ESTEEM Academic Journal
Vol. 16, June 2020, 74-85

Pre-Processing Methods for Vehicle Lane Detection

Muhammad Naim Mazani[1], Shuzlina Abdul-Rahman[2*] and Sofianita Mutalib[3]
[1,2,3]Research Initiative Group of Intelligent Systems, Faculty of Computer and Mathematical Science Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia

**corresponding author: shuzlina@fskm.uitm.edu.my*

ARTICLE HISTORY

Received
23 April 2020

Accepted
3 June 2020

Available online
30 June 2020

ABSTRACT

This study presents pre-processing methods for detecting lane detection using camera and Light Detection and Ranging (LiDAR) sensor technologies. Standard image processing methods are not suitable for complicated roads with various sign on the ground. Thus, determining the right techniques for pre-processing such data would be a challenge. The objectives of this study are to pre-process the scanned images and apply the image recognition algorithm for lane detection. The study employed Canny Edge Detection and Hough Transform algorithms on several sets of images. A different region of interest was experimented to find the optimal one. The experimental results showed that the proposed algorithms could be practical in terms of effectively detecting road lines and generate lane detection

Keywords: *image pre-processing; lane detection; mapping; region of interest*

1. INTRODUCTION

Technology is evolving rapidly and revolutionising the standard driving style. However, the safety of the vehicle itself needs to be guaranteed by ensuring that the car can position itself on the lane with a certain boundary. As the advancement in technology is on the rise, some of the technology will give impact to the society, which will improve vehicle safety in driving that can ensure the safety of the passengers and the driver. The technology can assist the driver, for example, the Advanced Driver Assistance System (ADAS) which can detect some objects, do basic classification, and assist lane-keeping [1]. Nowadays, most cars are equipped with ADAS technology that is useful for road navigation.

The autonomous vehicle is one of the future technologies in the new generation, which can be the potential solution to reduce road accident deaths. Thompson and Read [2] stated that road accidents took more than a million lives a year and injured nearly fifty million more globally. Therefore, an autonomous vehicle can be one of the solutions to this problem. This technology was able to identify the situation by learning to avoid serious cases that lead to accidents. According to [3], autonomous vehicle technology enables the journey to be safer and more efficient for the passengers without the need for human intervention. Commercial companies and academic institutions are currently developing new driver assistances and fully autonomous systems to navigate the vehicle.

Lane detection plays a major part in vehicle sensing in the context of road detection. Although many sensors have been implemented, such as radar and camera, there are still significant deaths each year in road departure crashes between vehicles that are caused by driver inattention [4]. Sensors such as radar which possess machine sights lack in performance today to detect all

p-ISSN 1675-7939; e-ISSN 2289-4934

important features with the reliability necessary for safe driving. On 23rd March 2018, a fatal Tesla car crashed into a roadside barrier and caught on fire while on autopilot, which also resulted in the death of the driver [5]. Using LiDAR, another type of sensor which uses laser scanning would provide better accuracy compared to radar sensors.

Based on the study by [6], most approaches used in lane detection are camera-based where the input images are fed into the system to detect lines. As an example, a Hough transform (HT) is one of the very typical methods and has been widely applied to computer processing, image processing, and digital image processing [7]. The advantages of using the image to detect lines can be seen in the utilisation of colour and the whole environment features. It is also common to use grayscale or full-colour in analysing the image to catch the line. However, evaluating the depth of the lines using the image is inaccurate. Therefore, some methods started to combine LiDAR sensor to increase the accuracy of the result of detection [6]. The main objective of this study is to apply image recognition algorithm using two different sensors for lane detection. The work which develops image recognition is separated into two parts which is data pre-processing and model development. The remainder of this paper is written in four sections. Section 2 is an overview of the technology and the algorihms involved. Details of how the study was conducted are given in Section 3. Section 4 presents the results and discussion of the study.

2. LITERATURE

2.1 Sensor Technology

Sensor technology is a device that is capable of detecting or measuring specific features and responds to certain input from the world, such as light, motion, heat. Most common sensors for vehicle sensing are camera and radar. In driving, camera sensors are assumed to be sufficient because driving primarily requires vision to observe the environment [8].

In order for a vehicle to have 360 degrees view of the surrounding, multiple camera sensors need to be equipped on the car to cover every side. In order to estimate the distance, the stereoscopic camera is used alongside algorithms to achieve the task. However, camera sensors are inefficient when it comes to adjusting range in the night while maintaining good image quality [8]. The inefficiency may cause by the colour that is almost similar at night due to less light surroundings. Situations where the contrast is low and colour of the objects overlap with the background becomes dangerous for autonomous driving with camera reliance that was meant for daytime lighting. Additional sensors are required for a safe driving system.

LiDAR is a short form Light Detection and Ranging and a well-known outdoor sensing method used for measuring the exact distance of objects such as structures, buildings, roads and bridges. According to [9], LiDAR was first used as laser scanners mounted onto airplanes in 1960, but most people do not care about its existence until twenty years later. After the Global Positioning System (GPS) was introduced, LiDAR turned out as one of the popular methods for calculating geospatial measurements. LiDAR is comparable to radar, apart from the fact that the sensor technology shoots laser beams that are harmless to the eyes [10]. The technology can create a 3D representation of the observed environment. LiDAR is also used in many industries and sectors, such as automotive, trucking, UAV/drones, industrial, mapping, and others.

2.2 Image Processing

Image processing is known to perform operations on an image to get an enhanced version of the image and also extract some useful features and information from the image. Image processing can do pattern and pixel analysis of any images to recognise the images as any particular or specific objects. It can visualise the environment as pixel and images that can be analysed. For example, in recognising lanes, the standard steps are done with two-step pipelines in which the lane markings mask will be predicted and segmented first. Then, a lane like model such as parabola or a spline is applied to the post-processed mask. According to [11] this method is widely used for segmenting the lane markings. Some image processing methods include filtering, edge detection, and ground plane extraction.

Edge detection can be defined as identifying sharp changes in intensity in any adjacent pixels. This technique is widely used in image processing where it can detect the boundary of an image which works by observing the discontinuity in the difference of brightness. Edge detection can be used for segmentation of an image and extracting data on the region of interests in the area such as computer vision, image processing and also machine vision. The edge detection method works when the input takes an image as input that may contain multiple channels, such as an RGB. The task is to label each pixel with a binary variable indicating whether the pixel contains an edge or not. The detection of edges is a critical pre-processing step for a variety of tasks, including object recognition [12].

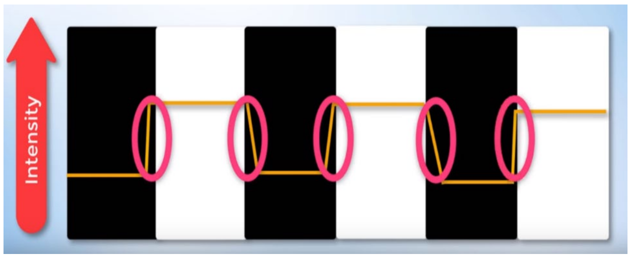

Figure 1: Differences in the intensity of pixels

Figure 1 shows an image pixel that contains different intensity based on the colour attributes such as red, green, and blue (RGB). The differences in intensity value can be detected when the colour pixel changes from low-intensity colour to high-intensity colour.

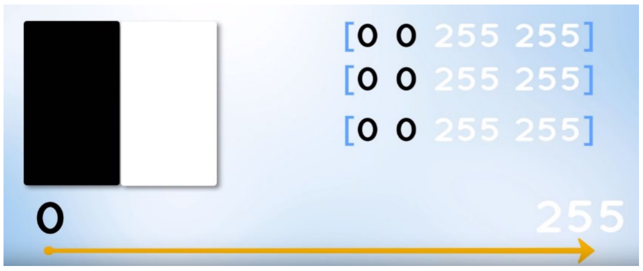

Figure 2: Matrix of an image

The image can also be read as a matrix which is an array of pixels. Figure 2 shows that each of the pixel intensity denoted by a numeric value that ranges from 0 to 255. As the number suggested, an intensity with value 0 indicates no intensity while pixel with an intensity value of 255 indicates the highest intensity.

2.3 Hough Transform Algorithm

Hough Transform (HT) method was invented in 1972 that efficiently identifies lines in images. This algorithm is a standard method for road detection that focuses on straight-line road detection. There are many variants of Hough transform with different computational complexity levels and processing time. Hough Transform technique is capable of isolating features of a specific shape in an image. According to [13], most detection of regular curves such as lines, circles and ellipses is done by using Hough Transform that requires the desired features be specified in some parametric form. Hough algorithm can also be applied using MATLAB, OpenCV, Python implementation using built-in functions in MATLAB [14]. Some examples are :

Hough : Apply the Hough transform on a binary image data, and then it will return the accumulator.

Houghpeaks : Detects lines by interpreting the accumulator and the threshold of the accumulator is set to a certain percentage to specify the maximum number of peaks.

Houghlines : Detect straight lines by converting infinite lines to finite lines.

Before applying the algorithm, this process requires the edge detection pre-processing method. There are many methods for edge detection, for example, Sobel, Prewitt, Log (Laplacian of Gaussian), and Canny methods that work by analysing the data intensity [13]. Hough Transform is capable of detecting shape in binary images by using the parameter space array [7], as shown in Figure 3. In many studies, the Hough Transform is the algorithm that was widely used since it uses edge pixel images for lane detection [15]. Every point in the binary images is voted as the parameters space. Along with the described methods above, this study continues to develop the lane detection model with the selected image pre-processing methods.

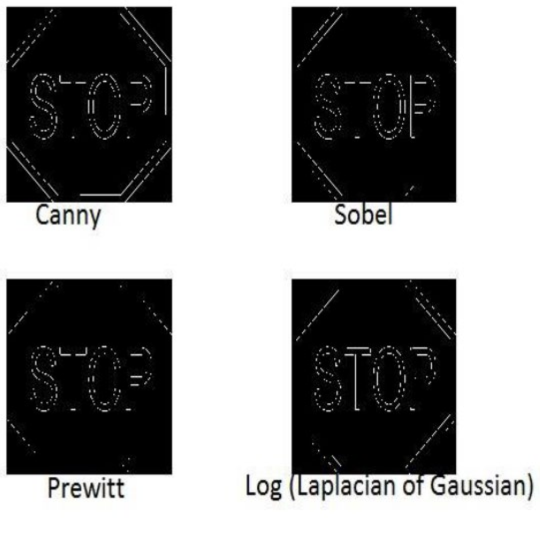

Figure 3: Example of edge detection methods image Sources [13]

3. METHODOLOGY

3.1 Data Pre-Processing

LiDAR data and camera data were pre-processed in two different ways. LiDAR data was processed using MATLAB. However, the data that was collected using LiDAR sensor is in the form of .PCAP which is a packet data and cannot be used in MATLAB. The data need to be converted into .MAT file to be processed for segmentation. The LiDAR data will be segmented and comprised of the following segments: ground points, ego points, obstacle points, and unlabelled points.

By segmentating the images, it can facilitate the drivable path planning for vehicle navigation. The ground plane was segmented and also t any nearby obstacles. The data was processed using fast indexing and search which is the key to the performance and later were organised using a KD tree data structure. It is a space portioning for organising points in a k-dimensional space.

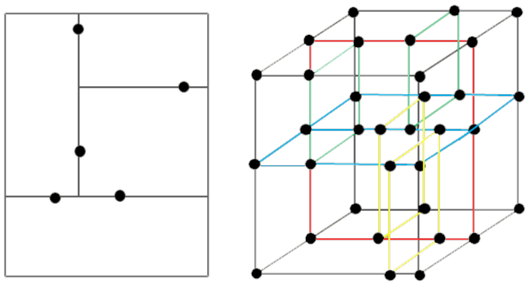

Figure 4: 2D (left) and 3D (right) of KD tree algorithms

It is a binary search tree with other constraints imposed on it. KD trees are beneficial for a range of and nearest neighbour searches. Figure 4 illustrates the differences of 2D and 3D of KD tree algorithms. KD trees are a special case of binary space partitioning trees.

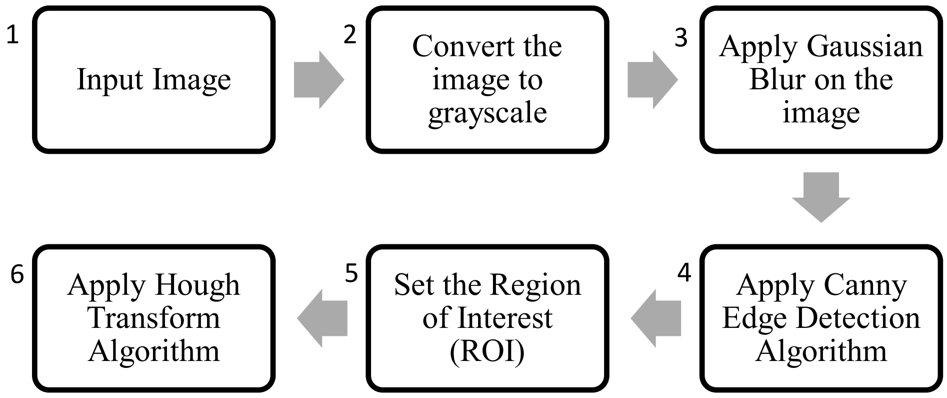
1
Input Image
2
Convert the image to grayscale
3
Apply Gaussian Blur on the image
4
Apply Canny Edge Detection Algorithm
5
Set the Region of Interest (ROI)
6
Apply Hough Transform Algorithm

Figure 5 Camera data pre-processing steps

The first step in pre-processing the camera data is filtering the data to obtain precise data such as the lane. Based on Figure 5 it shows the process that the camera data will undergo in order to develop the lane detection algorithm. In filtering, structure and obstacle data will be filtered because it is unnecessary to be used in this study, and it will cost the processing time due to the large data. First, the image will be converted into grayscale in order to recognise the intensity. This is because images are a mix of pixels with 3-channel colour images which are of red, green, and blue (RGB). Each of the pixels is a combination of the three intensity values.

Meanwhile, a grayscale image only possesses one channel where each of the pixel has only one intensity value that ranges from 0 until 255. The use of the grayscale makes the processing faster and requires less computational power since it only has one channel compared to colour image with three channels. The intensity of the image can be determined by the sharp changes in the colour and by converting it to grayscale. This process is needed to find the gradient of the image. A gradient is a measure of the change in brightness over adjacent pixels. Based on Figure 6, it is shown that a strong gradient indicates an abrupt change, while a small gradient represents slight or undetectable changes. By using this feature, it can differentiate structures in the environment.

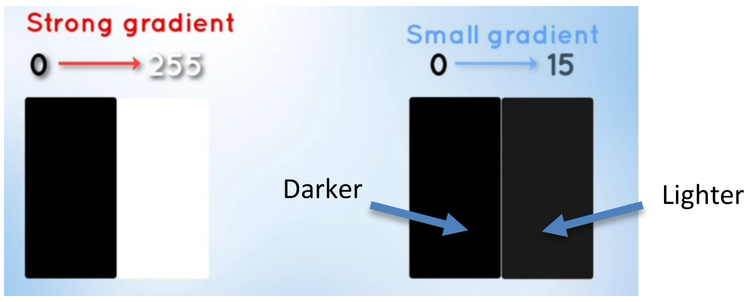

Figure 6: Differences in the gradient of a pixel

The next process is the rasterisation of the image in order to increase the intensity of the data. The Gaussian filter was applied to the image to rasterise the image. This process is done to improve the identification of the intensity data more efficiently. This process will reduce the noise in the image and smoothen the image. Image noise can possess risk in creating false edges affecting the edge detection, which is the reason why it is needed to be filtered out. The Gaussian filter works by modifying the value of a pixel with the average value of the pixel intensities around the image. Figure 7 illustrates how the Gaussian filter works by changing the pixel. The pixel is modified by averaging out the value in the pixel with the surrounding pixels using a kernel of distributed numbers which is run across the entire image. The kernel will set the pixel value equal to the weighed average of its neighbouring pixels which will smoothen the image. The kernel size is 5x5 kernel which is the right size for most cases that will return a new image.

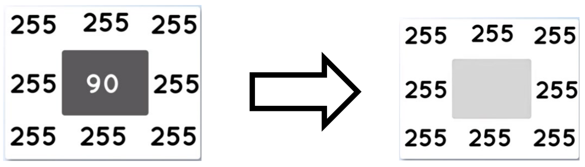

Figure 7: Illustration of how Gaussian filter works

After the image has been smoothened, Canny Edge Detection was applied to identify edges in the image. An edge can be detected when there is a sharp change in intensity or sharp change in colour between the pixels in the image. As mentioned earlier, the changes in brightness over a series of pixels is the gradient. It can also be represented in a two-dimensional coordinate space which is X and Y as shown in Figure 8. The X-axis traverses the width of the image while the Y-axis traverses the height of the image such that the width represents the number of columns in the image and height represents the number of rows in the image respectively. The product of both the X and Y value gives the total number of pixels in the image. Figure 9 shows that the image can be represented in matrix and continuous function of X and Y, meaning that it can process mathematical operations needed for experimenting with the pixel values.

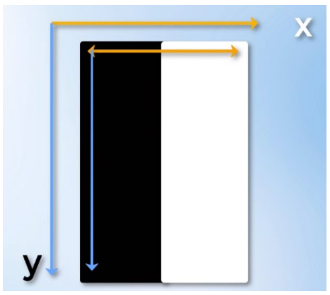

Figure 8: Representation of pixels in 2-D coordinate space

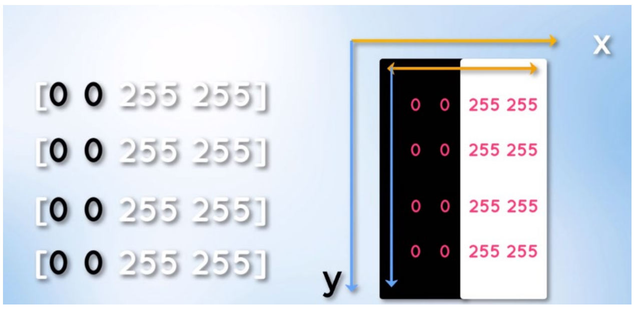

Figure 9: Representation of the image in the matrix and continuous function

The next step is the process of setting the region of interest (ROI) of the image. The region of interest needs to be specified because it will be used to detect the lines. It will enclose the field of view based on the width and height of the image coordinate. This process is extracting the features such as road and line-markings which is the input for lane detection algorithm. The data is converted to be used in the mapping of the environment to get a better view of the data obtained. The last step is the implementation of the Hough Transform algorithm. This algorithm will identify possible lines from a series of points on the 2-D coordinate space. It can also be represented on the hough space, and the algorithm will choose the pixel where the most point intercepts.

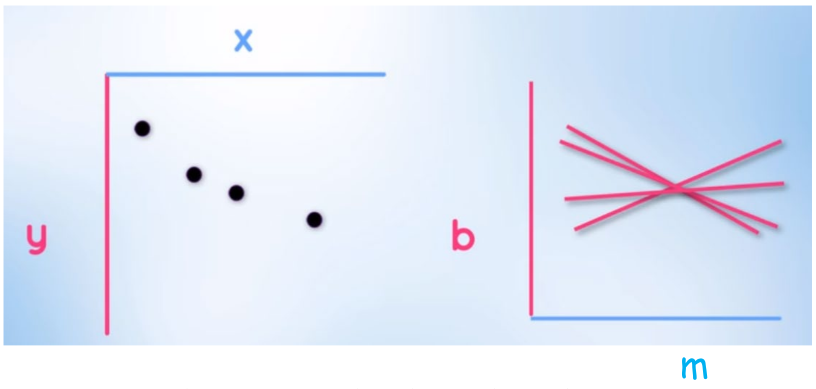

Figure 10: Cartesian Plane and Hough Space

Based on Figure 10, the four points in the cartesian plane corresponds to the hough spaceline. The hough space is represented by b as the y-axis and m as the x-axis. Each point can form a number of lines in the hough space. The hough space is split into the grid, and each bin inside the grid corresponds to the slope and y-intercept value of a possible candidate line.

3.2 Model Development

This section explains the step of developing the model in two different software platforms, namely MATLAB and Atom using Python. MATLAB is a programming platform with high-performance language that allows for the integration of algorithm and manipulation of data. Atom is an open-source text and source code editor that can run and compile codes in Python language.

The point cloud data is segmented using the MATLAB platform. The MATLAB version is 2019a which is currently the second latest version of MATLAB. First, the Velodyne reader is created using the command below:

```
veloReader = velodyneFileReader(fileName, deviceModel);
```

After that, it will be able to read the point cloud data and can be displayed. To display the data, another command is needed because it will not automatically display the output data.

```
lidarViewer = pcplayer(xlimits, ylimits, zlimits);
```

The command above is used to display the point cloud data with specific 3D-coordinate. Then, we will set the display to four different segments, namely the unlabelled data, ground plane, ego points, and obstacle. The command for that is as follows:

```
colormap(lidarViewer.Axes, colorLabels);
```

Lastly, the segmentation process is done by using the input that has been set and segmented into four different segments. The data will be displayed in a video form after the segmentation process is performed.

The data of the camera is processed and filtered using Python. The image is converted to grayscale to reduce the complexity of the pixel. The code is written on Atom platform with Python for the lane detection algorithm. The command segments for image conversion is as follows:

```
gray = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
```

Then, it was applied with Gaussian blur in order to smoothen the image and reduce noise of the pixels. This processed is done before edge detection to avoid false edge detected after applying the edge detection. Below is the command for applying Gaussian Blur filter:

```
blur =cv2.GaussianBlur(gray,(5, 5), 0)
```

After that, the edge detection will be applied to the data to detect every edge in the data. The detection of the edge is needed so that it is able to find the edge of the road lines. The Canny Edge Detection is used and the command segment is shown below:

```
canny = cv2.Canny(blur, 50, 150)
```

The data displays an image with every edge detected in the pixels using the intensity of the pixels. After that, the lane detection will be tested with a different region of interest to find the optimal region of interest that can get the best detection.

4. RESULTS AND DISCUSSION

This section shows the result of the point cloud data before and after being segmented using MATLAB. The data was segmented into four different segments. The LiDAR data was segmented based on four labels which are unlabelled points, ground points, ego points, and obstacle points. Each of the colours denoted different segments. Yellow stands for the ego point of the vehicle. Ego stands for the points that belonged to the vehicle itself to segment the source. The ego point can be detected by observing how the LiDAR is mounted at the location specified in the vehicle coordinate system. Then, it used the XYZ location to specify the direction that the vehicle is moving in order to get the buffer around the ego vehicle. Figure 11 shows the unfiltered data gathered while figure 12 shows the filtered data with the four labels.

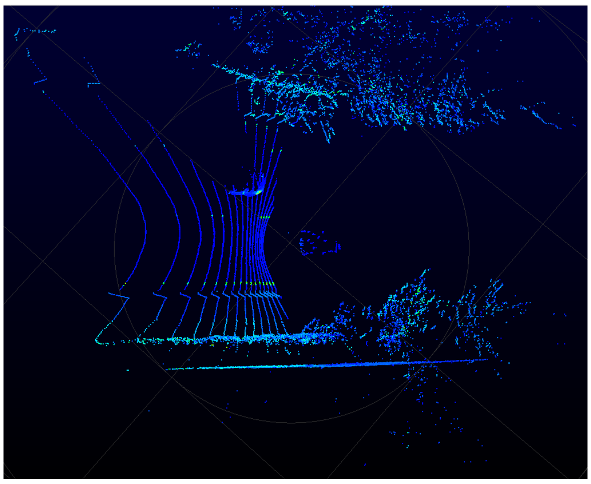

Figure 11: Unfiltered Data

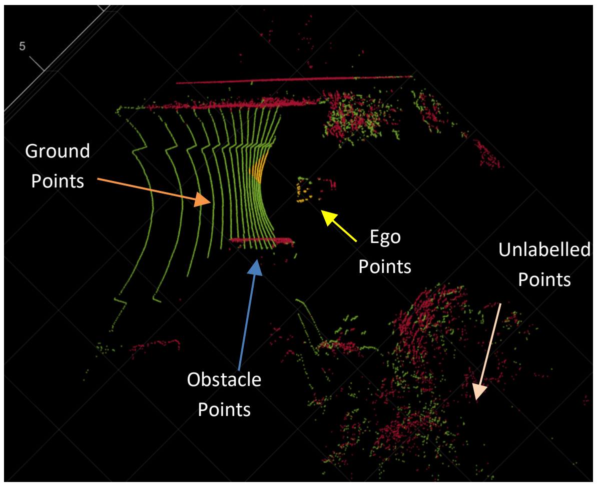

Figure 12: Filtered Data

The lane detection is successfully done without any false detection due to the optimal region of interest value that has been set. This proves that the region of interest is highly essential when conducting the experiment and testing the algorithm to get the algorithm fully functional. In order to ensure that the lane detection can be working correctly, an accuracy test is done to see the accuracy rate. Figure 13 displays the lane marking using reflective intensity while figure 14 shows the lane detected using the lane detection algorithm.

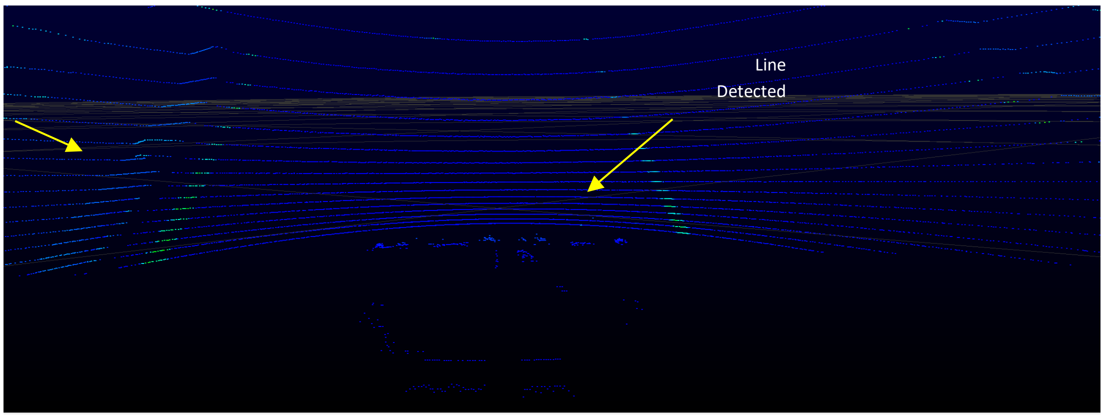

Figure 13: Point cloud data with reflective intensity

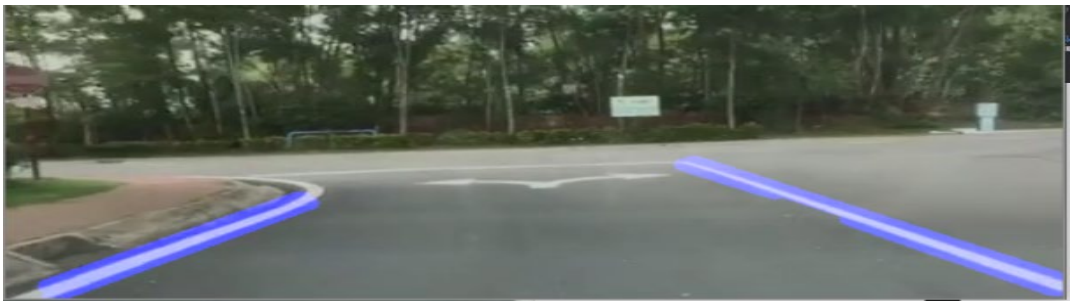
Figure 14: Data with lane detection

Equation (1) shows the calculation of the accuracy taken from [16]. Let P be the number of actual lines detected using point cloud data and Q be the differences between the detected line and false detected line from the lane detection algorithm. P is the line taken from the point cloud data, as shown in Figure 13, while Q is the detected line, as shown in Figure 14.

$$Accuracy\ rate = 100\% - \left(\frac{|P-Q|}{P}\ x\ 100\%\right) \qquad (1)$$

Table 1 is a tabulation of the accuracy of the algorithm. The accuracy is calculated using the results collected in different experiments done at different locations. Experiment 3 is the main experiments, and the accuracy is calculated based on ten frames with a rate of 3 frames per second. The accuracy rate shows that the results are acceptable even though there are some situations where the algorithm made false detection. Experiment 2 achieved perfect accuracy because the data in experiment 2 is low and not complicated. In contrast, experiment 3 shows the lowest accuracy because the data is quite complicated, and the algorithm is unable to detect all of it. Overall, the performance of the model is satisfactory with an average of 72.4% over five sets of experiments.

Table 1: Experiments Results

Experiment	Actual line (P)	Detected (Q)	|P-Q|	Accuracy (%)
1	2	1	1	50.00
2	2	2	0	100.00
3	13	10	3	76.92
3	18	12	6	66.67
3	19	13	6	68.42
			Average	72.40

5. CONCLUSION AND FUTURE WORK

In this paper, we presented the pre-processing method and model development for vehicle lane detection. In the pre-processing method, the detailed steps of pre-processing the LiDAR and camera images were explained. Methods of Canny Edge and Hough Transform algorithms for vehicle lane detection were presented. For the model development, the steps and the functions in Matlab and Python were shown and explained. The entire process of developing and detecting the lane was found to be satisfactory and the algorithm was able to reach the goal point successfully. As for future work, an improvement in processing the point cloud can be done so that it can generate the lane detection itself that can further be extended to lane-level map generation. Additionally, enhancing the algorithm for image processing to be able to read a vast amount of data in a short amount of time.

ACKNOWLEDGEMENT

The authors would like to thank the Research Management Centre, Universiti Teknologi MARA, Selangor, Malaysia for FRGS 600-IRMI/FRGS 5/3 (461/2019) and the Faculty of Computer and Mathematical Sciences for all the given support.

REFERENCES

[1] A. Saxena, "Top 7 ADAS technologies that improve vehicle safety," 11 April 2018. [Online]. Available: https://www.einfochips.com/blog/top-7-adas-technologies-that-improve-vehicle-safety/.

[2] J. Thompson and G. Read, "Nothing to fear? How humans (and other intelligent animals) might ruin the autonomous vehicle utopia," 29 April 2019. [Online]. Available: https://theconversation.com/nothing-to-fear-how-humans-and-other-intelligent-animals-might-ruin-the-autonomous-vehicle-utopia-114504.

[3] V. M. Lane, Obstacle Detection and Tracking in an Urban Environment Using 3D LiDAR and a Mobileye 560, Massachusetts Institute of Technology, 2017.

[4] A. M Kumar and P. Simon, "Review of lane detection and tracking algorithms in advanced driver assistance system," *International Journal of Computer Science and Information Technology,* vol. 7, no. 4, pp. 65-78, 2015.

[5] BBC, "BBC NEWS: Tesla Model 3: Autopilot engaged during fatal crash," 17 MAY 2019. [Online]. Available: https://www.bbc.com/news/technology-48308852.

[6] Y. F. Wang and Y.-S. Tsai, "A lane detection method based on 3D-LiDAR," 2018.

[7] W. Liu, Z. Zhang, S. Li and D. Tao, "Road detection by using a generalized hough transform," *Remote Sensing,* vol. 9, no. 6, 2017.

[8] M. Dorazio, "A Difficult Drive: Autonomous Vehicle Sensor Limitations," 07 August 2018. [Online]. Available: https://www.concannonbc.com/a-difficult-drive-autonomous-vehicle-sensor-limitations/.

[9] B. Sharma, "What is LiDAR technology and how does it work?," 30 January 2019. [Online]. Available: https://www.geospatialworld.net/blogs/what-is-lidar-technology-and-how-does-it-work/.

[10] B. schwarz, "Mapping the world in 3-D," *Proceedings, IEEE Aerospace Conference,* vol. 1, no. July, pp. 3-4, 2010.

[11] B. De Brabandere, W. Van Gansbeke, D. Neven, M. Proesmans and L. Van Gool, "End-to-end lane detection through differentiable least-squares fitting," *IEEE Transactions on Intelligent Transportation Systems,* vol. 17, no. 2, pp. 420-429, 2019.

[12] V. Ferrari, L. Fevrier, F. Jurie and C. Schmid, "Groups of adjacent contour segments for object detection," *IEEE transactions on pattern analysis and machine intelligence, 30(1),* vol. 17, no. 2, pp. 36-51, 2008.

[13] P. Amaradi, N. Sriramoju, D. Li, G. S. Tewolde and J. Kwon, "Lane following and obstacle detection techniques in autonomous driving vehicles," *IEEE International Conference on Electro Information Technology,* vol. 2016, no. August, pp. 674-679, 2016.

[14] A. University, "Line Detection by Hough transformation," *Transformation,* pp. 2-8, 2009.

[15] Y. Xing, C. Lv, L. Chen, H. Wang, H. Wang, D. Cao, E. Velenis and F. Y. Wang, "Advances in Vision-Based Lane Detection: Algorithms, Integration, Assessment, and Perspectives on ACP-Based Parallel Vision," *IEEE/CAA Journal of Automatica Sinica,* pp. 645-661, 2018.

[16] C. Deziel, "How to Calculate Percent Accuracy," 13 March 2018. [Online]. Available: https://sciencing.com/calculate-percent-accuracy-6199228.html.

p-ISSN 1675-7939; e-ISSN 2289-4934