Lane Keeping Algorithm for Off-The-Shelf Wide-Angle Camera

Kyaw KoKo HTET, Kok Kiong TAN
(*Department of Electrical and Computer Engineering, National University of Singapore, Singapore* 117583)

Abstract: Automobile accidents cost over a trillion-dollar every year and this figure will continue increasing without employing new technological solutions. Among these solutions, the automated lane-keeping system is one of the promising ones and such a system consists of two essential technologies: road detection and steering control. In this paper, novel lane keeping algorithms are proposed and are implemented using only a single off-the-shelf wide-angle camera as input. The implemented system is verified, through both simulation and experiments, and is found providing satisfactory performance for an automated lane-keeping system. When compared to the state-of-the-art lane-keeping systems, the implemented system can perform consistently across various ambient light conditions including the most challenging ones.

Key words: Automotive accidents, Autonomous driving, Road detection, Steering control, Lane detection, Lane keeping, Sensing with mono camera.

1 Introduction

The ever-increasing car ownership in the whole world leads to a rapid increment of traffic accidents. As a result, every year over a trillion-dollar is spent on medical cost, property damage, and other costs associated with automotive accidents [1]. Many of these accidents are triggered by drivers' inattention and carelessness, resulting in roadway departure crashes. Every year, over one million people around the world are killed in such road crashes. This figure is projected to increase by approximately 65% over the next 20 years in the absence of new commitments to preventive technology [2]. Among the potential solutions, the automated lane-keeping system is a promising one. In such a system, automated road detection and automated steering control are essential. [3]

Over the last two decades, manypieces of research have focused on road analysis and lane detection. In the lane detection field, numerous research papers can provide excellent performance in terms of application requirements. Some examples are Multiple-hyperbola-road model [4], open uniform B-spline curve model [5], and a K-means cluster algorithm[6]. They are often implemented alongside with tracking algorithms such as Kalman filter and particle filter. Due to these efforts, many lane-detection systems have shown satisfactory performance in many challenging road conditions. However, these algorithms generally demand high computation cost due to the use of genetic algorithms or particle filters as well as the derivation of the whole lane's structure. They often required high-performance cameras or stereo cameras. For the steering algorithms, the use of the lane's structure information leads to complexity as well as higher computation cost. Moreover, even though lane detection and lane-keeping algorithms are generally proposed independently in literature, the processing of these lane algorithms in an actual system takes substantial processing time thereby, introducing significant latency in the steering action and causing a deterioration in performance and stability upon integrating of the twos.

This paper proposes an automatic lane-keeping algorithm integrated together with a lane detection algorithm. The lane-keeping system utilizes an off-the-shelf wide-angle camera as an input and uses the proposed algorithm to estimate the position of a lane

center referenced to the camera position instead of the whole lane. After that, the vehicle's travel distance and direction during the processing are computed and the estimated lane position is adjusted accordingly. And then, the new estimated position is used to compute the required steering angle for maintaining the vehicle at the center of the lane through the proposed steering algorithm. The algorithm consists of only four simple equations and thus, the complexity and the computation cost are kept as low as possible while providing satisfactory road keeping performance.

The paper first provides an overview of the proposed lane-keeping system, followed by its detailed implementation. After that, the performance of the implemented system is investigated through simulations and the results are presented to observe the effectiveness of the proposed system. The system is then physically implemented in an electric vehicle (EV) utilizing only an off-the-shelf webcam as the input. The implemented system's performance is evaluated through a set of driving experiments under both day and night road conditions.

2 Lane Keeping System

The lane-keeping system contains three main subsystems: a lane detection subsystem, a Latency-Error-Correction (LEC) subsystem and an automated steering control subsystem. The lane detection uses images captured from an off-the-shelf wide-angle camera as an input. The captured RGB images are first transformed into grey-scaled images. The transformed images are then further processed to remove effects from the ambient-light condition by using a local-area-intensity-transformation (LAIT) algorithm. The algorithm is discussed details in section III-A. After that, the processed images are transformed into two-dimensional planar images which are based on the world coordinate system by applying the inverse perspective mapping (IPM) algorithm [7]. The resultant images are thresholded into black and white (BW) images, marking detected lane lines in white and the rest in black. From these BW images, positions of the lane lines in each image are estimated using the Hough line detection algorithm [8]. These resulting Hough lines are grouped to cluster together pixels belonging to same lane lines. These grouped pixels are chunked together and plotted on a histogram. Then, the histogram is normalized resulting in a probability density function of the lane positions. From the function, the maximum likelihood of lane position is estimated, and the reliability of the estimation is computed as well.

Even though the above algorithm can estimate the lane position, the time, taken between the image capturing and the position estimation, can be significant especially with a slow processor. During this period, the vehicle can travel a few meters and thus, the current actual position can be a few meters away from the estimated one. To address the issue, an LEC, which can compute the current lane position from the estimated one, is designed and proposed in the present paper.

The calculated position is fed into the steering control subsystem. The steering control uses the position information and calculates an appropriate steering angle that will drive a vehicle toward and maintain at the center of the lane. In the following sections, the mentioned subsystems are discussed in detail.

3 Lane Detection System

3.1 Preprocessing

First, the images from the onboard camera are transformed into greyscale ($I_{grey} = 0.299R + 0.587G + 0.114B$) to lessen computational cost. A sample of the resultant images (I_{grey}) is shown in Fig. 1. In I_{grey} , the lighting condition directly influences its pixel-intensity value. Especially in the night, the lighting condition is unevenly distributed as shown in Fig. 1. Likewise, during the day, the light intensity of the shaded area is significantly lower than that under direct sunlight. Hence, minimizing this influence is crucial to achieving a consistent lane-detection

performance under various lighting conditions.

Fig. 1 A sample night road view.

To minimize the influence of the lighting condition, the LAIT algorithm is proposed and applied. As lane markings are painted on roads which have darker colors, the algorithm indirectly improves the performance of detecting lane markings. The LAIT involves two stages. The first one is computing the light intensity of the local area by using a low pass filter (LPF), as is given by,

$$H_{LPF} = \frac{J_{n,m}}{nm} \tag{1}$$

where H_{LPF} is the LPF filter, $J_{n,m}$ is an all-ones matrix with the dimension (n x m), and n and m are the number of row and column of the filter respectively. The resultant image, I_{LPF}, is shown in Fig. 2b. The second stage is to derive the relative intensity with respect to the background by applying,

$$I_{RI} = I_{grey} - I_{LPF} \tag{2}$$

where I_{RI} is the image with relative intensity and the resulting image is shown in Fig. 2c. The processing time of the LAIT algorithm can be reduced by selective processing of only the area of concern from the image, as shown in Fig. 2a.

3.2 Inverse Perspective Mapping

As pixels are equally distributed across viewing angles in both horizontal and vertical planes, the vertical viewing angle, θ, relative to the horizontal line can be translated based on each pixel's y coordinate by using,

$$\theta = \frac{y \times \theta_{max}}{N_{row}} + \theta_{offset} \tag{3}$$

where θ_{max} is the maximum vertical viewing angle, Nrow is the number of pixel rows, and θ_{offset} is the angle between the most upper viewing angle and the horizontal line as shown in Fig. 3b. The real-world distance, y, in the Y direction from the camera can be calculated by using,

$$y = \frac{h}{tan\ (\theta)} \tag{4}$$

where h is the height of the camera respected from the ground.

Similarly, the horizontal viewing angle, α, relative to the horizontal center line, A, can be translated based on each pixel's x coordinate by using,

$$\propto = \frac{x\text{-}0.5 \times x_{max}}{x_{max}} \propto_{max} \tag{5}$$

The real-world distance, x, in X direction from the camera can be calculated by using,

$$x = ytan\ (\propto) \tag{6}$$

Using real-world coordinate information (x, y) and remapping pixels will result in the bird's eye view of the image as shown in Fig. 2d and thus, removing the perspective effect [9-11,7].

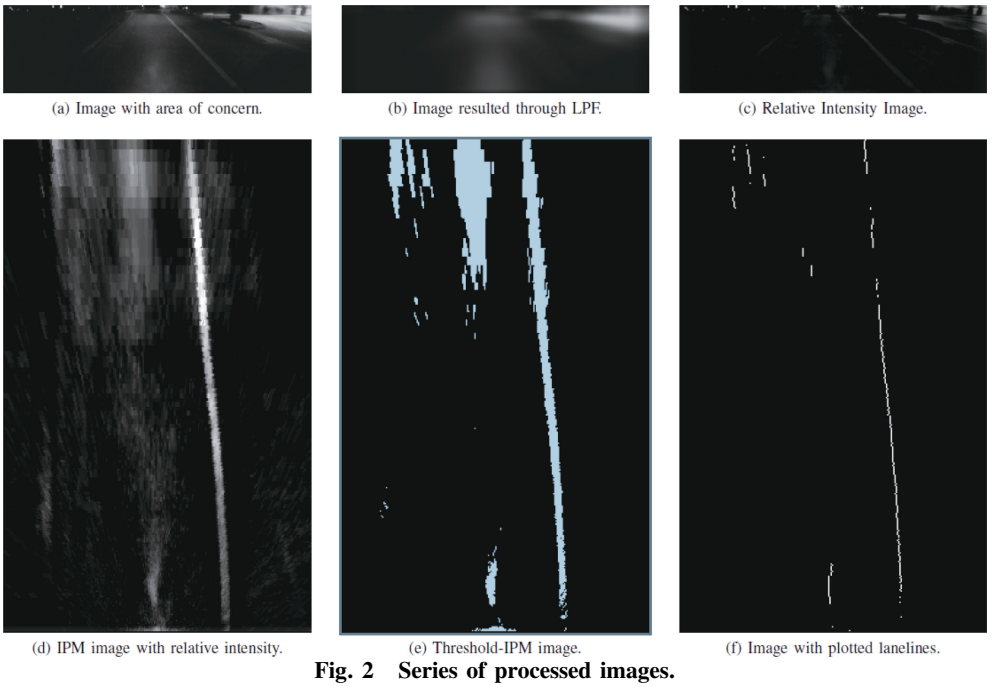
3.3 Plotting Lane Lines

The IPM image is performed thresholding to identify potential lane-markings' areas. In this stage, these potential areas are represented in white while the rest are represented in black; resulting in a black and white (BW) image, as shown in Fig. 2e.

By using a horizontal and vertical scanning method, the center points of the lane lines are mapped. During the scanning, the widths of the white lines are measured and those, outside a predefined lane-width-range, are discarded. And then, noises in the template are further reduced by omitting unconnected pixels. The result is shown in Fig. 2f.

3.4 Computing Lane Center Point in a Real-World Distance

In this step, straight lines in the template are detected using Hough transform[12]. The resulting Hough lines are grouped to cluster together pixels

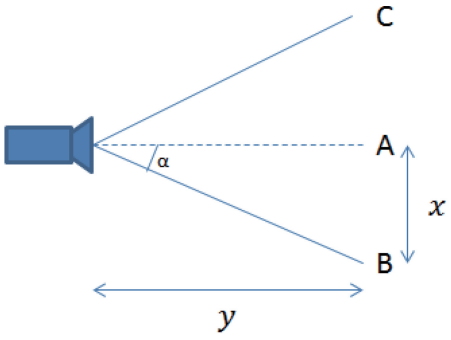


(a) Image with area of concern. (b) Image resulted through LPF. (c) Relative Intensity Image.

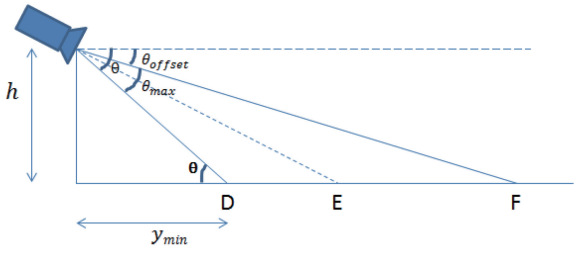
(d) IPM image with relative intensity. (e) Threshold-IPM image. (f) Image with plotted lanelines.

Fig. 2 Series of processed images.

belonging to same lane lines. In Hough transform, lines are grouped based on their distance from the center and their angle, creating the matrix image as shown in Fig. 4.



(a) Horizontal



(b) Vertical

Fig. 3 Cross-section of the viewing angle.

From the matrix, lines are again grouped based on their pointed direction towards lane positions on

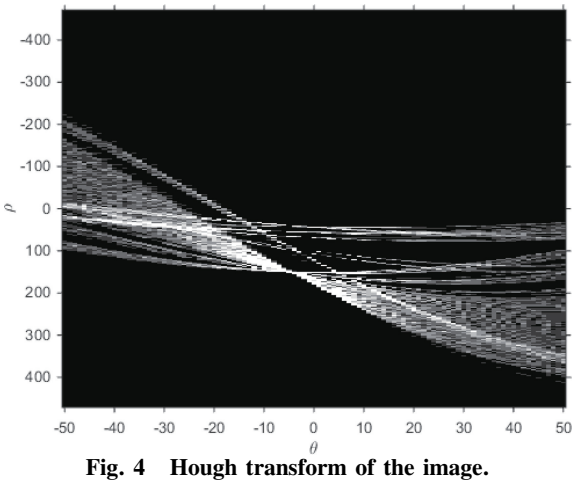


Fig. 4 Hough transform of the image.

an x-plane line (shown in Fig. 5a), creating the histogram, which is normalized to form the Probability plot shown in Fig. 5b. The mapping relationship between the Hough transform and the histogram, which is shown in Fig. 6, can be derived using (7). During the mapping process, the pixels with insignificant magnitudes from the Hough transform are ignored to avoid noise accumulations.

$$x = \frac{\gamma}{\sin\beta} - \frac{y}{\tan\beta} \tag{7}$$

With respect tothe single-side lane line, the left half of the probability plot is merged with the right

half by using,

$$P_{merge}(x) = P(x) + P(x + \omega) \quad (8)$$

where $x = \{1,2,\cdots x_{max},w\}$, w is the width of the lane, $P(x)$ and $P_{merge}(x)$ are the probability before and after merging respectively. The resulting histogram is shown in Fig. (7). In order to remove sampling effects, the histogram is filtered by using a low pass filter and the resulting one is shown in Fig. (8).

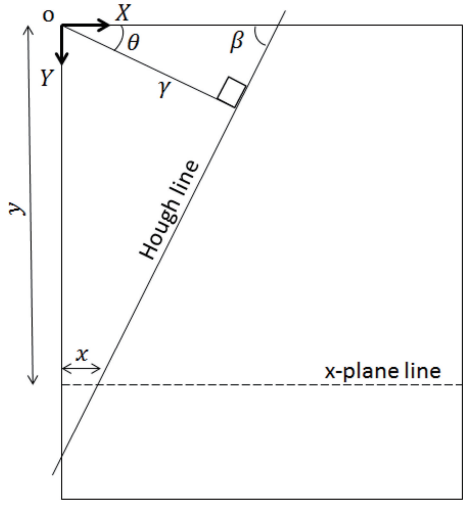


Fig. 6 Mapping the relationship between Hough transform and histogram.

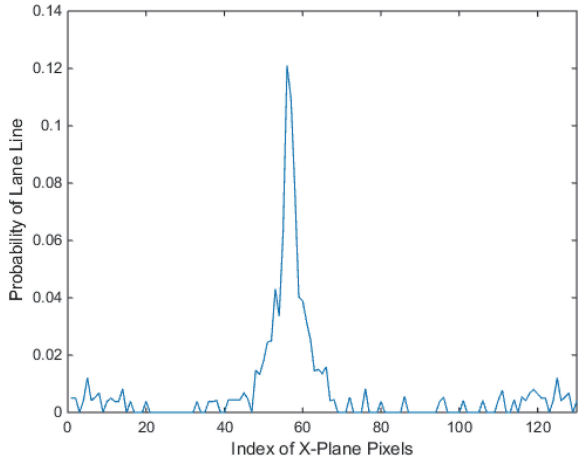


Fig. 7 Merged histogram.

From the histogram, the maximum likelihood of lane's positionsis identified and then, the center of the lane is calculated by using,

$$x_{LC} = x_{LLP} + \frac{\omega}{2} \quad (9)$$

where x_{LC} is the lane center position, x_{LLP} is the left lane position, and w is the width of the lane. The lane center position is plotted on an IPM image as shown in Fig. 9.

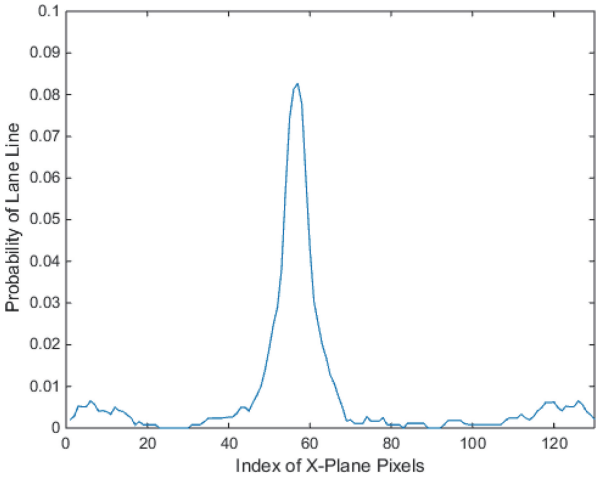


Fig. 8 Histogram after low-pass filter.

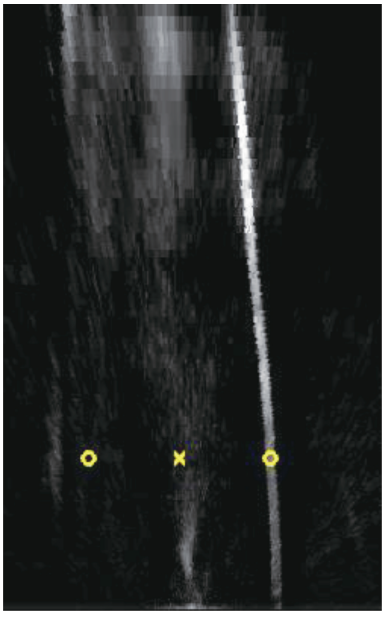


Fig. 9 IPM image with lane center marked with 'x' and its sides marked with 'o'.

The reliability (Γ) of the derived lane center position is analyzed by means of standard deviation. First, the standard deviation (σ) of the Histogram (Fig. 8) is computed. The value of Γ should be zero when the histogram is uniformly distributed or σ becomes 68.2% of the histogram's width (w_H). The

value of R should be one when the histogram is concentrated on a single point or σ becomes zero. Driven by the above correlation, the relationship can be derived as

$$\Gamma = 1 - \frac{\sigma}{0.682\, w_H} \tag{10}$$

The derived position is fused with information from simultaneous localization and mapping (SLAM)[13] in which the weight of the derived value is directly proportional to Γ. In this way, the position of the lane at the time of image capturing can be estimated. However, due to the computation time, the actual current lane position can be significantly different from that of the estimated position. This phenomenon is termed as processing latency. The next section proposes an algorithm that can predict the lane's position more accurately.

4 Latency Error Correction Algorithm

After the image has been captured, the lane detection system takes time to process the image to find the center of the lane. During this process, the vehicle typically travels some distance and sometimes even changes its orientation by a few degrees. For example, when the lane detection system takes 250 ms to process while traveling at 10km/hr, the vehicle can travel 0.7m during the processing and thus, the estimated lane center position is 0.7m away from the actual position. Moreover, during the processing, its orientation can be shifted up to a few degrees as well. To derive a more actuate lane center position, the changes in the vehicle travel distance and orientation must be considered by using a LEC algorithm.

In this paper, the integration of vehicle velocity is used to estimate the total distance (s) travelled. During the image processing, the vehicle has moved a distance ($\vec{d}$) and its orientation is changed through an angle (α), as well as the reference position is shifted from position B to A as shown in Fig. 10.

The travel distance ($\vec{d}$) during the image processing is assumed mainly in YB direction and thus

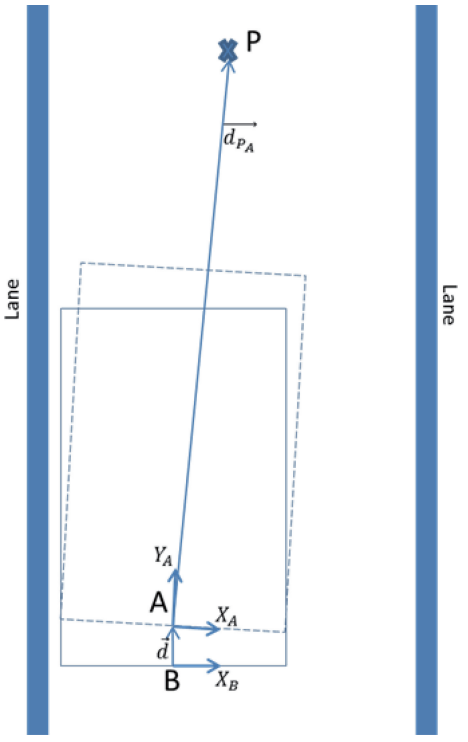


Fig. 10 Vehicle movement during image processing.

the $\vec{d}$ can be expressed as,

$$\vec{d} = \begin{vmatrix} d_{xB} \\ d_{yB} \end{vmatrix} = \begin{vmatrix} 0 \\ s \end{vmatrix} \tag{11}$$

where d_{xB} and d_{yB} are the travel distance in X_B and Y_B direction, respectively, and s is the displacement. The orientation changes ($\propto$) can be derived as,

$$\propto = \frac{s}{L}\tan\delta \tag{12}$$

where L is the distance between the rear and front wheels and δ is the steering angle. Detailed derivation of $\propto$ is given in the next section. By using the movement distance ($\vec{d}$) and the rotation angle ($\propto$), the transformation matrix (${}^{A}_{B}T$) can be described as [14],

$${}^{A}_{B}T = \begin{vmatrix} \cos\propto & -\sin\propto & -s\sin\propto \\ \sin\propto & \cos\propto & -s\cos\propto \\ 0 & 0 & 1 \end{vmatrix} \tag{13}$$

The lane detection system can provide ($\overrightarrow{d_{P_B}}$) which is the lane center position with respect to the reference B, stated as,

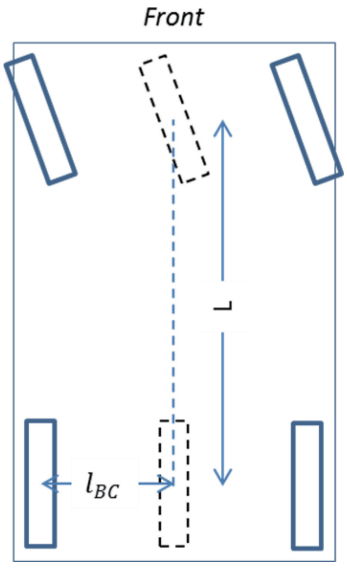


Fig. 11 EV's Bicycle Model

$$\overrightarrow{d_{P_B}} = \begin{vmatrix} d_{P_{xB}} \\ d_{P_{yB}} \end{vmatrix} \tag{14}$$

where $d_{P_{xB}}$ and $d_{P_{yB}}$ are the distance between Point B and the lane center in X_B and Y_B direction respectively. However, the current lane center position $\overrightarrow{d_{P_B}}$ is referenced from the new position A and thus, the transformation matrix is used to transform $\overrightarrow{d_{P_A}}$ from $\overrightarrow{d_{P_B}}$ as given by,

$$\begin{vmatrix} \overrightarrow{d_{P_A}} \\ 1 \end{vmatrix} = {}^{A}_{B}T \begin{vmatrix} \overrightarrow{d_{P_B}} \\ 1 \end{vmatrix} \tag{15}$$

The estimated current lane center position isfed into the steering algorithm, which is discussed in more detail in the next section.

5 Steering Algorithm

Steering algorithm is based on the bicycle model of an Ackermann steered vehicle[15]. According to this model, the rotation radius of the EV for corresponding steering angles can be estimated. In this model, the vehicle's trajectory is modeled as a bicycle model where its steering angle is parallel to the vehicle, its rear wheel exists exactly in the middle point between two rear wheels of the vehicle, and its front wheel exists exactly in the middle point between two front wheels as shown in Fig. 11. Its geometric model[15] is shown in Fig. 12. Based on this model, the geometric relationship between steering angle and rotation radius, R, can be written as,

$$R = \frac{L}{\tan\delta} \tag{16}$$

where R is the vehicle cycling radius, L is the distance between front and rear wheels, and δ is the steering angle. Coordinates and points from Fig. 13 are represented in a 2D plane and are shown in Fig. 14.

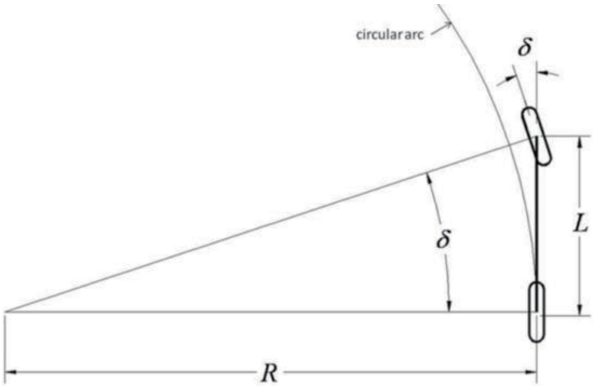


Fig. 12 Bicycle's Geometric Model

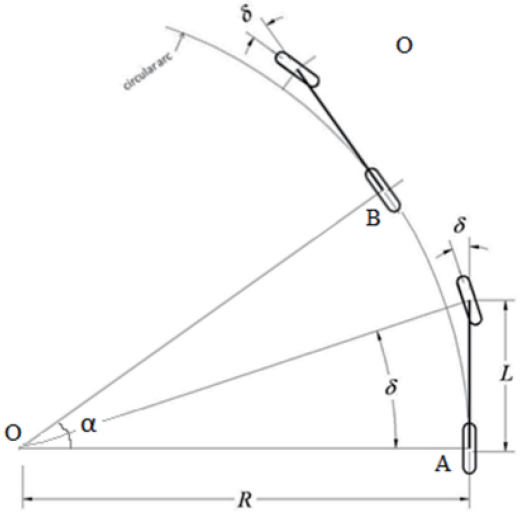


Fig. 13 Bicycle circling at a radius, R

Fig. 13 and 14 show that the rear wheel of the bicycle moves from point A ($\overrightarrow{P_A}$) to point B ($\overrightarrow{P_B}$) by applying a specific steering angle, δ. There are some steps involved in order to reverse compute the

required steering angle, δ, given $\overrightarrow{P_A}$ and $\overrightarrow{P_B}$. Firstly, the distance vector ($\vec{d}$) is derived by using,

$$\vec{d} = \overrightarrow{P_B} - \overrightarrow{P_A} \quad (17)$$

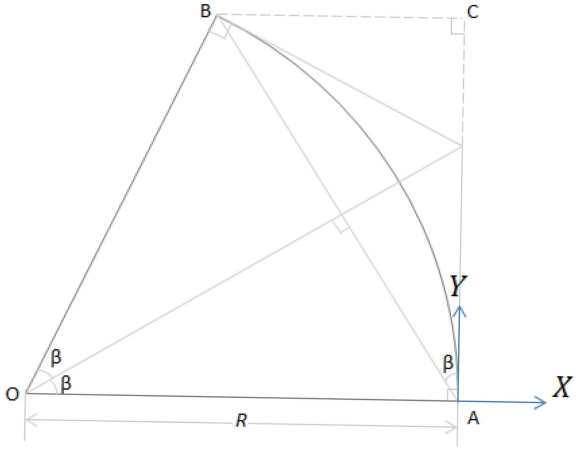


Fig. 14 Representation of coordinates and points in 2D plane

The angle (β) between d and X can be calculated by using,

$$\beta = \tan^{-1}\frac{d_x}{d_y} \quad (18)$$

where d_x and d_y are d vector in X direction and Y direction respectively.

After that, the required circling radius can be calculated by using,

$$R = \frac{d}{2\sin\beta} \quad (19)$$

By using the circling radius, the steering angle can be computed by using,

$$\delta = \tan^{-1}\frac{L}{R} \quad (20)$$

6 Simulation

A set of simulations has been carried to study the proposed steering algorithm's performance as well as the LEC algorithm's. The map of the simulated lane is shown in Fig. 15.

To emulate arealistic control scenario, 100 ms and 10 rad/s are chosen as the sampling period and the maximum turning speed of the steering motor respectively. Additionally, 100 ms of the computational latency is added in the simulation. A constant of 10 m/s or 36 km/hr is used as the vehicle's speed. The dimensions of the vehicle, mentioned in section VII, are used to model in this simulation. The vehicle's traveling path resulted from applying the proposed algorithms and the lane's center are compared. The error, the distance between the vehicle's travel path and the lane's center, is presented in Fig. 16. In this simulation, the root-mean-square (RMS) error of 0.3875 m is achieved.

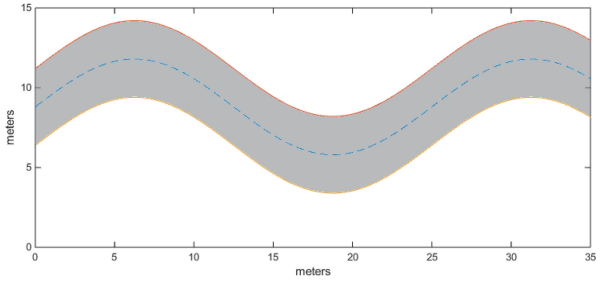


Fig. 15 Map ofthe simulated lane.

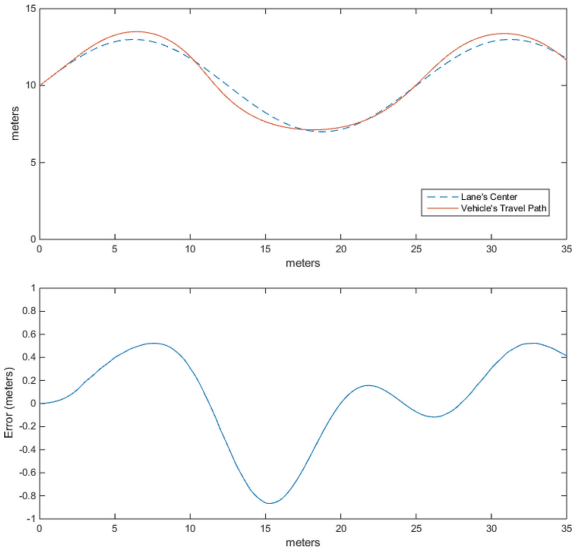


Fig. 16 Actual traveling path of a vehicle and resulting error in lane centering with computational latency.

Based on the above simulation, the lane- keeping error contains oscillatory components, which result from the latency that is present. To reduce the oscillation, the LEC algorithm is deployed and the result is shown in Fig. 17. By applying the LEC, the RMS error can be reduced from 0.3875m to 0.0620m

which is significant.

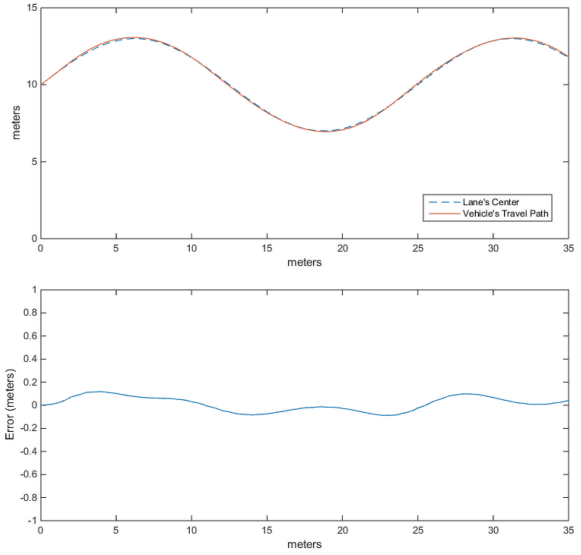


Fig. 17 Actual traveling path of a vehicle and resulted error in lane centering.

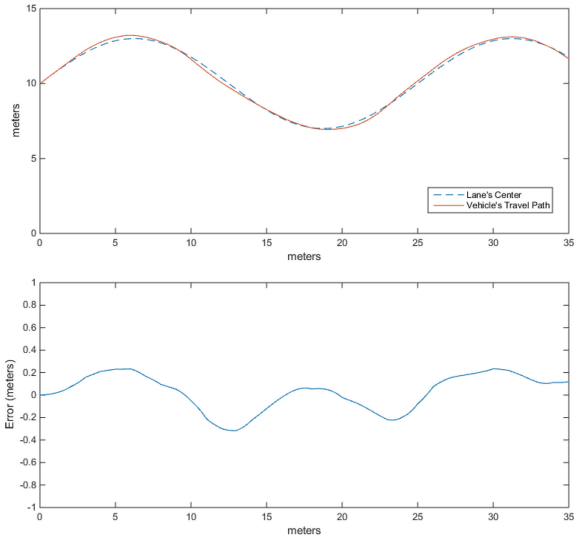


Fig. 18 Actual traveling path of a vehicle and resulted error in lane centering with random noise.

According to Fig. 17, under such a curvy road condition, the lane-keeping error of less than 0.15 m is achieved. Although an actuate estimation of lane position is used in the above simulation, the estimated positions of lane detection algorithms in an actual deployment can be noisy. To emulate that, another simulation is carried out with random noise of +/- 0.5 meters are induced in the estimation of the lane center's position. The result of the simulation is shown in Fig. 18. The RMS error and the maximum absolute error are 0.1509 m and 0.3155 m respectively.

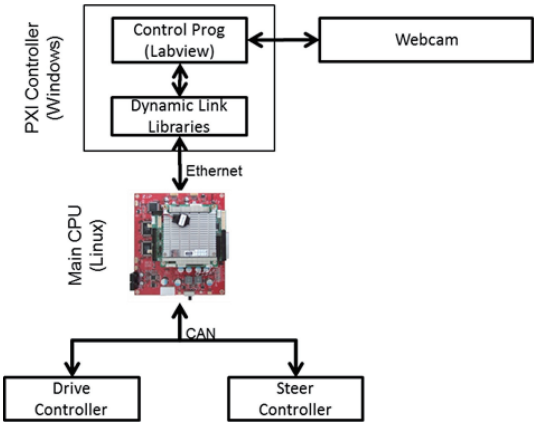


Fig. 19 Block diagram of overall system setup.

According to the above simulation, even though the noise increases the magnitude of the error, the proposed system is still stable and can keep track of the lane center. The next section will assess the actual performance of the proposed algorithms through physical experiments.

7 Experiment

The proposed algorithms are implemented in a Toyota Tsusho vehicle which is a single-seater EV. The overview of the system setup is shown in Fig. 19. A camera (Genius WideCam F100 [16]) is mounted on EV and is connected to the PXI controller (PCI extensions for Instrumentation) [17]. The specifications of the controller are shown in Table. I. LabView [18] is used as the main programming language to control the steering angle of the EV while the proposed algorithms are implemented in the MathScript Module [19] of the LabView.

Table 1 PXI's Specification

Description	Specifications
Model	NI PXIe-8133
Processor	i7-820QM, 1.73GHz
Memory	4 GB (4 x 1GB DIMM)
Operation System	Windows 7

The major components of the EV and its dimension are shown in Fig. 20 and Fig. 21 respectively. The EV contains a main central processing unit (CPU) controlling different parts of the vehicles mainly through a CAN bus. It directly controls the Drive controller, that is responsible for regulating the rear wheels' speed, and the Steer controller, that is responsible for adjusting the turning angle of the front wheels. The PXI controller is connected to the main CPU via Ethernet as shown in Fig. 19.

Fig. 20 Major components of EV.

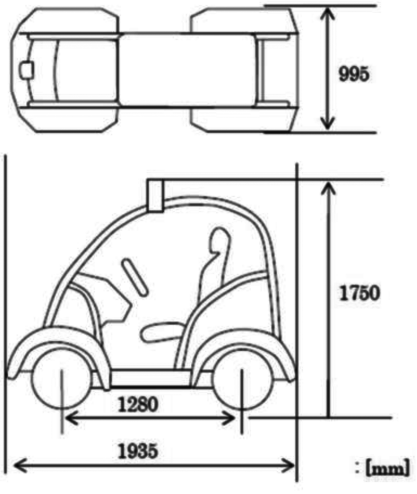


Fig. 21 Dimensions of EV.

Experiments were carried out to validate the performance of the implemented system. During the experiments, the EV was driven by the algorithms on a road section of the National University of Singapore (NUS) campus. The road section consists of unevenly distributed tree shading and the map of the road section is shown in Fig. 22. During the experiments, the lane tracking and the steering actions were done autonomously by the proposed algorithms at a sampling frequency of 4 Hz and the driving speed was kept at 10 km/hr.

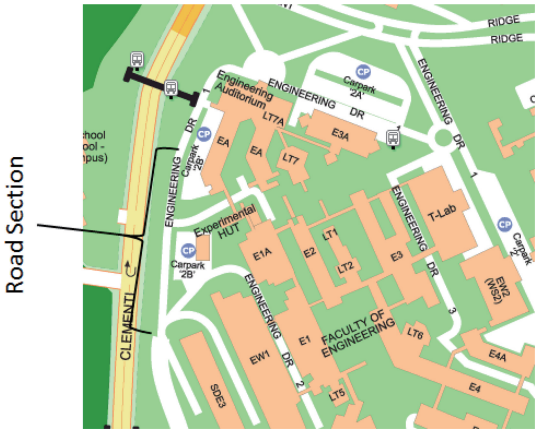


Fig. 22 The map of the road.

Three experiments were performed and the detail results from the 1st experiment are shown in Fig. 23. These experiments were carried out during the day and there were unevenly distributed tree shadows present on the road. In Fig. 25, a set of sample images captured with 20-frames-interval during the 1st experiment is provided. The lane-keeping errors during the next four experiments are plotted in Fig. 24. The experimental data are analyzed, and its summary is given in Table II.

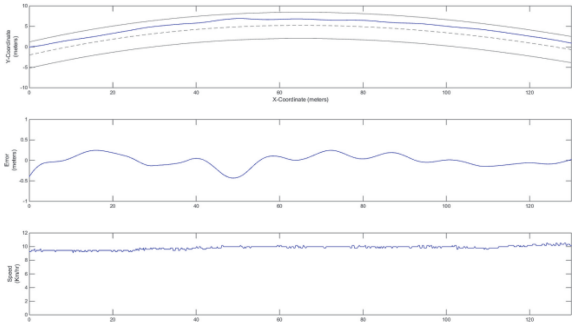


Fig. 23 Results from 1st Experiment.

According to Table II, the proposed system can keep a maximum error below 0.45m and the RMS error is an average of 0.124m. To validate the

Table 2 Experiments' Data

Experiment No.	1st	2nd	3rd	4th	5th
RMS Error (m)	0.143	0.119	0.103	0.131	0.123
Maximum Absolute Error (m)	0.432	0.313	0.254	0.284	0.291
Average Driving Speed (km/hr)	10.0	10.0	10.0	10.0	10.0

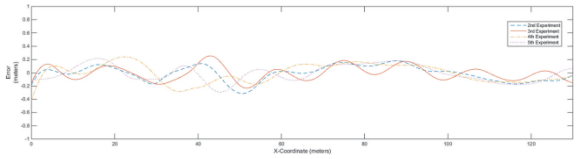


Fig. 24 Error results from the other four Experiments.

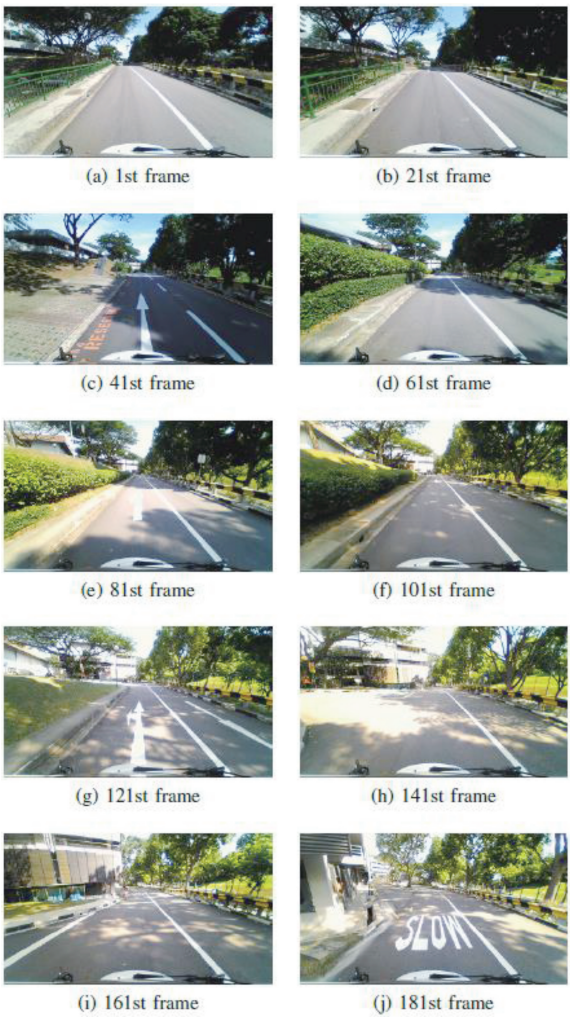


Fig. 25 Selected frames during the 1st experiment.

performance of the proposed algorithms in a night environment, another five sets of experiments were carried out on the same road section of the NUS campus. The first set of results are shown in Fig. 26 and the error results from the test are shown in Fig. 27. In Fig. 28, a set of sample images captured with 20-frames-interval during the first night experiment is shown. The data from the night experiments is investigated and its summary is presented in Table III.

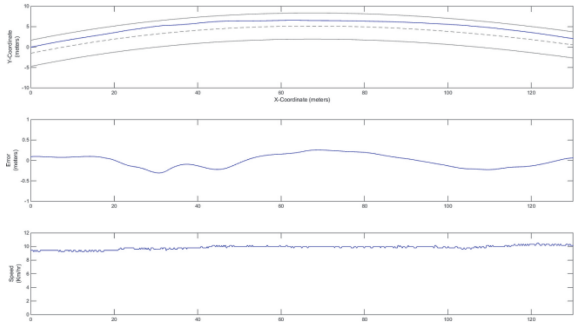


Fig. 26 Results from 1st Experiment.

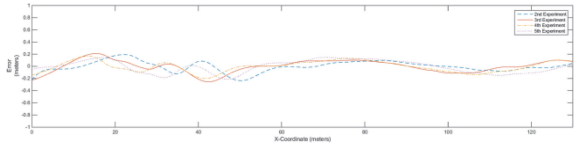


Fig. 27 Results from the other four Experiments.

From Table III, it is evident that the proposed system can keep the maximum error below 0.35m and the RMS error is at an average of 0.105m. During the experiment, the lighting condition is evenly distributed, and the road is relatively flat. Therefore, to verify the performance in a bad ambient condition on an undulating road, another five sets of experiments were carried out on a different road section of the NUS campus and the experiment results of the first run are shown in Fig. 29. The error results from the other four are shown in Fig. 30. In Fig. 31, a set of sample images captured with 20-frames-interval during the first experiment with a bad light condition is shown. The data from these night experiments is

Table 3 Night-Experiments' Data

Experiment No.	1st	2nd	3rd	4th	5th
RMS Error (m)	0.157	0.085	0.095	0.085	0.102
Maximum Absolute Error (m)	0.307	0.235	0.254	0.201	0.213
Average Driving Speed (km/hr)	10.0	10.0	10.0	10.0	10.0

Table 4 Night-Experiments' Data under the challenging condition

Experiment No.	1st	2nd	3rd	4th	5th
RMS Error (m)	0.142	0.127	0.145	0.094	0.141
Maximum Absolute Error (m)	0.329	0.302	0.332	0.255	0.330
Average Driving Speed (km/hr)	10.0	10.0	10.0	10.0	10.0

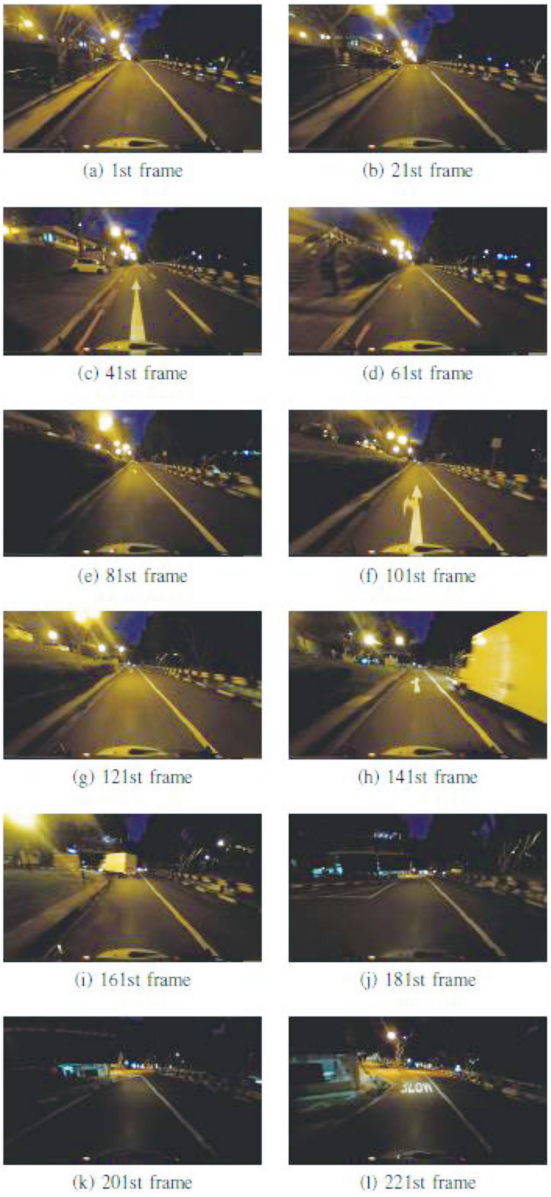


Fig. 28 Selected frames during 1st night-experiment.

investigated and its summary is presented in Table IV.

According to Table IV, the proposed algorithms can keep the maximum error below 0.35m and the RMS error is at an average of 0.130m. Even though the ambient condition is much worse than those in the previous two sets of experiments, the performance is still similar and thus, the LAIT algorithm can effectively reject the ambient effects from the environment.

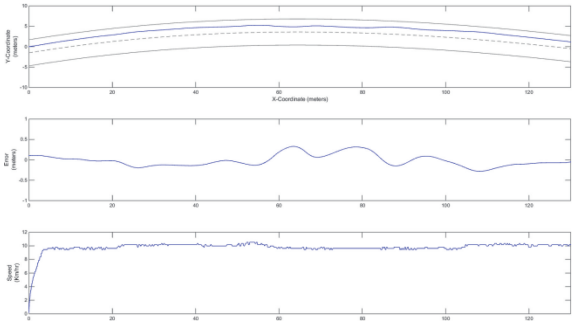


Fig. 29 Results from 1st Night Experiment with bad lighting.

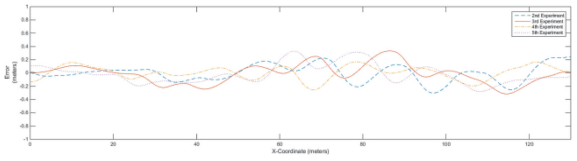


Fig. 30 Error results from the other four Experiments.

Therefore, the proposed system is verified to be able to keep the vehicle in the center of the lane within 0.45 m and with an average of 0.12m from

the center under three different environmental conditions. Video-footprints of the experiments can be seen in[20-21].

Fig. 28 Selected frames during 1st night-experiment under a bad lighting condition.

The average RMS errors for the three sets of experiments are 0.1238, 0.1048, and 0.1298. The average maximum absolute errors are 0.3148, 0.2420, and 0.3096. The variations in the RMS errors and the maximum absolute errors between different ambient light conditions are both within +/- 15% of the averages. Therefore, the proposed system can provide similar performance in different ambient lighting conditions.

8 Conclusion

This paper presented a lane-keeping approach which consisted of the lane detection system, using input only from a mono wide-angle camera, LEC algorithm and the steering control system. Through both simulations and experiments, the approach was verified to achieve a satisfactory performance as the average errors were less than the width of a lane marking while the maximum error was approximately 10% of average lane's width during all the experiments conducted under different lighting and road conditions. As this paper focused on utilizing only off-the-shelf web camera and a generic processor to perform the auto lane-keeping task, the performance can be further enhanced by fusing this system with additional data from motion sensors, GPS, and local map. Furthermore, a dedicated graphics processor or a field-programmable gate array (FPGA) can further improve computation efficiency.

References

[1] World Health Organization. (2009). Global status report on road safety: time for action. World Health Organization.

[2] Krug, E. (2012). Decade of action for road safety 2011-2020. Injury, 43(1):6-7.

[3] Deng, J. and Han, Y. (2013). A real-time system of lane detection and tracking based on optimized ransac b-spline fitting. In Proceedings of the 2013 Research in Adaptive and Convergent Systems, pages 157-164.

[4] Bai, L. and Wang, Y. (2011). Road tracking using particle filters with partition sampling and auxiliary variables. Computer Vision and Image Understanding, 115(10):1463-1471.

[5] Xu, H. Wang, X. Huang, H. Wu, K. and Fang, Q. (2009). A fast and stable lane detection method based on b-spline curve. In Computer-Aided Industrial Design & Conceptual Design, 2009. CAID & CD 2009. IEEE 10th International Conference on, pages 1036-1040.

[6] Miao, X. Li, S. and Shen, H. (2012). On-board lane detection system for intelligent vehicle based on

monocular vision. International journal on smart sensing and intelligent systems, 5(4):957-972.

[7] Muad, A. Hussain, A. Samad, S. Mustaffa, M. and Majlis, B. (2004). Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system. In TENCON 2004. 2004 IEEE Region 10 Conference, pages 207-210.

[8] Mukhopadhyay, P. and Chaudhuri, B. (2015). A survey of hough transform. Pattern Recognition, 48(3):993-1010.

[9] Broggi, A. (1995). An image reorganization procedure for automotive road following systems. In Image Processing, 1995. Proceedings., International Conference on, volume 3, pages 532-535.

[10] Hill, F. (2000). Computer graphics using opengl. Prentice Hall, page 678.

[11] Ballard, D and Brown, C. (1982). Computer vision, article, 4 pages prentice-hall. Englewood Cliffs, New Jersey, believed to be published more than one year prior to the filing date of the present application.

[12] Hough, P. (1962). Method and means for recognizing complex patterns. Technical report.

[13] Rogers, J. Trevor, A. Nieto-Granda, C. Cunningham, A. Paluri, M. Michael, N. Dellaert, F. Christensen, H. and Kumar, V. (2014). Effects of sensory precision on mobile robot localization and mapping. In Experimental Robotics, pages 433-446.

[14] Anand, V. (1996). Computer graphics and geometric modeling for engineers. John Wiley & Sons, Inc.

[15] Snider, J. (2009). Automatic steering methods for autonomous automobile path tracking. Robotics Institute, Pittsburgh, PA, Tech. Rep. CMURITR-09-08.

[16] Genius. Genius, Ultrawide angle Full HD webcam. Available at: http://www.ni.com/pxi/. [Accessed 1-Dec-2014].

[17] National Instrument. PXI Platform - National Instrument. Available at: http://www.geniusnet.com/Genius/wSite/ct?xItem=53214&ctNode=161. [Accessed 27-May-2015].

[18] National Instrument. NI LabVIEW - National Instrument. Available at: http://www.ni.com/labview/. [Accessed 1-Dec-2014].

[19] National Instrument. LabVIEWMathScript RT Module - National Instrument. Available at: http://www.ni.com/labview/mathscript/. [Accessed 1-Dec-2014].

[20] Htet, K. Youtube - Lane Keeping Experiments CameraViews: 3 each. Available at: https://www.youtube.com/watch?v=F5jbJq-2bUg. [Accessed 28-May-2015].

[21] Htet, K. Youtube - Lane Keeping Experiments on 23 May 2015. Available at: https://www.youtube.com/watch?v=bSrUA-4nZmg. [Accessed 28-May-2015].

Authors' Biographies

Kyaw Ko Ko Htet received his PhD in 2016 from the National University of Singapore (NUS). His main research interest is robotics, automation, and machine learning.

E-mail: Kyaw.Ko_Ko_Htet@u.nus.edu

Tan Kok Kiong received his PhD in 1995 from the National University of Singapore (NUS). Prior to joining the university, he was a research fellow at SIMTech. He is currently a professor with NUS and his current research interests are in precision motion control and instrumentation, advanced process control and auto-tuning, and general industrial automation.

E-mail: kktan@nus.edu.sg