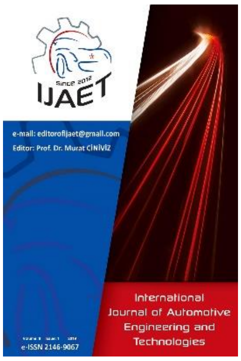
e-ISSN: 2146 - 9067

International Journal of Automotive Engineering and Technologies

journal homepage:

https://dergipark.org.tr/en/pub/ijaet


IJAET
Since 2012
e-mail: editorofijaet@gmail.com
Editor: Prof. Dr. Murat CİNİVİZ
International Journal of Automotive Engineering and Technologies
e-ISSN 2146-9067

Original Research Article

Small size vehicle application with lane tracking capability via CHEVP algorithm

Abdulhamit Sevgi[1], Emel Soylu[2, *], Raif Bayır[3]

[1] Karabük University, Natural, and Applied Sciences, Department of Mechatronics Engineering, 78050, Karabük, Turkey.
[2*] Samsun University, Faculty of Engineering, Department of Software Engineering, Samsun 55420, Turkey.
[3] Karabük University, Faculty of Technology, Department of Mechatronics Engineering, 78050, Karabük , Turkey.

ARTICLE INFO

[1] 0000-0003-3567-848X

[2] 0000-0003-2774-9778

[3] 0000-0003-3155-8771

* Corresponding author
emel.soylu@samsun.edu.tr

Received: Oct 28, 2019
Accepted: Mar 10, 2020

Published by Editorial Board Members of IJAET

ABSTRACT

Humans are living things that make mistakes. Failure of drivers while driving can cause life, cost, organ loss. Some accidents may remain traumatic in people's memory and adversely affect their future lives. Traffic rules have been developed to prevent such accidents. Traffic is formally organized in many jurisdictions, with marked lanes, junctions, intersections, interchanges, traffic signals, or signs. Rules of the road and driving etiquette are the general practices and procedures that road users are required to follow. Thanks to these guides placed on the roads, drivers can go on the road in harmony. Despite all these precautions, there are many traffic accidents caused by the driver's carelessness, sleeplessness and tiredness. In today's technology, it is possible to utilize methods of processing images from cameras located in the vehicle to minimize driver-induced accidents. In this study, a prototype system was established in order to use the technologies used in autonomous vehicles and to teach these technologies. Camera and computer are placed on a battery-powered vehicle. Using the OpenCV library, lane tracking is performed successfully using the Canny/Hough Estimation of Vanishing Points (CHEVP) method. The developed system is suitable for the use and development of image processing technologies that are used in autonomous vehicle technology. The system is tested in real-time on a designed runway. From the real-time experimental studies, high-performance results were obtained.

Keywords: Lane Tracking, CHEVP, Autonomous vehicle

1. Introduction

The definition of the autonomous vehicle in PC Mag magazine is made as a computer controlled self-driving vehicle [1]. Autonomous vehicles can detect objects around them by using technologies and techniques such as radar, LIDAR, GPS, odometer, and computer vision. Fully autonomous vehicles do not need drivers because they have the ability to control the road, traffic flow, and environmental factors. Autonomous vehicle technology is a new technology that has significant impacts on road transport in the medium and long term.

The history of the autonomous vehicle dates back to the 1920s. In the first trials, a second vehicle-controlled the other vehicle by radio signals [2]. A few decades later, studies were carried out in the UK and America with the idea of placing embedded electronic devices on the roads and driverless vehicles to be managed thanks to these devices, but because the expected results could not be achieved and this method was very expensive, investment funds were withdrawn. In the 1980s, with the team of Ernst Dickmanns of the Bundeswehr University, the Mercedes-Benz van succeeded in the work of a driverless vehicle with a speed of over 20km and reaching a maximum of 96km on an empty road. In 1989, a limited number of objects were recognized by the robotic vehicle. In 1990, lane changes were detected.

Studies on fully autonomous vehicles continue without losing speed. Developed countries are encouraging this issue by organizing autonomous vehicle competitions. Large companies such as Google, Tesla, Uber make investments on autonomous vehicles and lead the way in this field. Academics and large companies attach great importance to R & D (Research and Development) to develop autonomous vehicle technologies. According to the research conducted by The FP Think Working Group (Bierstedt et al., 2014), the use rate of autonomous vehicles is estimated to reach 95% in the 2040s. In 2050, it is foreseen that a completely driverless life will be started [3].

One of the important issues in R & D studies on autonomous vehicles is to keep the vehicle within the boundaries of the lane. It is desirable for the autonomous vehicle to recognize the lanes and move safely between these lanes. Kluge and Lakshmanan divide the problem of holding the vehicle in two-lane boundary lanes. One of them is the lane detection and the other is the lane monitoring [4].

The lane determination is carried out by processing the frames taken by the cameras located in the vehicle. The camera images are interpreted to produce meaningful information such as vehicle position, road signs, road boundaries, and vehicle direction. Moving the vehicle within the defined geometric strip boundaries is a relatively easy application than lane determination. Several algorithms have been implemented to identify lanes on highways and many filtering algorithms have been developed to improve image distortion.

Image processing and machine learning are two of the techniques used in lane identification. These two techniques have disadvantages in terms of compliance with environmental conditions and operation in different environments. For example, machine learning requires a learned path with well-painted lines or strong ribbon edges, otherwise, it will fail. On the other hand, the image processing-based technique uses only strips.

Assuming that strip shapes can be presented with straight line [5 - 8] or parabolic curve [4, 9, 9 - 11], this technique is much more robust to noise and missing data with machine learning by calculating and filtering parameters. For estimation of the parameters, the lane is determined by a probability function [4 – 6, 8, 9] Hough transformation [7] and the fragmentation modeling of roads [10, 11] and similar methods. CHEVP is resistant to shadows, lighting changes, and different noises, a problem that characterizes previously developed lane detection algorithms. Assumes that the path has two parallel lane marks to reflect an intersecting point. Due to the use of parallel information of waypoints, it defines a broader range of lane structures compared to other lane detection algorithms [12, 13]. The CHEVP algorithm is robust against noise, shadows, and illumination. This method is applicable to the dash and the solid paint line roads marked and the unmarked roads [14].

There are several lane models in the literature including flat and parabolic, Spline and so on methods can define strips more accurately [15 - 17]. However, depending on the algorithms, noise, shadows and different light environments may reduce accuracy. In addition, the calculation time may be higher as there are more parameters to optimize the resulting complexity in the calculation.

Studies on autonomous vehicles require high costs. In this study, a prototype experimental setup was created to use the technologies used in autonomous vehicles, to develop new technologies and to teach the existing technologies to the people working in this field. The camera and computer are placed on a battery-powered car that is one-fifth the width of a standard vehicle. With the help of computer vision, the vehicle is moved autonomously. In this application, lane tracking was applied successfully.

In the second part of the article, the materials used in the experimental setup of the study and the methods used in the software are given. Detailed information about lane identification and lane tracking is provided. Experimental results are given in section 3. Finally, in the last section, there is a discussion about the results obtained from this study and how it can be improved.

2. Materials and Methods

2.1. Experimental setup

An experimental test vehicle has been built to implement software used in autonomous vehicles in real-time. Since the width of the test vehicle is about 1/5 of a standard vehicle, the road on which the vehicle will travel is designed as 1/5 of the standard road. Instead of making the vehicle from scratch, an experimental system was created by making modifications to an existing battery-powered vehicle. In order to perform the experiments, the battery-powered vehicle has been preferred as seen in Fig. 2. The engines and wheels of this vehicle were replaced. On camera, motor drive, steering motor drive, steering motor encoder has been added. The computer was placed on the vehicle for image detection and lane detection.

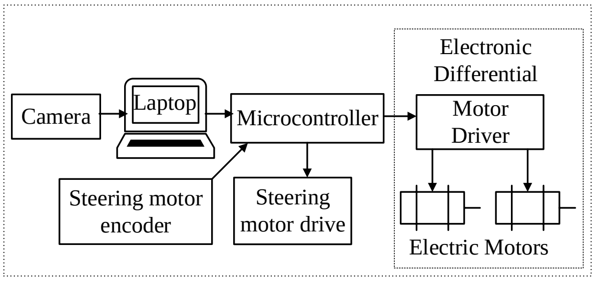


Figure 1. Block diagram of the experimental setup

Fig. 1 shows the block diagram of the test setup. Images taken from the camera are processed on the computer and lanes are determined. Steering and engine control are performed to place and move the vehicle between the designated lanes. Motor speed and steering are controlled by a microcontroller. Fig. 2 shows the test vehicle on the runway. The components shown in the block diagram are marked on the photo.

Arduino MEGA 2560 has been preferred as a microcontroller. It is an open-source development platform. It is one of the most advanced ATmega-based Arduinos in terms of processors. This board has a total of 70 pins, digital and analog, one 16 MHz crystal oscillator, USB connection, power jack and reset button. The Arduino Mega 2560 contains all of the components needed to support a microcontroller. The Arduino Mega 2560 can be connected to a computer, powered by an adapter or battery.

Experimental setup photos from different angles are given in Fig. 3, Fig. 4, Fig. 5 respectively.

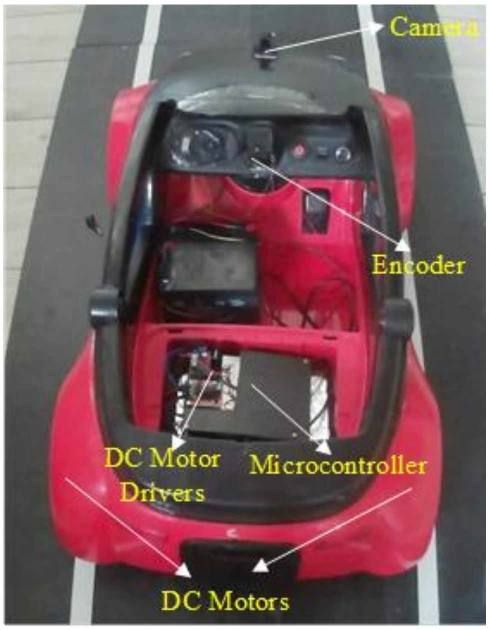


Figure 2. Experimental Setup

Figure 3. Experimental setup view from above

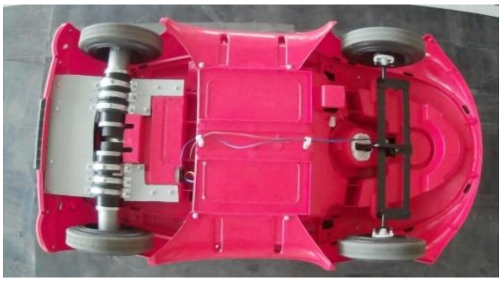


Figure 4. Bottom view of the experimental setup

Figure 5. Inside view of the experimental setup

Sj4000 model action camera is used as a camera. This camera has a 12 MP CMOS Sensor and supports 12 MP, 10 MP, 8 MP, 5 MP, 3 MP 2MHD, VGA and 1.3MP shooting and has Wifi feature. Shooting options of 30 fps in Full HD, 60 fps in HD, and 60 fps in 480p provide fast data support for real-time image processing.

3590S 10K multi-turn potentiometer is used as the encoder for steering control. Steering angles are transmitted analogously to the microcontroller to control the position of the steering motor and steering.

Two DC motors are used in the vehicle. These motors are 24V, 90W, 330d / min, 2.5Nm planetary geared motors.

Roboclaw 2x7A is used to drive DC motors. RoboClaw has Dual channel quadrature decoding and can drive 2 DC motors. Position and speed control is possible. The input voltage range is 6-34V and can offer 7.5A continuous current per channel. Enables easy PID programming with the encoder connection on the board.

2.2. Lane detection and lane tracking

The flow diagram of the software used in lane recognition and tracking is given in Fig. 6. This section describes the steps in this flow diagram. The road view taken from the vehicle is not a bird's eye view, but a perspective view. Mathematical modeling is done first for clipping the path from the image. Mathematical modeling of the strip plays an important role in the determination of the strip. With the two-dimensional mathematical modeling, the model in Fig. 7 is created by making assumptions for the three-dimensional real world.

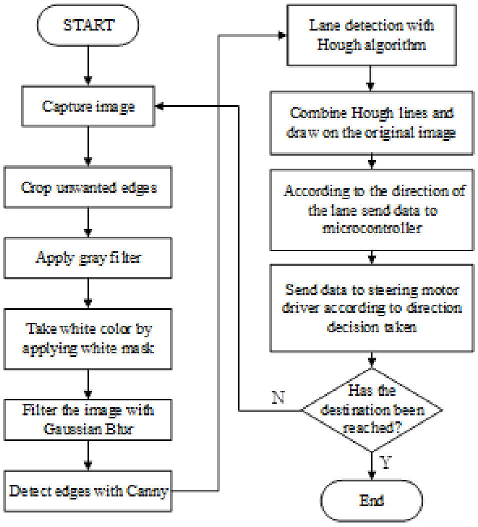

START
Capture image
Crop unwanted edges
Apply gray filter
Take white color by applying white mask
Filter the image with Gaussian Blur
Detect edges with Canny
Lane detection with Hough algorithm
Combine Hough lines and draw on the original image
According to the direction of the lane send data to microcontroller
Send data to steering motor driver according to direction decision taken
Has the destination been reached?
N
Y
End

Figure 6. Flow chart of the system

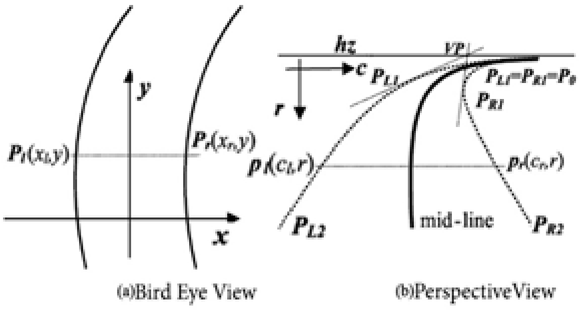


Figure 7. Bird's eye view and perspective view of the lanes [9]

The distance between the points $P_l(x_l,y)$ and $P_r(x_r,y)$ shown in Fig. 7 (a) is x_l- x_r, while the distance between them in the perspective view in Fig. 7 (b);

$$D = k(y - hz) \tag{1}$$

Primarily, the significant path width (D) depends on the distance between *a* given horizon and the *y* coordinate and the constant *k*. *k* is a constant derived from the focal length of the path used and the height of the camera position and is different for each path [11].

We can find the equivalents of a point in the middle line in the left and right lanes with Equations 2 and 3, respectively.

$$P_l = x_c - \frac{1}{2}D = x_c - \frac{1}{2}k(y - hz) \tag{2}$$

$$P_r = x_c + \frac{1}{2}D = x_c + \frac{1}{2}k(y - hz) \tag{3}$$

In order to perform the above mathematical modeling at an optimum level, the parameter *k* must be set according to the width of the path. The results of decreasing or increasing the parameter *k* are given in Fig. 8 (Straight lines are real road edges, dashed lines are lane models.).

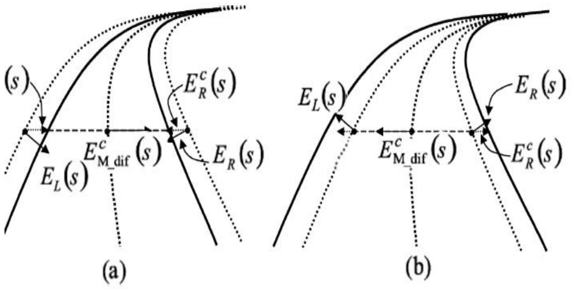


Figure 8. Reducing the k parameter (a) Increasing the k parameter (b) [9]

According to mathematical modeling, after the image is cropped, the number of colors is reduced to two by applying a gray filter to the image. The resulting output is only white color by white masking. White colors displayed on the output are detected by the Hough transform method. Noise in the picture is eliminated with a Gaussian Blur filter. For the detection of the strips, the CHEVP algorithm, which Wang et. [16] discovered, is first applied. In this algorithm, the borderlines are determined with CHEVP. In this method, the path is assumed to be a plane consisting of two parallel lines. In this study, unlike classical CHEVP, perspective image is transformed into a bird's eye view with mathematical modeling. The resulting image was the input of the CHEVP algorithm [9]. This algorithm has six processing steps.

The vanishing point is determined for each section by dividing the image into sections. The vanishing point is estimated as the intersection point between the left lane and the right lane. Each vanishing point is the trigger for the next movement. The division of the image is shown in Fig. 9. Then the divided image is converted from perspective to a bird's eye view.

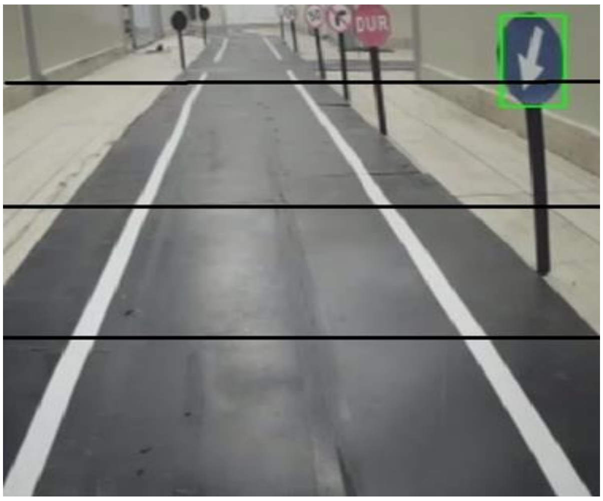


Figure 9. Image of split road

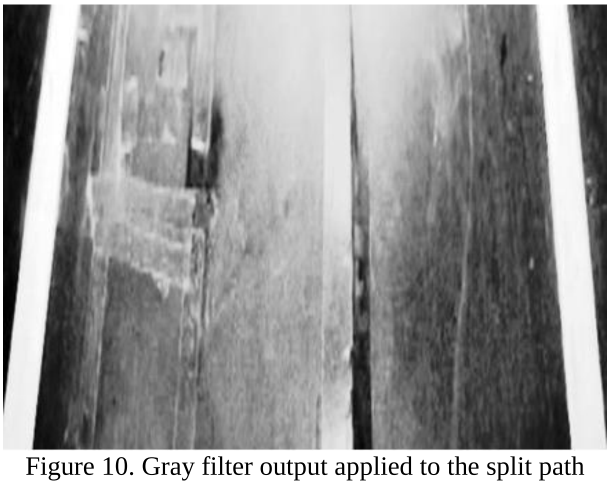


Figure 10. Gray filter output applied to the split path image

In the second step, the gray filter is applied according to the average method. Each color pixel is described by a triple (R, G, B) of intensities for red, green, and blue. The average method simply averages the values: (R + G + B) / 3 [18]. At this stage, the presence of yellow and white strips is made and the contrast of the strips is increased. The output is given in Fig. 10.

For gray image output, a white filter and a gauss filter are applied. The white filter is for the detection of strips and the gauss is the preprocessing step used to reduce or correct the noise in the image [19]. This preprocessing step is used to remove many detected edges and keep only the most prominent edges in the image. The image output is shown in Fig. 11.

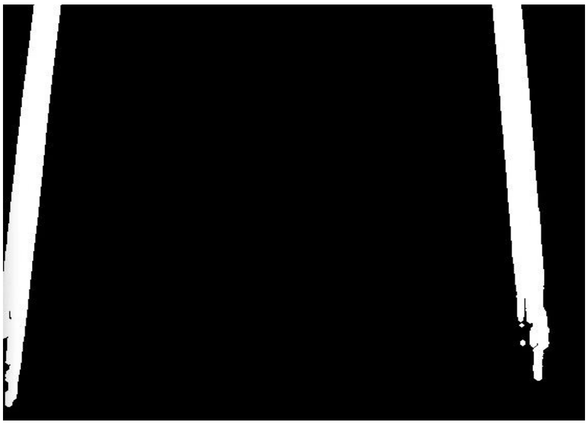


Figure 11. The Gaussian blur and the White filter output

The Gaussian blur is a type of image-blurring filter that uses a Gaussian function for calculating the transformation to apply to each pixel in the image. The formula of a Gaussian function in two dimensions is given in Equation 4 where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution [20].

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4}$$

Edge detection and edge pixel extraction are performed using the Canny algorithm [21]. In Fig. 14 Canny is basically an algorithm that detects edges. The Canny algorithm has four main steps:

- Noise reduction with Gauss filter (In this system, this step is done before the Canny algorithm in the third stage).
- Determination of gradient density; A Sobel core is applied along the x and y axes to determine whether the edges are diagonal, horizontal or vertical in the flattened image. Sobel kernel equations are given in Equations 5 and 6.

$$(G) = \sqrt{G_x^2 + G_y^2} \quad (5)$$

$$\theta = \tan^{-1}\frac{G_x}{G_y} \quad (6)$$

The edges are sharpened and compared to the previously calculated gradient value for each pixel. The point A is the edge shown in Fig. 12. Points B and C are gradient points. The points B and C are checked to see if point A is the local maximum. If the local does not form a maximum, it is synchronized to 0.

- Hysteresis threshold; at this stage, it is decided whether the edges are the actual edge. This requires two thresholds, minVal and maxVal. It is not the actual edge if the density gradient is greater than maxVal and not the true edge if it is below minVal. In Fig. 13, in the hysteresis thresholding example, the actual edge is assumed because point A is above maxVal, and the actual edge is assumed because point B is connected to point A. Since point C is below minVal, it is not considered the actual edge.

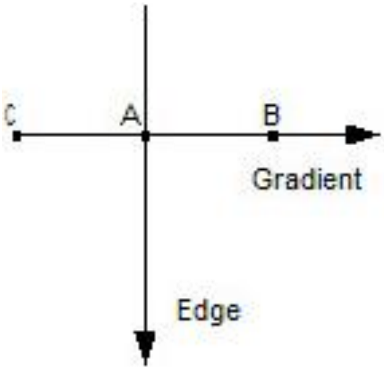


Figure 12. Non-maximum Suppression representation with three points

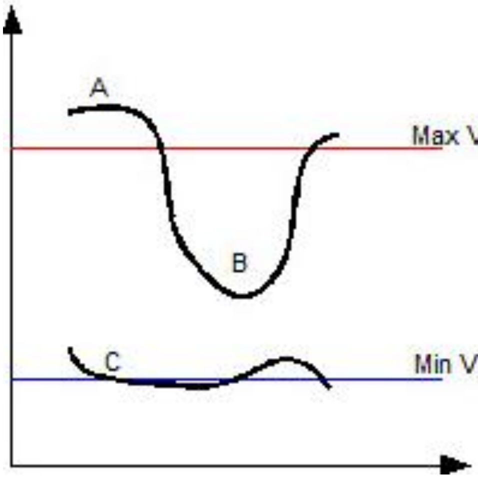


Figure 13. Hysteresis thresholding example

After the Canny algorithm, straight lines are detected by the Hough transform method [22]. This is done by showing a line as a dot and the points are shown sinusoidal. If multiple lines cross the sinusoidal point, it can be detected that these points are in the same line. Equation 7 is used for this:

$$x \sin\theta + y \sin\theta = r \quad (7)$$

r in Equation 6 represents the distance from the reference point to the line and θ represents the angle that r makes with the positive x trajectory. This is shown graphically in Fig. 15.

Figure 14. Canny edge algorithm output

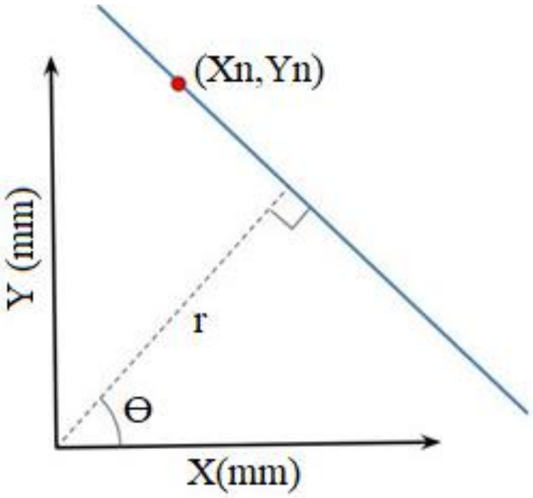


Figure 15. Representation of points in space (x, y)

Each point will be transformed into a sinusoidal shaped line. Fig. 16 shows an example of this. The points in the line will have a common intersection, which represents the correct θ and r values of that line.

One of the major advantages of the Hough transformation is its ability to cope with noise, missing lines, and similar negativities. Since we do not want to detect absolute precision and horizontal lines, only the full and positive *sin (θ, r)* fields are used. The output from the Canny edge sensor will have more effect on the strong edges than the weak edge during the Hough transformation.

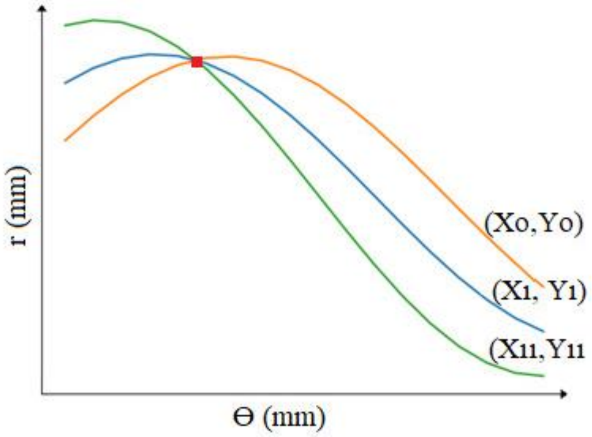


Figure 16. Representation of lines in space (θ, r)

The right and left lanes in the cropped image of the path are identified. They are subject to manual prediction criteria based on their angle and length. The parameters that meet the criteria are defined by drawing in green color as shown in Fig. 17. A midpoint determination is made for the right and left strips.

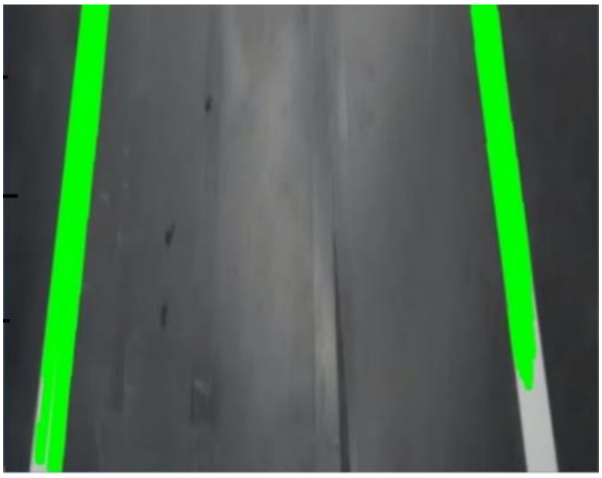


Figure 17. The marked output of lanes

Figures 18 to 22 show visuals of lane fixation in road views taken at different angles. The leftmost is the original image. As you move to the right, the image is passed through the lane tracking stages, respectively. The result is the rightmost images.

In real-time experiments, the vehicle is held on the road according to the angle between the two lanes. The angle is found according to Equations 8 and 9 where y_{last} y lane last value on the y axis, y_{first} is initial value, x_{last} is x lane last value on the x axis, x_{first} is initial value and Θ is the angle between two lanes.

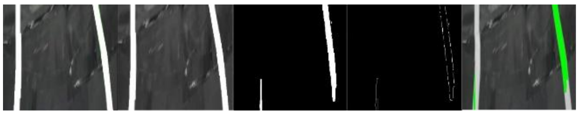


Figure 18. Road view 1

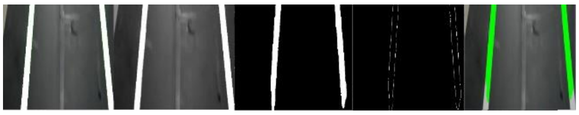


Figure 19. Road view 2

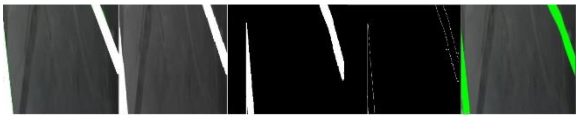


Figure 20. Road view 3

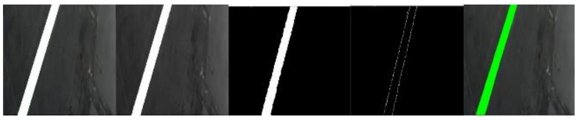


Figure 21. Road view 4

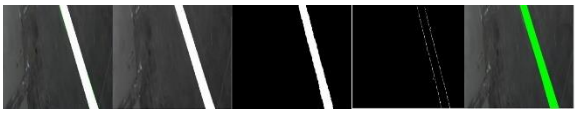


Figure 22. Road view 5

$$m = \frac{y_{last} - y_{first}}{x_{last} - x_{first}} \tag{8}$$

$$\emptyset = \arctan(m) \tag{9}$$

3. Results and Discussion

The runway has given in Fig. 23 was established to perform experimental studies. There are straight sections, sharp cornering sections and spline sections on the designed road.

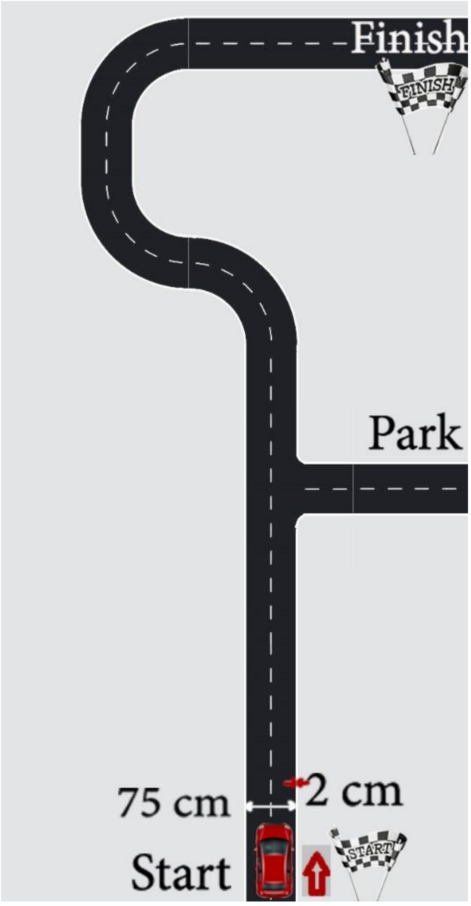


Figure 23. Vehicle runway

The test vehicle was started 50 times at different times of the day from the starting point and successfully reached the target point 49 times autonomously. This result shows that the developed system works with an accuracy rate of 98%. Even if the angle of incidence, brightness changes, the system performs vehicle lane tracking in a robust manner. The advantage of Canny over the conventional color filter method is that it can operate stably even in light and shadow differences.

For three situations the reference angle and the measured angles according to the vehicle's position on the road are shown in Fig. 24. The reference angles are calculated on the runway for a vehicle driving right in the middle of the two lanes. Measured value is the angle of the vehicle to the lanes.

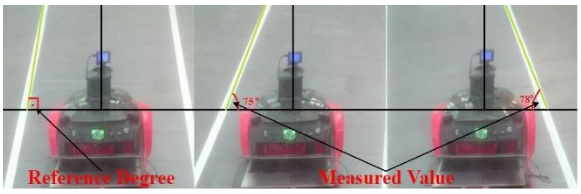


Figure 24. Reference and measured degrees on runway

In the following 4 graphs (Fig.25 - Fig. 28), the reference angles between the two lanes of the 4 real-time experiments and the angle values calculated according to the position of the vehicle are given. 90-degree angles in the reference angle curve refer to the straight path, other angles to left or right bends. As can be seen from the measurement curves, the vehicle manages to stay between the two lanes.

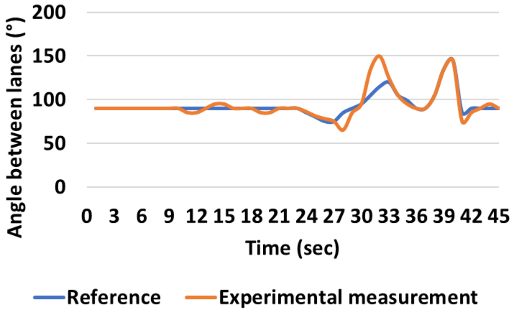


Figure 25. Real-time experiment 1

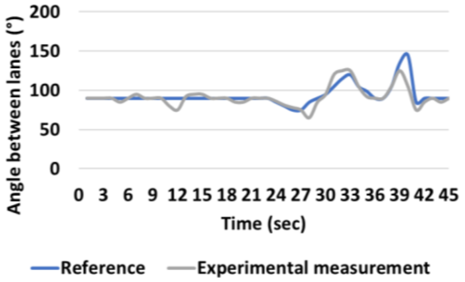


Figure 26. Real-time experiment 2

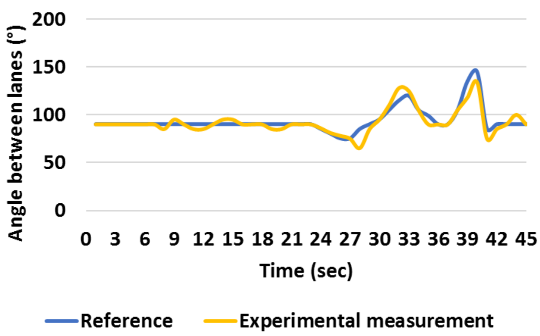


Figure 27. Real-time experiment 3

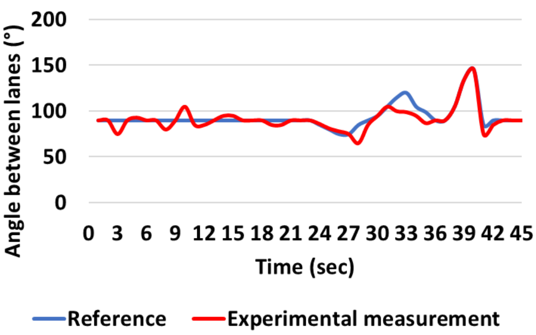


Figure 28. Real-time experiment 4

$$AE = |RV - MV| \tag{10}$$

$$RE = \frac{AE}{RV} \tag{11}$$

The reference value changes according to the driving speed. To calculate the approximate accuracy of the ride, we manually calculated this value with a sampling rate of 1 second according to the position of the vehicle on the runway. For the comparison and accuracy calculation, we chose 4 experiments with the same experiment duration as 45 seconds. Relative errors are calculated for experiments according to Equations 10 and 11 for each second of experiment. *AE* is absolute error, *RV* is reference value, *MV* is measured value, *RE* is relative error in these equations [23]. Average relative errors for experiment 1 during 45 seconds is 3.53%, for experiment 2 is 4.24 %, for experiment 3 is 3.51% and for experiment 4 is 4.14%. The average error of these four experiment is 3.85%. According to these four experiments, the system works with an accuracy of 96.15%.

4. Conclusions

In this study, an autonomous vehicle application that is able to follow the lane has been applied. A battery-powered vehicle was modified to perform the experiments. The runway was created for lane tracking. CHEVP algorithm was used for lane tracking. With this method, lane tracking was performed in real-time.

Experimental results show that this algorithm works optimally. The vehicle tracking can be successfully completed at 98% and deviations can be minimized even at different light angles. Testing software on real cars is costly. To that and finding the accuracy of the experiments, closed roads are needed. With this in mind the experimental setup we have developed, it is much cheaper to develop and test autonomous vehicle software. Furthermore, it is easy to make changes on the test track and it takes a short time.

The test apparatus and runway formed the basis of the studies to be carried out thereafter. As well as new algorithms can be tested on the developed system and the validity of the developed algorithms can be tested. In addition to lane tracking, more specific systems can be developed by adding features such as traffic sign recognition and object recognition. Developed software can be run on embedded system cards to save weight and cost.

Acknowledgment

Karabük University supported this study within the scope of Scientific Research Projects (KBÜ-BAP-18-YL-163).

We would like to thank KBÜELAR electric vehicle and ROBOTAXI team for their contributions.

6. References

1. “Definition of: self-driving car”, 2019. [Online]. Available: https://www.pcmag.com/encyclopedia/term/65738/self-driving-car. [Accessed: 03-Sep-2019].
2. J. Berrada and F. Leurent, “Modeling Transportation Systems involving Autonomous Vehicles: A State of the Art”, Transp. Res. Procedia, 27, 215–221, 2017.
3. P. Davidson and A. Spinoulas, “Autonomous vehicles: what could this mean for the future of transport”, in Australian Institute of Traffic Planning and Management (AITPM) National Conference, Brisbane, Queensland, 2015.
4. K. Kluge and S. Lakshmanan”, A deformable-template approach to lane detection,” in Proceedings of the Intelligent Vehicles’ 95. Symposium, 54–59,1995.
5. S. Lakshmanan and D. Grimmer, “A deformable template approach to detecting straight edges in radar images”, IEEE Trans. Pattern Anal. Mach. Intell., 18: 4, 438–443, 1996.
6. K. Kaliyaperumal, S. Lakshmanan, and K. Kluge “, An algorithm for detecting roads and obstacles in radar images”, IEEE Trans. Veh. Technol., 50:1, 170–182, 2001.
7. D. J. Kang, J. W. Choi, and I. S. Kweon, “Finding and tracking road lanes using” line-snakes”, in Proceedings of Conference on Intelligent Vehicles, 189–194,1996.
8. S.-P. Liou and R. C. Jain, “Road following using vanishing points”, Comput. vision, Graph. image Process., 39:1, 116–130, 1987.
9. S. Lakshmanan and K. C. Kluge, “Lane detection for automotive sensors”, in 1995 International Conference on Acoustics, Speech, and Signal Processing, 5,2955–2958, 1995.
10. Kaske, R. Husson, and D. Wolf, “Chi-square fitting of deformable templates for lane boundary detection”, in IAR Annual Meeting, 95,1995.
11. Kaske, D. Wolf, and R. Husson, “Lane boundary detection using statistical criteria”, in International Conference on Quality by Artificial Vision, QCAV9, 28–30, 1997.
12. Y. Wang, E. K. Teoh, and D. Shen, “Lane detection and tracking using B-Snake”, Image Vis. Comput., 22:4, 269–280, 2004.
13. G. C. Verghese, “Estimation with Minimum Mean Square Error”, in ignals, Systems and Inference, 139–160, 2010.
14. P. T. Mandlik and L. Tracking, “A Review on Lane Detection and Tracking Techniques”, 3:5, 37–44, 2016.
15. Y. Wang, D. Shen, and E. K. Teoh, “Lane detection using catmull-rom spline”, in IEEE International Conference on Intelligent Vehicles, 1, 51–57,1998.
16. Y. Wang, D. Shen, E. K. Teoh, and H. Wang, “A Novel Lane Model for Lane Boundary Detection.”, in MVA, 27–30, 1998.
17. Y. Wang, D. Shen, and E. K. Teoh, “Lane detection using spline model”, Pattern Recognit. Lett., 21:8, 677–689, 2000.
18. J. D. Cook, “Three algorithms for converting color to grayscale”, 2009. [Online]. Available: https://www.johndcook.com/blog/2009/08/24/algorithms-convert-color-grayscale/. [Accessed: 10-Dec-2019].

19. S. Irandoust-Pakchin, A. Ayanzadeh, and S. Beikzadeh, "Gaussian Three-Dimensional kernel SVM for Edge Detection Applications", arXiv Prepr. arXiv1710.01260, 2017.

20. L. G. Shapiro and G. C. Stockman, Computer Vision. Prentice Hall, 2001.

21. J. Canny, "A computational approach to edge detection", IEEE Trans. Pattern Anal. Mach. Intell., 6, 679–698, 1986.

22. J. Illingworth and J. Kittler, "A survey of the Hough transform", Comput. vision, Graph. image Process., 44:1, 87–116, 1988.

23. M. Hazewinkel, "Theory of errors.", Encyclopedia of mathematics, 62, 2001.