Design and implementation of obstacles detection in self-driving car prototype

R P Sari¹², I P D Wibawa¹ and C Ekaputri¹
¹Telkom University, School of Electrical Engineering, Bandung, Indonesia

E-mail: rizkaps@student.telkomuniversity.ac.id

Abstract. Technology progresses very rapidly that makes a variety of industries vying to produce new innovations, especially the automotive industry. One example of innovation in the automotive industry is a self-driving car that can drive itself. Therefore, a self-driving car prototype is needed. This research focuses on the obstacles detection and calculation of obstacle distance. The obstacle detection uses a webcam with digital image processing. Obstacles which are detected are quiescent blue object that blocks the path of a self-driving car prototype and red light of traffic lights. The results of this research have a success rate at distance from webcam to object with actual distances of 38.3 cm and 55 cm amounting to 85.9% and 92.83%, the success rate at distance from webcam to blue obstacle with actual distances of 37.8 cm and 44.5 cm are 91.865% and 97.744%, the test results of red HSV value have accuracy 80%, precision 76.47%, and recall amounting to 86.667%, and the test results of blue HSV value have accuracy 83.33%, precision 81.25%, and recall 86.667%.

1. Introduction
Technology progresses very rapidly that makes a variety of industries vying to produce new innovations, especially the automotive industry. One example of innovation in the automotive industry is a self-driving car that can drive itself which can help humans to drive from one place to another [1].

Some research on the self-driving car has been conducted since the 1900s and some automotive companies such as General Motors (GM), Ford, Honda, Toyota, Renault-Nissan, Volvo, Hyundai, Daimler, BMW, and others will release related products self-driving car on in the 2020s. In several trials of the self-driving car, there have been various accidents. One of them was the Tesla Model S which happened in Florida because it failed to brake on time [2].

Based on the explanation, this research raises the topic of a self-driving car prototype that focuses on obstacles detection. Detection of obstacle or object has been carried out in several studies in a different application. One of them according to Shivani Godha in her journal entitled "On-Road Obstacles Detection System for Driver Assistance" using boundary detection algorithms with Matlab gives improper results and failure sometimes due to some processing delay [3]. So, we choose the HSV color segmentation method on Python program with Open CV to detect the obstacles that the implementation of this method on video sequences has shown properly extracts moving objects while removing their shadows [4]. Obstacles detection is processed when the object is quiescent and blue which blocks a self-driving car prototype path and red light of traffic lights. This obstacle detection produces distances between the webcam and the obstacles and then can be processed to control the speed of a self-driving car prototype.
2. Literature Review

2.1. Self-driving car
A self-driving car is a vehicle that is able to operate by looking at the surrounding environment and determining the direction of the destination without a driver. Autonomous driving has three major components: algorithms, including sensing, perception, and decision; client systems, including the operating system and the hardware platform; and the cloud platform [5]. The sensing of this prototype is a webcam.

2.2. Image processing
Digital image processing is the study of image processing techniques. Digital image processing generally refers to digital two-dimensional image processing (images). The image is in the form of a still image (photo) and a moving image (from a webcam). A moving image is a series of still images displayed sequentially (frame to frame). An image is represented numerically with discrete values so that it can be processed with a digital computer. A digital image is a two-dimensional array or matrix f(x, y) [6].

2.3. HSV Segmentation
Obstacles detection of this prototype uses a color segmentation method. The color segmentation method is applied to the frame to divide the image into several parts based on color. The first step in this segmentation is to determine the HSV value that is right, red for red light detection and blue for other obstacles. The HSV value is set to make parts other than red or blue in the unread image. The HSV value is obtained from the conversion of RGB values. Calculation of RGB to HSV conversion can be formulated as follows:

\[ H = \tan \left( \frac{3(G - B)}{(R - G) + (R - B)} \right) \] (1)

\[ S = 1 - \left( \frac{\min(R, G, B)}{V} \right) \] (2)

\[ V = \frac{R + G + B}{3} \] (3)

However, in the above formula when \( S = 0 \) then \( H \) can’t be determined [7]. For that, RGB normalization is needed first with the following formula:

\[ r = \frac{R}{R + G + B} \] (4)

\[ g = \frac{G}{R + G + B} \] (5)

\[ b = \frac{B}{R + G + B} \] (6)

By utilizing the normalized values of \( r, g, \) and \( b, \) the RGB to HSV transformation formula is as follows:

\[ V = \max(r, g, b) \] (7)
\[ S = \begin{cases} 0 & \text{ji} V = 0 \\ \frac{\min(r, g, b)}{V} & \text{ji} V > 0 \end{cases} \] 

(8)

\[ H = \begin{cases} 0 & \text{ji} S = 0 \\ \frac{60 \times (g - b)}{S \times V} & \text{ji} V = r \\ \frac{60 \times \left[ 2 + \frac{(b - r)}{S \times V} \right]}{S \times V} & \text{ji} V = g \\ \frac{60 \times \left[ 4 + \frac{(r - g)}{S \times V} \right]}{S \times V} & \text{ji} V = b \end{cases} \] 

(9)

\[ H = H + 360 \] 

ji \( H < 0 \) 

(10)

After segmenting the color, the next step determines the contour to help get the distance from the webcam to the obstacle. Determining the contours of the frame can be seen in Figure 1 and Figure 2. The center point obtained serves as a helping component in getting distance from the webcam to the obstacle of a self-driving car prototype.

Figure 1. The contours of red lights

Figure 2. Blue obstacle contours

The formula of distance from webcam to obstacle is obtained based on the relationship between several actual distance samples with a large pixel area using Geogebra Application that is showed in Figure 3 for a red light and Figure 4 for a blue obstacle. From these samples, the following formula is obtained:

\[ \text{Red light distance (cm)} = 1443.08 \times \text{Area of Pixel}^{-0.46}. \] 

(11)

\[ \text{Blue obstacle distance (cm)} = 459.94 \times \text{Area of Pixel}^{-0.25}. \] 

(12)
The distance formulas are produced by Geogebra Application with power trendline. Power trendline is chosen because power trendline has the second-highest $R^2$ value after polynomial trendline. The polynomial trendline is not chosen because coefficient of pixel area value is zero. Therefore, obstacle distance is unrelated to the pixel area.

3. Design and Implementation
The control system for a self-driving car prototype was tested on the 1:10 scaled vehicle, as showed in Figure 5. This self-driving car prototype is designed to be able to follow the road and detect obstacles automatically using a webcam. This prototype has a maximum angle and a minimum angle of 120
degrees and 60 degrees. Image processing of this self-driving car prototype runs on python programs with open CV using a personal computer and using Arduino Uno for the control system. The whole system block diagram can be seen in Figure 6.

![Figure 5. A self-driving car prototype design](image)

**Figure 5.** A self-driving car prototype design

**Figure 6.** System Block Diagram

Practical experiments were conducted on a private lane which can be seen in Figure 7. The lane is composed of several cornering sections with obstacles added. During testing, the placement of a camera is set beforehand to suit the lane that has been designed. The placement of a camera as depicted in Figure 8 affects webcam capture to determine the maximum distance from webcam to objects. A self-driving prototype detects obstacles based on color, the process of obstacles detection can be seen in Figure 9.

![Figure 7. A self-driving car prototype lane](image)
4. Experiments and Results
Distance measurement test is done by measuring the distance from webcam to obstacle using a 3 m measuring tape. Testing is done by comparing the read distance on python with the actual distance, testing red HSV value to find out if another red object is detected or not for red light detection, testing blue HSV value to find out if another blue object is detected or not for blue obstacle detection. Figure 10–13 and Table 1-2 show test results of the distance from webcam to the object.

Figure 8. The placement of a camera

Figure 9. The flowchart of obstacles detection

Figure 10. The distance testing result at 38.3 cm for a red light
Figure 11. The distance at 55 cm for a red light

Figure 12. The distance testing result for a blue obstacle

Figure 13. The result of blue obstacle distance at 44.5 cm

Table 1. Test results of red HSV value for red light detection

|          | Red Light | Not Red Light |
|----------|-----------|---------------|
| Positive | $\sum TP = 13$ | $\sum FP = 4$ |
| Negative | $\sum FN = 2$   | $\sum TN = 11$ |

Table 2. Test results of blue HSV value for blue obstacle detection

|          | Blue Obstacle | Not Blue Obstacle |
|----------|---------------|-------------------|
| Positive | $\sum TP = 13$ | $\sum FP = 3$   |
| Negative | $\sum FN = 2$   | $\sum TN = 12$   |
The success rate at distance from webcam to red light with actual distances of 38.3 cm and 55 cm with an average distance of 44.5875 cm and 51.319 cm are 85.9% and 92.83%. The success rate at distance from webcam to blue obstacle with actual distances of 37.8 cm and 44.5 cm with an average distance of 34.956 cm and 43.518 cm are 91.865% and 97.744%. The greater obstacle distance detected the smaller error generated. The test results of red HSV value have accuracy 80%, precision 76.47%, and recall amounting to 86.667% and the test result of blue HSV value have accuracy 83.33%, precision 81.25%, and recall 86.667%.

5. Conclusions
Based on the results of obstacle detection on self-driving car prototype that has been done, conclusions can be taken that based on distance testing for red light with actual distance of 38.3 cm and 5 cm have error 14.1% and 7.17%, for blue obstacle with actual distance of 37.8 cm and 44.5 cm have error 8.1346% and 2.256%, test results of red HSV value, this research have accuracy 80%, precision 76.47%, and recall amounting to 86.667%, and test result of blue HSV value have accuracy 83.33%, precision 81.25%, and recall 86.667%. Based on testing results, a self-driving car prototype can detect obstacles to form red light and quiescent blue object pretty well.

Acknowledgments
This research was funded by the Telkom University for research collaboration with Hanbat National University and Research Institute for Tea and Cinchona.

References
[1] L. Dormehl and S. Edelstein, “Sit Back, Relax, And Enjoy a Ride Through The History of Self-Driving Cars,” Digitaltrends, 03-Feb-2019.
[2] “Tesla’s Autopilot was engaged during Utah crash,” CNBC, 15-May-2018.
[3] S. Godha, “On-Road Obstacle Detection System for Driver Assistance,” Asia Pacific J. Eng. Sci. Technol., pp. 16–21, 2017.
[4] B. Hdioud, M. E. H. Tirari, R. O. H. Thami, and R. Faizi, “Detecting and shadows in the HSV color space using dynamic thresholds,” Int. J. Electr. Comput. Eng., vol. 8, no. 3, pp. 1513–1521, 2018.
[5] S. Liu, L. Li, J. Tang, S. Wu, and J.-L. Gaudiot, Creating Autonomous Vehicle Systems. Morgan&Claypool, 2018.
[6] D. Putra, Pengolahan Citra Digital. Yogyakarta: Andi, 2010.
[7] W. Budiharto, Machine Learning and Computational Intelligence. Yogyakarta: Andi, 2016.