DOI: http://dx.doi.org/10.26483/ijarcs.v9i1.5350

ISSN No. 0976-5697

Volume 9, No. 1, January-February 2018

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

VEHICLE NAVIGATION USING ADVANCED OPEN SOURCE COMPUTER VISION

Ms. Deipali Gore
Assistant Professor, PES MCOE
Department of Computer Engineering
Pune, Maharashtra, India

Paritosh Medhekar
Final Year Student
Department of Computer Engineering
Pune, Maharashtra, India

Soubhik Das
Final Year Student
Department of Computer Engineering
Pune, Maharashtra, India

Ameya Kale
Final Year Student
Department of Computer Engineering
Pune, Maharashtra, India

Yash Gugale
Final Year Student
Department of Computer Engineering
Pune, Maharashtra, India

Abstract: The current operational transport and vehicle systems consist of vehicles running on fossil fuels or battery powered systems. The navigation requires control by a human driver who is responsible for a safe and comfortable journey from one place to another. However, with human intervention there are several drawbacks that may lead to a poor performance by the system. Negligence in driving leading to fatal accidents, environmental damage, infrastructure damage and destruction, health problems due to constrained sitting postures, long duration of operation and several others, have motivated researchers to look for solutions that will automate the driving process. Considering all these shortcomings of current systems, the new research consists of the use of self driving cars for transport and navigation. The complexity of this problem was seen when the initial systems were built using machine learning techniques that tried to understand and model the dynamic nature of the environment. As the research progressed, we realized that the system must be trained to respond to a number of unpredictable situations such as rain, snow, lightning, oil spills, potholes, passerby pedestrians and animals, approaching vehicles and many more. We need to consider all these aspects before a fully functional real-time system can be used. We consider the problem of autonomous vehicle by focusing on three major aspects of any self driving car which form the foundation of the entire system. Firstly, we need to be able to detect the lane lines so that our vehicle can orient itself correctly and continue to follow a safe path while being aware of the dynamic environment. Further, it needs to know its departure from the center of the lane in the scenario that it needs to move in order to avoid potholes or other road obstacles.

Keywords: autonomous driving, vehicle navigation, OpenCV, Linear SVM, Canny Edge, CNN

I. INTRODUCTION

A vehicle navigation [1] or automotive navigation system [2] is used to find the direction of an automobile [3]. A satellite navigation system can be used to get the exact position on road. Sensors are also used along with accelerometers and gyroscope for accuracy. Autonomous vehicle is capable of fulfilling human transportation abilities of a traditional vehicle.

The first step is to sense the surrounding environment. That is done using techniques like radar, Lidar, OpenCV (open source computer vision) and GPS [4]. We have defined vehicle navigation as combination of three different competences: lane detection, vehicle detection and traffic sign detection as well as classification. Lane departure and detection is used to guide the vehicle along the path smoothly. Vehicle detection is used to identify all the nearby vehicles and objects and behave accordingly on road. Traffic sign module is used to instruct the vehicle to behave according to the traffic signs on road. As far as mobile robots like automatic driving, exploration of dangerous regions [5], etc. are concerned, there are many potential application areas of autonomous navigation.

The paper is proposed to develop a vehicle navigation agent used in intelligent vehicle [6]. The proposed system has a pedestrian detection and road safety signs detection along with lane detection modules that can perform in close to real-time based on visual cues alone. This video-only detection [6] makes systems for spotting pedestrians and other objects highly effective. Existing cars use expensive hardware such as Lidar, Radar, Velodyne Lasers etc. to detect pedestrians and other objects such as traffic signs on the road. Getting rid of some of that equipment could make the cars cheaper and easier to design.

One of the solutions is to develop the said system using Open source computer vision [7] alone. The said agent perceives the dynamic environment using action camera and computes using captured live feed. The lane lines [7] are marked using Canny Edge and the curvature as well as distance from road is calculated.

II. LITERATURE SURVEY

A. Lane Marking

Using, decision of edge detection algorithm [8] is made. The following algorithms exist for marking lane lines:

1) Sobel Operator

The operator consists of a pair of 3X3 convolution kernels. When one kernel is rotated by 90', we get the other. Separate measurements of the gradient component [8] in each orientation can be produced by applying kernels individually to the input image, which is one if the ways.

2) Canny Edge Algorithm

The steps for this algorithm [9] are as follows:

Step 1: Perform noise filtering before edge detection. Then apply Gaussian smoothing using standard convolution method.

Step 2: Taking into consideration gradient of the image, calculate edge strength [9]. Compute the edge direction both x and y direction. When the said gradient in x direction reaches zero, the edge direction equals to 90' or 0'.

Step 3: The image should be traceable. The edge direction [9] to a direction should be related so that the above condition is satisfied.

Step 4: Apply Non-maximum suppression [9] to trace along the edge in the edge direction .This will give a thin line in the output image. A thin line helps in clarity. All these make it more distinct.

Step 5: Streaking is eliminated using Hysteresis. The edge contours caused operator output fluctuating both above and below the threshold is broken by streaking.

3) Robert's Cross Operator

Robert's Operator is very similar to Sobel Operator [9] and contains multiple 2X2 convolution kernels. Here, the gradient of an image through discrete differentiation m is calculated mathematically.

For various regions of an image, Canny Edge Algorithm is taken into account. The algorithm yields thin lines for its image and taking into consideration all the factors; it becomes clear that Canny Edge algorithm yields the best results.

B. Vehicle Detection

Reference uses object detection with the help of Linear SVM [10] and HOG [11]. Following are the algorithms used here:

1) Linear SVM Classifier

Define a function to which we pass an image and the list of windows to be searched Parameters:

a. The list of windows to be searched.

b. The classifier used. Here, we are using the Linear SVM classifier as we have a linear kernel and other n-dimensional kernels like Gaussian, tan, etc are not required.

2) Hog feature extraction

The HOG descriptors count the occurrences of gradient orientation in localized portions of an image. Collect all the RGB channels and append to an array. The image must be a colored image and can be checked by calling 'image.shape' and checking if the third parameter is 3. If yes, then the image is colored image. This is the main HOG feature extractor function. We need to pass it several arguments and if not given explicitly, then the default values

are taken. Function of each parameter: images: pass the image vector as input color space: pass the desired color space. (It can take the values as : RGB, HSV, LUV, YUV, YCrCb).

3) Radar data fusion

Fade, a vehicle detection and tracking system features monocular color vision and radar data fusion. At each step and for each target, the fusion system fuses the results of four different image processing algorithms and radar information by automatically combining 12 different features and generating many possible target position proposals. It generates a belief network organized in three layers: sources, position proposal, and correlation between proposals.

4) Linear SVM classifiers and HOG feature extraction have the best accuracy and performance. Radar data fusion is often successful at improves the spatial resolution; however, they tend to distort the original spectral signatures to some extent.

C. Traffic Sign Classification

The dataset for training and testing [12] was decided using. Following are the algorithms used:

1) Convolution neural networks

Convolution Neural Networks are related to Neural Networks [12] or Neural Nets. They consist of a series of neurons that have learnable weights and biases, along with related components and attributes. The neurons act just like they do in the human brain. Every neuron [12] receives some random specific inputs, performs a vector dot product and finally (optionally) follows it with a non-linearity.

2) Adam Optimizer

This is an optimization algorithm which is different from Neural Networks and can replace classical stochastic gradient [12] descent procedure. It makes use of optimization functions. It is used to update iterative based network weights in training data.

3) Sigmoid activation function: The works of saturated neurons [13] is to kill the other neurons through back propagation.

4) Relu (Rectilinear Linear Unit) [13] activation function: They are initialized with slightly positive biases as compared to others as dead Relu will activate/update.

Convolution neural networks have an advantage as they allow networks to have fewer weights and they are given a highly effective tool - convolutions - for image processing, which Laplacian filters don't. Laplacian filters are further less accurate and therefore less efficient. Adam optimization is mostly used as it can be used to update network weights efficiently. Relu activation function is used as Relu function does not saturate in positive region and is computationally efficient as well as faster than sigmoid.

III. MODULE WORKING USING OPENCV

There was an option to build a vehicle navigation agent [14] using expensive hardware but we are building it open source computer vision. This is because being economical is an advantage of open source over expensive hardware. Few challenges exist in deployment like invisible lane markings, small sized traffic signs [14] and the scenario being less illuminated.

Self-driving cars have been subject of extensive research and development. Every autonomous car would contain the following core modules as a part of the whole system:

1) Lane line detection

2) Traffic sign classification

3) Vehicle detection

A. Camera Calibration

Image distortions are some of the biggest issues faced today in cameras. To optimize performance, they have to be reduced, if not cut down. One of the ways to avoid image distortion is camera calibration [15]. A whole lot of distortion to images is introduced by today's pinhole

cameras which is tough to handle. There are many types of distortion which can degrade the image and major types among them are radial and tangential distortion. Radial distortion is one of the worst possible radiations simply due to the fact that straight lines generally appear curved here. The distortion keeps increasing as move away from the image center [15].

For obtaining 3D object points images are taken from a stationary camera and chess boards placed at different locations with different orientations. Python packages are imported and OpenCV is extensively used here. Camera calibration is done using given object points and also image points by using OpenCV library and line 'cv2.calibrateCamera() :'.

B. Architectural design

The autonomous cars will help driver assistance evolve gradually using these modules. The agent sensory apparatus [15] does not in any way give access to the complete state of the environment and thus that environment is said to be in-accessible to the agent. The environment is deterministic [15] given it is inaccessible and can change dynamically while the agent is deliberating and changing continuously. In that case, the environment is for sure dynamic.

For detecting lane lines from a continuous video stream [Figure 1.], lane detection method [16] requires presence of lane markings on the road. If the sizes of the traffic signs are very small, then signs would not be detected. Traffic sign detection module is largely impacted by illumination as performance accuracy decreases at night or in bad weather.

For lane detection, marking lanes and calculation of curvature of the lane [16], distance from the centre of the lane, first the camera calibration is computed and distortion coefficients when a set of chessboard images is given. Distortion correction is applied on the raw image. A threshold binary image [16] is created using color transforms, gradients, etc. A perspective transform is applied to rectify the binary image. Radius (curvature) of the lane is detected and the previously detected lane boundaries are twisted out of shape to the original image at last. The detected as well as marked lane lines and their numerical estimation are displayed [16] finally. .

For traffic sign detection and classification, first the dataset is loaded. Then, the data set is explored, summarized and visualized. The model architecture is designed, trained and tested. Then the model is used to make predictions. Softmax probabilities of the new images are analyzed. Ultimately, the results are summarized.

For vehicle detection module [17], color transformation is applied before Histogram Oriented Gradients (HOG) extraction function is performed on a labeled training set of images and a classifier is trained using Linear SVM. There is a need for searching vehicles in images. This is implemented using a sliding window technique with the help of a trained classifier [17]. Our pipeline is run on an input video stream and heat map is created of recurring detections frame by frame to reject some outliers as well as irregular data and follow detected vehicles. When the vehicles are finally detected, a bounding box [17] is created around them.

The final output is annotated video feed [17] from all the modules.

The figure [Figure 1] below shows architectural block diagram.

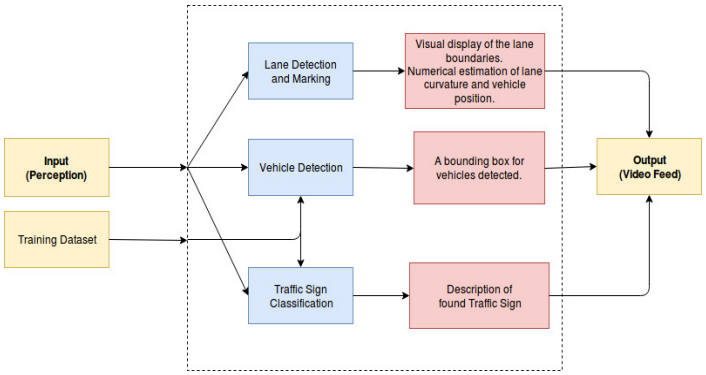

Figure 1. Architectural Block Diagram

C. Dataset Design

Two among the three modules require some form of data on which computation results are expected. The following table shows the description of the datasets used.

D. Testing

Following are some of the test [Table I] images:

Table I

Dataset	No. of Training Samples	No. of Testing Samples	Size	Format	Purpose
GTSRB	39209	12630	32X32	.ppm (portable pixmaps)	Traffic Sign Classification
KITTI Vision Benchmark suite	4000	3425	64X64	.jpeg (Joint Photographic Experts Group)	

a. Lane Detection Module [Figure 1, Figure 2, Figure 3]

Figure 2. Test Image

Figure 3. Test Image

b. Vehicle Detection Module [Figure 4]

Figure 4. Test Image

c. Traffic Sign Recognition Module [Figure 5, Figure 6]

Figure 5. Test Image

Figure 6. Test Image

Testing is also known as validation. Training helps us in checking the accuracy of our trained model. The various types of testing are unit testing, integration testing, positive testing, system testing and negative testing.

IV. INTEGRATION OF THE MODULES

Initially, all the modules run individually. A Jupyter Notebook is created and the python file [18] is executed. Then, they are integrated and merged in a single window.

The figure shows the use case diagram [Figure 7].

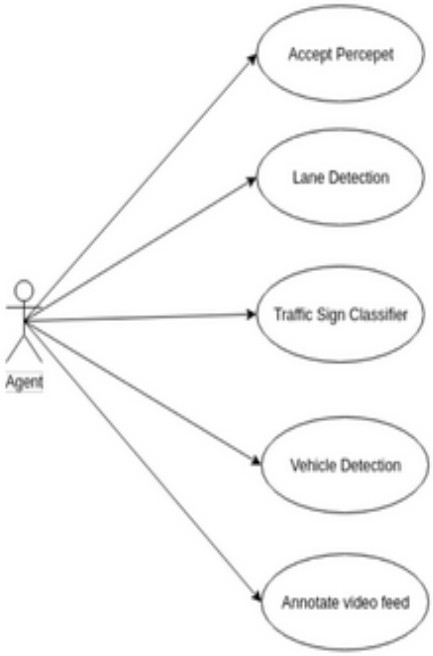

Figure 7. Use Case Diagram

The figure shows the state transition diagram.

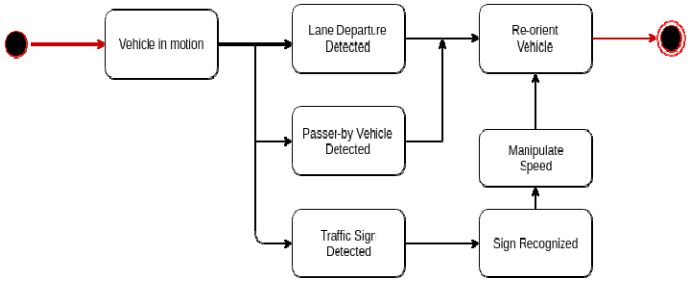

Figure 8. State Transition Diagram

V. RESULTS

5.1 Software Testing

Among software testing [19], various phases of testing were carried out. They are as follows:

Table II

	ID	Description	Scenario	Expected Output
	1	Lane Marking using Canny Edge algorithm and Sobel operator	Lane has been detected and then marked	Lane lines detected on road

A. ***Unit Testing***

1) Lane Detection

Table III

Type of Testing used:

- Unit Testing
- Integration Testing
- System Testing
- Positive Testing
- Negative Testing

2) Vehicle Detection

5.1.2 Test Cases and Results

ID	Description	Scenario	Expected Output
2	Vehicle Detection using Linear SVM and HOG extractor function	Vehicle Detection	Marked vehicles on road

3) Traffic Sign Recognition

Table IV

ID	Description	Scenario	Expected Output
3	Traffic sign correctly classified	Traffic sign detected and classified by CNN with enough accuracy	Annotate video frame with traffic sign interpretation

B. ***Integration Testing***

Table V

ID	Description	Scenario	Expected Output
1	Lane Marking using Canny Edge algorithm and Sobel operator	Lane has been detected and then marked	Lane lines detected on road
2	Lane departure detected	If distance from the centre of the lane threshold	Annotate video frame with the warning
3	Vehicle Detection using Linear SVM and HOG extractor function	Vehicle Detection CNN with enough accuracy	Marked vehicles on road interpretation
4	Traffic sign detection using CNN and Relu activation function	Traffic sign detection and classification	Detected traffic sign and their classification
5	Traffic sign correctly classified	Traffic sign detected and classified by CNN with enough accuracy	Annotate video frame with traffic sign interpretation

C. System Testing

1) Normal Road

Table VI

ID	*Description*	*Scenario*	*Expected Output*
1	Road lane marking	Live video feed	Detected and marked lane lines
2	Road vehicle detection	Live video feed	Detected vehicles on road
3	Road traffic sign detection	Live video feed	Detected and classified traffic signs

.D. *Positive Testing*

Scenario: Roads with lane lines, vehicles and traffic signs

Table VII

ID	*Description*	*Scenario*	*Expected Output*
1	Road lane marking	Live video feed	Detected and marked lane lines
2	Road vehicle detection	Live video feed	Detected vehicles on road
3	Road traffic sign detection	Live video feed	Detected and classified traffic signs if no obstruction

E. Negative Testing

Scenario: Empty road with no lane lines, no vehicles and no traffic signs.

Table VIII

ID	*Description*	*Scenario*	*Expected Output*
1	Road lane marking	Live video feed	No lane lines detected
2	Road vehicle detection	Live video feed	No vehicles detected
3	Road traffic sign detection	Live video feed	No traffic signs recognized

F. *Prototype Implementation*

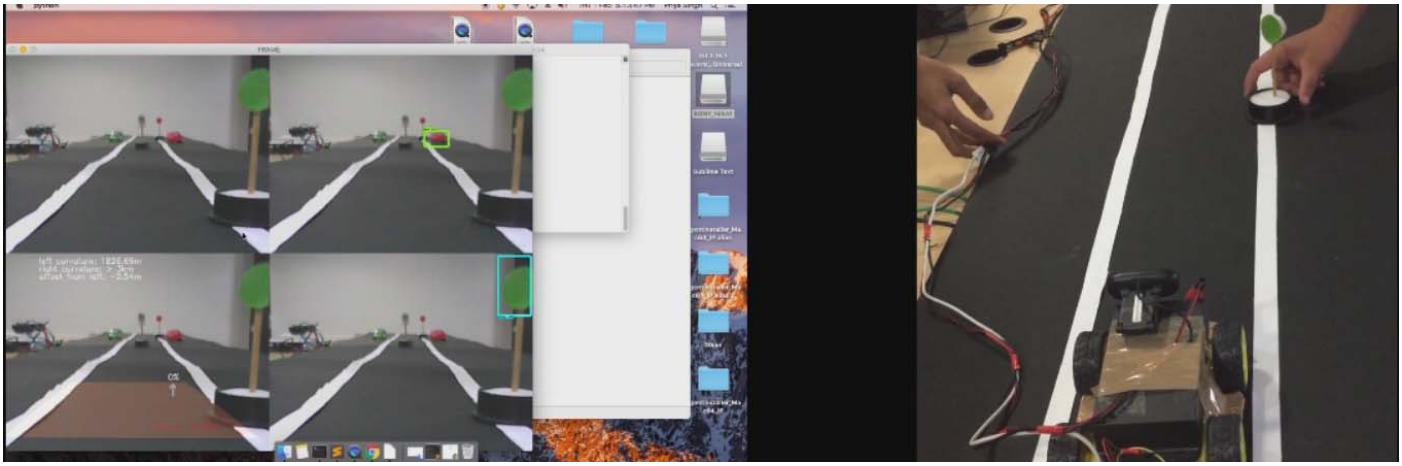

Figure 9. Sample testing on an artificial track

(clockwise starting from top right:Vehicle Detection window,
Traffic Sign Detection Window and Lane Marking window)

We tested our algorithms on a toy car running on a five metres long track and showed accuracy close to 85% [Figure 9]. The traffic sign module was initially trained onthree signs (No Parking, Stop sign and No Horn) and later onmany more signs.

VI. CONCLUSION

The project addresses the problem of building an autonomous driving agent that can assist vehicle even after facing various obstacles while navigation. As the system is highly cohesive and loosely coupled, the accuracy of the system is higher than existing systems. Today's car crash-avoidance systems and experimental driverless cars rely on radar and other sensors to detect pedestrians on the road. Eg. Google's robotic cars (Waymo) have about 150,000 USD in equipment including a 70,000 USD LIDAR system.

The three modules of the proposed system based on computer vision are lane departure detection, vehicle detection and traffic sign classifier. The lane departure module is capable of detecting lane
lines, calculating curvature of lane lines and detecting lane departure by calculating the distance from the center of the lane using Canny edge detection and binary thresholding. The vehicle detection module is capable of detecting the passer by vehicles using linear SVM and Histogram Oriented Gradients. This module can be extended to detect pedestrians using similar technique. The traffic sign classifier utilizes technologies like TensorFlow, Convolution Neural Networks for recognition of traffic signs. This module assists the system to enforce traffic rules in efficient way.

Modern deep Convolution Neural Networks can outperform humans [20] in tasks such as recognizing objects, with accuracy rates of over 99.5 percent.

VII. REFERENCES

[1] Regan, Michael A., et al. "Acceptability of in-vehicle intelligent transport systems to Victorian car drivers." *Monash University Accident Research Centre* (2002).

[2] Suzuki, Keisuke, and Håkan Jansson. "An analysis of driver's steering behaviour during auditory or haptic warnings for the designing of lane departure warning system." *JSAE review* 24.1 (2003): 65-70.

[3] Stefan, Jeffrey Michael, Dana Brian Fecher, and Gregory Howard Williams. "Vehicle navigation system having inferred user preferences." U.S. Patent No. 6,212,473. 3 Apr. 2001.

[4] Cui, Youjing, and Shuzhi Sam Ge. "Autonomous vehicle positioning with GPS in urban canyon environments." *IEEE transactions on robotics and automation* 19.1 (2003): 15-21.

[5] Li, Qingquan, et al. "A sensor-fusion drivable-region and lane-detection system for autonomous vehicle navigation in challenging road scenarios." *IEEE Transactions on Vehicular Technology* 63.2 (2014): 540-555.

[6] Lorsakul, Auranuch, and Jackrit Suthakorn. "Traffic sign recognition using neural network on OpenCV: Toward intelligent vehicle/driver assistance system." *4th International Conference on Ubiquitous Robots and Ambient Intelligence*. 2007.

[7] Zhu, Qiang, et al. "Fast human detection using a cascade of histograms of oriented gradients." *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*. Vol. 2. IEEE, 2006.

[8] Maini, Raman, and Himanshu Aggarwal. "Study and comparison of various image edge detection techniques." *International journal of image processing (IJIP)* 3.1 (2009): 1-11.

[9] Low, Chan Yee, Hairi Zamzuri, and Saiful Amri Mazlan. "Simple robust road lane detection algorithm." *Intelligent and Advanced Systems (ICIAS), 2014 5th International Conference on*. IEEE, 2014.

[10] Sun, Zehang, George Bebis, and Ronald Miller. "On-road vehicle detection using Gabor filters and support vector machines." *Digital Signal Processing, 2002. DSP 2002. 2002 14th International Conference on*. Vol. 2. IEEE, 2002.

[11] Sivaraman, Sayanan, and Mohan Manubhai Trivedi. "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis." *IEEE Transactions on Intelligent Transportation Systems* 14.4 (2013): 1773-1795.

[12] The German Traffic Sign Recognition Benchmark paper by Turk, Matthew A., et al. "VITS-A vision system for autonomous land vehicle navigation." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10.3 (1988): 342-361.

[13] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).

[14] Driankov, Dimiter, and Alessandro Saffiotti, eds. *Fuzzy logic techniques for autonomous vehicle navigation*. Vol. 61. Physica, 2013.

[15] Neto, A. Miranda, et al. "Robust horizon finding algorithm for real-time autonomous navigation based on monocular vision."

[16] Stafylopatis, Andreas, and Konstantinos Blekas. "Autonomous vehicle navigation using evolutionary reinforcement learning." *European Journal of Operational Research* 108.2 (1998): 306-318.

[17] Guo, Chunzhao, Seiichi Mita, and David McAllester. "A vision system for autonomous vehicle navigation in challenging traffic scenes using integrated cues." *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*. IEEE, 2010.

[18] Ujjainiya, Lohit, and M. Kalyan Chakravarthi. "Raspberry—Pi Based Cost Effective Vehicle Collision Avoidance System Using Image Processing." *ARPN J. Eng. Appl. Sci* 10 (2015): 3001-3005.

[19] Souza, Jefferson R., et al. "Template-based autonomous navigation in urban environments." *Proceedings of the 2011 ACM Symposium on Applied Computing*. ACM, 2011.

[20] Bojarski, Mariusz, et al. "End to end learning for self-driving cars." *arXiv preprint arXiv:1604.07316* (2016).