which are similar in sizes. This system has recognition rate of 82% with shadow elimination and 69% without elimination [14].

Techniques to obtain distance and speed measurements from front facing cameras have been investigated in [18] and [20]. The system proposed in [18] uses Inverse Perspective Mapping (IPM) to obtain a bird's eye view of camera view to obtain a linear relationship of distance in the image and in the real world. The system developed in [20] detects moving objects by utilizing frame differencing techniques, Next object tracking method is applied and the speed of the vehicle is measured based on the displacement of the object's centroid. the proposed algorithm has achieved the velocity accuracy estimation of about ± 1.7 km/h.

Vehicle classification and speed estimation using image processing is a much cheaper and accurate option since most roads, junctions and highways have cameras placed on the side of the road and at exit toll booths for security purposes, the maintenance and installation costs of video-based systems are very low, and they can be adjusted to provide better view. Furthermore, a variety of algorithms can be implemented to obtain additional data such traffic density and collision detection [1]. However, the algorithms used to classify vehicles should be real-time and robust with the ability to work well with different weather and lighting conditions.

As a result, two Real time Vehicle Classification algorithms and one Speed Estimation algorithms was created to be implemented on a low-cost, computationally powerful microcontroller: Raspberry pi 3.

The first system implemented was a vision-based system which uses a highly efficient Convolutional Neural Network Architecture: LeNet [16] to train a model to classify the vehicles based on various feature patterns learned by the model. This architecture allows the trained model to be implemented on the Raspberry pi 3 without the requirement of any expensive Graphics processing units (GPUs).

The second algorithm consists of a vision-based system augmented with an Ultrasonic sensor-based system. The vision-based system was used to calculate the speed of a moving vehicle, this speed was used by the ultrasonic sensors to obtain the length and maximum height of the vehicle. The classification of vehicles was then carried out based these parameters. The deficiencies in the vision-based system was supplemented by ultrasonic sensors and since ultrasonic sensors will be placed over head, they will not cause any damages to roads.

The developed algorithms were implemented on a protype of traffic scenario to test the performance of the algorithm under varying lighting conditions and camera angles. The protype allowed easy acquisition of data from the model vehicles created.

II. Experimental Setup

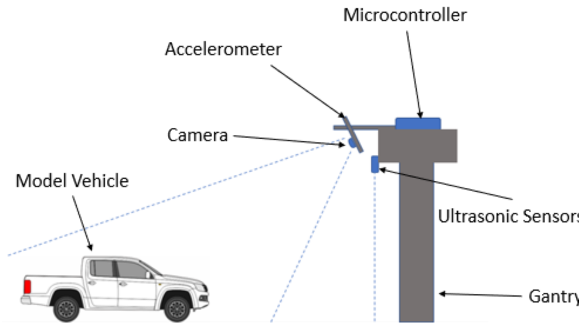

Figure 1: Design of the protype

Number of wheels	Height at front axle	Number of axles	Class	Example
4 or more	Under 1.3 metres	2	2	
		More than 2	3	
	1.3 metres or more	2	4	
		3 – 6	5	

Figure 2: Vehicle classification chart

The prototype consisted of a gantry, Raspberry pi 3 microcontroller, Raspberry pi camera and two ultrasonic sensors. The gantry was built by connecting 7 steel frames as shown in Figure 1. The Raspberry pi camera was mounted to the frame work at a height of 0.4 m above the ground along with an accelerometer to measure angle of inclination of the camera. Two ultrasonic sensors operating at frequency of 40Hz were placed on the steel gantry at height of 0.35m above the ground with a horizontal distance of 0.08m between them. Two ultrasonic sensors were used instead of one sensor to increase the range of vehicle detection.

Four model vehicles were created with each vehicle belonging to a class shown in the classification chart of Figure 2. The prototype vehicles were scaled down to obtain to actual to real dimension ratio of 1:20. The dimensions of the vehicles were ensured to be within the Maximum allowable dimensions for ground vehicles as stated in the Mass, Loading and Access (MLA) Regulations in NSW.

III. Methodology

A. Vehicle Detection

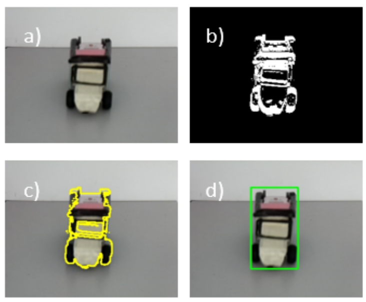

Figure 4: Steps taken for object detection

To detect a moving vehicle in the video frames, the current video frame shown in Figure 3 a) is converted to a grayscale image and is blurred (smoothened) using a Gaussian filter to reduce the noise. Next, the difference between the current frame and the weighted average of the previous frame is obtained and is then converted to black and white image based on a threshold value so that the difference between two frames appears in white while the rest of image which is common to both the frames appear in black as shown in Figure 3 b). The white areas of the image are then enlarged to create a connected region of the detected object, these connected regions are connected by curve joining all the continuous points (along the boundary), having same color or intensity, known as contours as shown in Figure 3 c). The areas of the contours are obtained and moving objects are identified by setting a threshold on the area of contours and choosing contours within a specified area, so that other moving objects in the background are ignored. A rectangle is drawn around the contour and is displayed on the original video to represent detected moving vehicles as shown in Figure 3 d).

B. Vision Based System

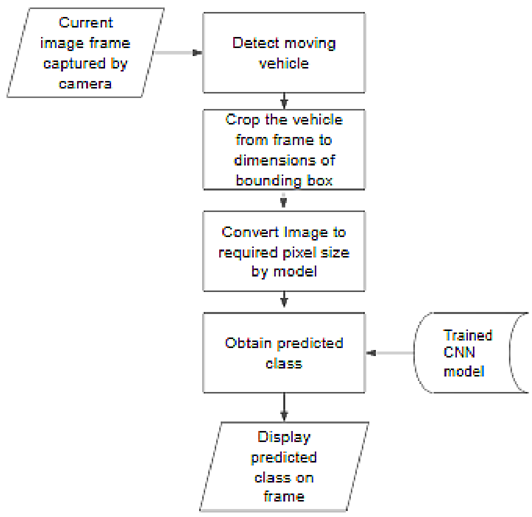

Figure 5: Flow chart of vision-based system

The vision-based system was used to classify vehicles to their respective classes by feeding the of detected vehicle cropped from original frame to a trained CNN which provides an estimate of the vehicle class based on the feature pattern learned by the model. The working algorithm is shown in Figure 4.

1) Architecture of CNN model

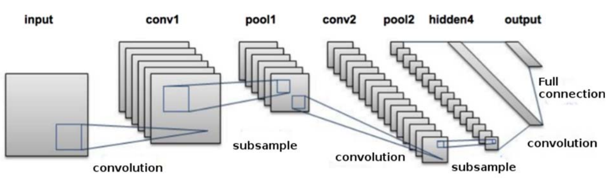

Figure 6: Architecture of Lenet

Due to restricted memory (1GB Ram) and limited processor speed of the Raspberry pi 3 microcontroller, training and testing large neural networks is not possible. Therefore, the LeNet architecture [16] was implemented to classify vehicles in real time due to its computational efficiency, and smaller processing footprint. The architecture of the LeNet is shown in Figure 5. And it consists of two sets of convolutional, activation, and pooling layers, followed by a fully-connected layer, activation layer, another fully-connected layer, and a SoftMax classifier

The CNN model was created and implemented using Keras [17] with TensorFlow backend. Keras is a high-level neural network application processing Interface written in Python and TensorFlow is an open-sourced symbolic math library used in machine learning applications.

2) Creating Datasets

A large image dataset is required to train the CNN to recognize visual patterns that allow it to classify images into various classes. To create a large dataset, the vehicle detection algorithm used on Raspberry Pi 3 was implemented to capture each frame of the video stream that has detected a moving model vehicle. For each frame for which moving vehicle was detected, the image of the vehicle cropped to the size of the bounding box that surround the contour of the detected vehicle was saved. Vehicle models belonging to each class was made to run below the gantry several times until enough images were collected and the images obtained for each vehicle class was stored in separate folders to be used as training data for the CNN. The algorithm was run at different lighting conditions and at different camera angles to increase the size and accuracy of the training dataset.

The pictures were pruned to remove irrelevant images and several datasets from these images were created to evaluate the performance of the model. The initial dataset consisted of 4 folders containing the cropped images of each model vehicle categorized based on the class name.

3) Training CNN model

From the dataset, the images are reshaped to dimension of 28x28x3 pixels which is spatial dimensions required by the LeNet architecture and is rescaled by converting the RGB range of the images [0, 255] to the range [0,1]. This data is stored in data and label lists which stores the image data and the corresponding class label of each image. Next, the test data is split with 75% of the images to be used as images for training and 25% as testing data. Additional image augmentation is performed using image generator which performs random rotations, shifts, flips, crops, and sheers on our image dataset to obtain accurate results with a small dataset.

The filter values or weighs at each neuron in the convolutional and dense layers are adjusted through training process known as back propagation which consists of 4 processes: forward pass, loss function, backward pass and weight update.

4) Testing the CNN model

Once the model was trained they were tested on a dataset consisting of vehicles of sample cropped images of the vehicle models. The images were not included in the dataset used to train the model and therefore, this test dataset was used to evaluate the performance of the CNN model.

The model with the highest accuracy was implemented on the Raspberry pi 3. This vision-based system used the vehicle detection algorithm to crop the image of detected vehicle from the frame, this cropped image was then fed to the CNN model perform vehicle classification in real time.

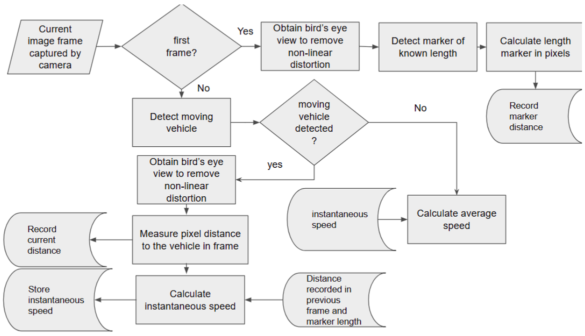

Figure 8: Flow chart of speed estimation algorithm

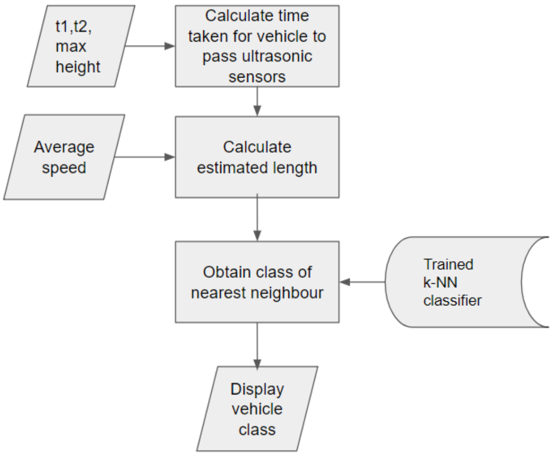

Figure 9: Flow chart of Augmented System

C. *Augmented System*

The augmented system consists of a camera and pair of ultrasonic sensors and the working speed estimation algorithm of the system is shown in **Error! Reference source not found.** and the classification algorithm is shown in Figure 7.

1) *Obtaining Bird's Eye view*

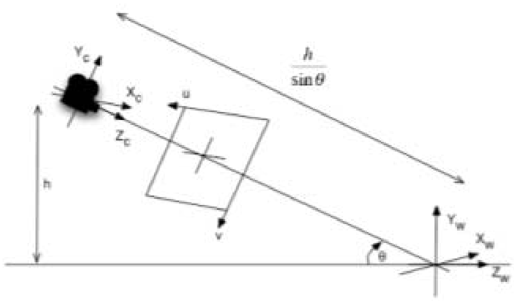

Figure 10: Pinhole Camera model

Planar homography relates the transformation between two planes in scale form and this concept can be used to transform the current camera pose to the required pose of Bird's eye view. To obtain the homography matrix the rotational and translation matrices of current camera pose, and the required camera pose is needed to be obtained.

The rotational matrix of current camera pose is equivalent to the rotational transformation of the camera angles obtained by the accelerometer. The translational vector of current camera pose is estimated by calibrating camera with a known pattern using OpenCV calibration libraries. Next, the required camera poses to obtain a bird's eye view is found be equivalent to a rotation of the world coordinates by 90º along *x*-axis based on the coordinate frame shown in Figure 8 . The translation vector relating required camera pose is equivalent to the current camera pose since the transformation required to obtain a bird's eye view is purely rotational. Transforming the image to a top-down view removes the non-linearity of distance and therefore, accurate measurements of the distance and speed of the vehicle can be calculated.

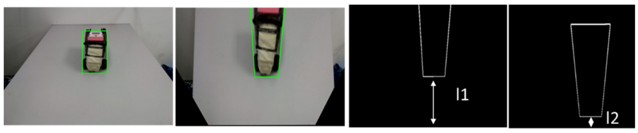

Figure 7: Steps taken for speed estimation

2) *Distance and Speed Estimation*

The frame of a detected vehicle is converted to a bird's eye view as shown in Figure 9. The length to the vehicle is calculated by measuring the pixel height from base to the lowest green pixel in frame. This height needed to be converted from a pixel length to an actual length, this process was carried out by measuring the pixel length taken up for known object of distance placed on the plane of the road A black marker of a known distance was placed in the line of sight of the camera and the pixel length of the marker was obtain. The actual length to the vehicle from the camera l_a in (m) was determined as:

$$l_a = \frac{h \times l_p}{l_a} \tag{1}$$

Where,

l_a – the actual length to the vehicle from camera (m)

l_m- marker length (m)

l_p- pixel length of marker

3) *Speed Estimation*

The instantaneous speed at frame *n*, of the vehicle was determined in Realtime as:

$$v_{ins}(n) = \frac{l_a(n) - l_a(n-1)}{f} \tag{2}$$

Where,

$l_a(n)$ - the length value obtained for current frame (m)

$l_a(n-1)$ - the length value obtained for the previous frame (m)

f – frame rate (frames/sec)

n - n^th frame with the presence of detected vehicle

The instantaneous speeds of the vehicle are recorded until the vehicle leaves the range of the camera frame. The average speed v_{avg} of the vehicle before entering the range of ultrasonic sensors is calculated as:

$$v_{avg} = \frac{\sum_{i=1}^{n} v_{ins}(i)}{n} \tag{3}$$

4) Obtaining vehicle dimensions

The presence of a vehicle is detected by ultrasonic sensors when the difference between the current height values h_1 and h_2 and the previous height values h_1' and h_2' of the two sensors exceeds a threshold value assigned by the user.

Time t_1 is the initial time at which the presence of a vehicle is detected by the pair of Ultrasonic sensors and t_2 is the time at which the height difference recorded at sensors is returned to a value within the threshold.

The lowest height value recorded by either one of the sensors (S_1 and S_2) is used to determine the maximum height of the vehicle. This max height h_{max} is equivalent to the difference between the height obtained at time t_2 and the lowest height value recorded .

5) Vehicle classification

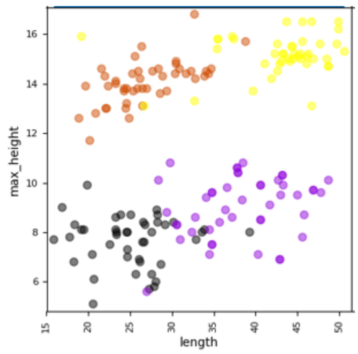

Figure 11: Scatter plot of length and max height

From the scatter matrix shown in Figure 10 it can be seen that the input data (max height and length) of the 4 classes of vehicles (represented by the 4 different colours of scatter points) is spread into 4 regions, with each region belonging to one class. From this observation k-NN algorithm [19] was chosen as the classification algorithm since it uses the concept of predicting class label based on the actual label of its nearest neighbors.

k-NN is considered a 'supervised' classification as it uses true class labels of training data to train the classifier and predict the label the of test data. Therefore, each model vehicle was run multiple times and a dataset was created by recording class label, max height and length obtained. This dataset was split into training and testing categories. Once the system was trained with the dataset and the k-nearest neighbors were found the classification system was implemented in the Raspberry pi 3 to classify vehicles in real time.

IV. RESULTS AND DISCUSSION

TABLE I. PERFORMACE OF CNN MODELS

Model	Image size	Epoch	Train acc.	Val acc.	Duration (s)	Size (MB)
1	28x28	25	87.0%	88.1	7.13	14.3
2	28x28	25	85.1%	87.7	10.56	14.3
3	28x28	25	93.2%	95.7	17.77	14.3
4	28x28	50	93.8%	91.7	21.75	14.3
5	64x64	50	88.4%	87.7	63.78	73.6
6	128x128	50	88.4%	87.7	183.5	293

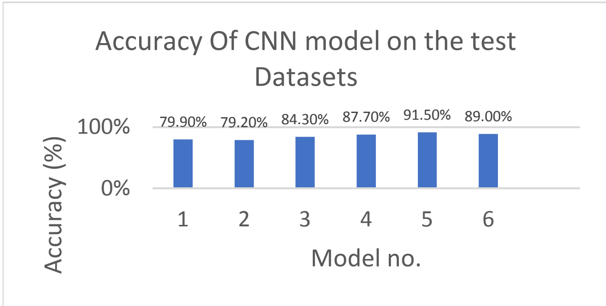

Figure 12: Performance of each model on test dataset

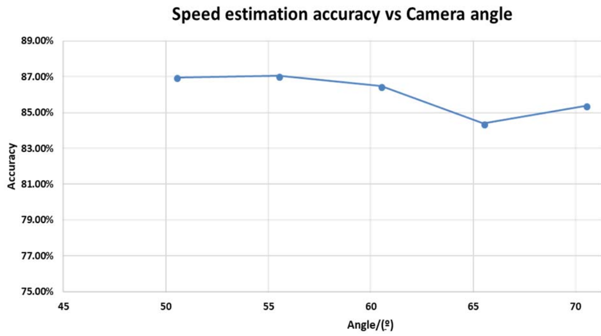

Figure 13: Performance of speed estimation

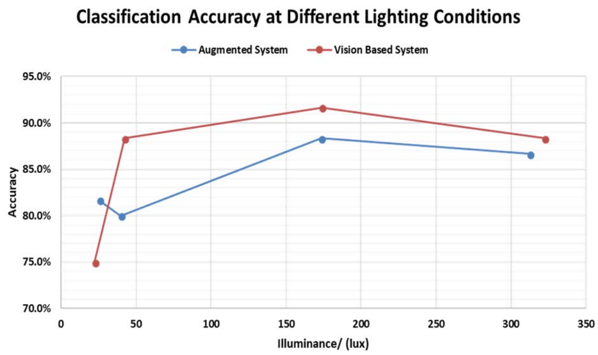

Figure 14: Performance under varying lighting conditions

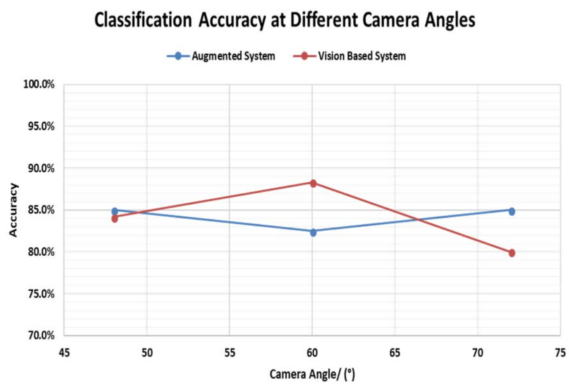

Figure 15: Performance under varying camera angles

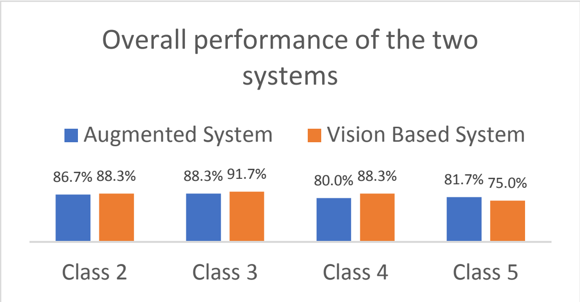

Figure 16: Overall Classification accuracies of vision-based system

The performance of the model created in Lenet architectur was analysed by varying:

- Size of image dataset
- Number of Epochs
- Resolution of images used in the dataset

Training results of each model created has been shown on Table I.

The training dataset has a large influence on the accuracy of the CNN model created and increasing the size of dataset allowed better training of weights at the convolution and dense layers of the model. Increasing the number of epochs allowed the weights of the model to reach a more optimum solution which decreased overall loss function and improved the accuracy of the mode. The Increase in the resolution of images led to the increase in training time which was coupled with a large increase in size of the resultant models. The best model in terms of validation and testing accuracy is model 3 since it had the highest validation accuracy of 95.67% and a high training accuracy of 93.18%.300 false positive images were introduced for models 5,6,7 to create model with to classify falsely detected non-vehicle images

These models 1, 2 and 3 were then implemented on *test dataset A* which consisted a set of 159 newly obtained images of the 4 classes of vehicles arranged randomly. Next, model 4, 5, 6 were implemented with *test dataset B* which consisted of the images obtained for test dataset A with additional set of false positive images. The results are shown in Figure 11.

Model 5 performed the best out of the 6-model created with an overall classification accuracy obtained of 91.5%. Models 6 and 4 also managed to obtain high accuracies of 89% and 87.7% respectively. Since, model 6 was evaluated as the best model in to of classification accuracy, this model was implemented on the Raspberry Pi 3 to classify model vehicles in real time.

Speed estimates with consistent accuracies were obtained over the allowable range of camera angles as shown in Figure 12. The drop-in accuracies at angles close to the maximum range was caused due to the fewer frames detecting moving vehicle which resulted in fewer instantaneous speeds obtained and thus reducing the accuracy of the average speed obtained.

As shown in Figure 13 , the classification accuracies of both systems reduced as the amount of lighting available reduced. The augmented system performed better than the vision-based system at low lighting conditions since it did not require the entire moving vehicle to be detected. The vision-based system was more prone to error since the feature patterns were less visible at low lighting conditions.

From Figure 14, the accuracy of vision-based system is seen to drop when the camera angle is increased. Increased angle displays fewer features of the vehicles since only the frontal face of the vehicle is seen at camera angles close to the maximum limit.

Figure 15 shows that the accuracies obtained for the two-system differed among vehicle classes. The model vehicle of Class 4 was at times classified by the CNN models as a vehicle of Class 5 since the model truck which belonged to class 4 was attached to a trailer and used as the vehicle model for class 5 which resulted in both model vehicles having a similar appearance, this caused the classification accuracy of vision-based system to drop for class 5.The vision based system was found to be better classifier than augmented system for vehicle classes 2, 3 and 4 and therefore can be considered as the better classifier out of the two systems.

V. Conclusion

In conclusion, two vehicle classification algorithms and a speed estimation algorithm has been considered, to be implemented on a Raspberry pi. The Vision Based system obtained an overall classification accuracy of 85.8% for four vehicle classes with the augmented system a slightly lower accuracy of 84.2%. The estimation algorithm developed obtained an overall accuracy 86.05% and was less error prone to varying lighting conditions. Both system performed well for varying lighting conditions and camera angles with slight drop in accuracy at low lighting conditions and at camera angle close to the maximum range allowed by the camera. This system is of low cost and can be implemented on Highway Gantries, toll gates and at traffic lights with minimum calibration required.

References

[1] Yousaf, K., Iftikhar, A. and Javed, A., 2012. Comparative analysis of automatic vehicle classification techniques: a survey. International Journal of Image, Graphics and Signal Processing, 4(9), p.52.

[2] Harlow, C. and Peng, S., 2001. Automatic vehicle classification system with range sensors. Transportation Research Part C: Emerging Technologies, 9(4), pp.231-247.

[3] Meta, S. and Cinsdikici, M.G., 2010. Vehicle-classification algorithm based on component analysis for single-loop inductive detector. IEEE Transactions on Vehicular Technology, 59(6), pp.2795-2805.

[4] Wicaksono, D.W. and Setiyono, B., 2017. Speed Estimation On Moving Vehicle Based On Digital Image Processing. International Journal of Computing Science and Applied Mathematics, 3(1), pp.21-26.

[5] Robert, K., 2009, September. Night-time traffic surveillance: A robust framework for multi-vehicle detection, classification and tracking. In Advanced Video and Signal Based Surveillance, 2009. AVSS'09. Sixth IEEE International Conference on (pp. 1-6).

[6] Gajda, J., Sroka, R., Stencel, M., Wajda, A. and Zeglen, T., 2001, May. A vehicle classification based on inductive loop detectors. In Instrumentation and Measurement Technology Conference, 2001. IMTC 2001. Proceedings of the 18th IEEE(Vol. 1, pp. 460-464).

[7] Kim, S.E., 1998. Performance comparison of loop/piezo and ultrasonic sensor-based traffic detection systems for collecting individual vehicle information. In TOWARDS THE NEW HORIZON TOGETHER. PROCEEDINGS OF THE 5TH WORLD CONGRESS ON INTELLIGENT TRANSPORT SYSTEMS, HELD 12-16 OCTOBER 1998, SEOUL, KOREA. PAPER NO. 4083.

[8] Kim, S.W., Kim, K., Lee, J.H. and Cho, D.I.D., 2001. Application of Fuzzy Logic to Vehicle Classification Algorithm in Loop/Piezo□ Sensor Fusion Systems. Asian Journal of Control, 3(1), pp.64-68.

[9] Goyal, A. and Verma, B., 2007, April. A neural network based approach for the vehicle classification. In Computational intelligence

in Image and Signal Processing, 2007. CIISP 2007. IEEE Symposium on (pp. 226-231).

[10] S. Cheung, S. Coleri, B. Dundar, S. Ganesh, C. Tan and P. Varaiya, "Traffic Measurement and Vehicle Classification with Single Magnetic Sensor", Transportation Research Record: Journal of the Transportation Research Board, vol. 1917, pp. 173-181, 2005.

[11] Taghvaeeyan, S. and Rajamani, R., 2014. Portable roadside sensors for vehicle counting, classification, and speed measurement. IEEE Transactions on Intelligent Transportation Systems, 15(1), pp.73-83.

[12] K. Ohtani, M. Baba and T. Konishi, "Position and posture measurements and shape recognition of columnar objects using an ultrasonic sensor array and neural networks", Systems and Computers in Japan, vol. 33, no. 11, pp. 27-38, 2002.

[13] J. Hsieh, S. Yu, Y. Chen and W. Hu, "Automatic Traffic Surveillance System for Vehicle Tracking and Classification", IEEE Transactions on Intelligent Transportation Systems, vol. 7, no. 2, pp. 175-187, 2006

[14] M. Kafai and B. Bhanu, "Dynamic Bayesian Networks for Vehicle Classification in Video", IEEE Transactions on Industrial Informatics, vol. 8, no. 1, pp. 100-109, 2012.

[15] Upton, E. and Halfacree, G., 2014. Raspberry Pi user guide. John Wiley & Sons..

[16] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P., 1998. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11), pp.2278-2324.

[17] Chollet, F., 2017. Deep learning with python. Manning Publications Co..

[18] Tuohy, S., O'Cualain, D., Jones, E. and Glavin, M., 2010. Distance determination for an automobile environment using inverse perspective mapping in OpenCV.

[19] Cunningham, P. and Delany, S.J., 2007. k-Nearest neighbour classifiers. Multiple Classifier Systems, 34, pp.1-17.

[20] Zhang, Z., 1999. Flexible camera calibration by viewing a plane from unknown orientations. In Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on (Vol. 1, pp. 666-673..