

# Street Mark Detection Using Raspberry PI for Self-driving System

Sumardi\*, Muhammad Taufiqurrahman, Munawar A Riyadi

Department of Electrical Engineering Diponegoro University  
Jln. Prof. Sudharto, Tembalang, Semarang, Indonesia

\*Corresponding author, e-mail: sumardi.undip@gmail.com

## Abstract

*Self driving is an autonomous vehicle that can follow the road with less human intervention. The development of self driving utilizes various methods such as radar, lidar, GPS, camera, or combination of them. In this research, street mark detection system was designed using webcam and raspberry-pi mini computer for processing the image. The image was processed by HSV color filtering method. The processing rate of this algorithm was 137.98 ms corresponding to 7.2 FPS. The self-driving prototype was found to be working optimally for "hue" threshold of 0-179, "saturation" threshold of 0-30, and "value" threshold of 200-255. Street mark detection has been obtained from the coordinates of street mark object which had range 4-167 on x axis and 4-139 on y axis. As a result, we have successfully built the street mark detection by COG method more effectively and smoothly in detection in comparison with Hough transform method.*

**Keywords:** Street mark detection, HSV color filtering, COG, Self-driving car.

**Copyright © 2018 Universitas Ahmad Dahlan. All rights reserved**

## 1. Introduction

Self-driving is one type of car control that enables to drive with less human intervention. Self driving is helpful, for example when the driver suffers certain conditions that need to relinquish the steer, for injuries or fainting etc. The development of self-driving research is predicted to be complete and ready to be implemented in 2023 [1]. Development of the research on autonomous driving car has been done in many ways. The researches in self driving were conducted in various focus either in actual realization [2] or prototype scaled [3]. Several methods for collision avoidance applied lidar or camera [2],[4], while it could also included mini computer like raspberry pi which can process the image [4]. Moreover, there are various image processing for pattern recognition method like in filtering image, hough method, and determining the tracking references [5-7]. In terms of driving guidance, several methods has been developed, including the use of marked and unmarked lanes [8-11].

The usage of street mark for guidance is popular, e.g in [10],[12]. The road mark detection in [12] was using a camera processing images. They used Intel processor T5750 2.0 GHz clock speed that has processing time below 14 ms for single processing. However, they faced obstacles in determining the boundary line in unfavorable turn conditions. Although it could be overcome by applying either a spline or set of line to approximate lane border, but the turning road was hard to detect because they used Hough transform. The usage of a mini computer for image processing is preferred for effective dimension, cost, and performance according to Ujainiya et al [4].

To that extend, we conduct research on self-driving which can detect lane of the road or the street mark. We propose a design of 1:10 scale prototype of self-driving car with image processing from camera in raspberry pi 2. The purpose from this research is to deploy street mark detection method for self-driving sistem in the prototype. It determines the coordinate by COG (Center of Gravity) method of detection area in filtered image result acquired by single camera as was used in [12]. This paper does not pay attention to the illumination effect [7], but only specify the filter parameters with HSV method to detect street marks.

## 2. Research Method

The street mark detection system design is based on the specification for self-driving system. The self-driving system requires street mark detection system that uses a vision sensor for autonomous car tracking system. Specifications to be achieved by prototype of self driving system, namely:

- a. using small dimension hardwares as a prototype with the 1:10 scale car and the maximum weight of 3 kg.
- b. able to see and capture the image from the camera in real time and can be repeated continuously.
- c. able to detect the street mark as white color around the black track.
- d. able to detect coordinates of street mark from the filtered images.

This project is built using mini computer raspberry pi 2 that has compact size, light weight and a 1GB RAM, quad-core processor. This computer is capable to process the image to detect the street mark. To detect street mark, raspberry pi 2 as mini computer must have a particular model of image processing that can perform color filtering. This project uses OpenCV as an image processing library to run HSV color filtering method. The prototype of designed autonomous car is shown in Figure 1.



Figure 1. Autonomous car prototype

Raspberry pi should be capable to take pictures from camera, run the library OpenCV for color filtering process, perform coordinate calculation and send data through serial. These tasks are done automatically and repeatedly using python programming. OpenCV is an image processing library that can perform color filtering process. The method used is HSV color filtering based on RGB color space, which is simpler than CMYK. HSV color filtering method has a color space that is mapped with 3 components: Hue, saturation, and value. The threshold is determined on the color space to detect the street mark of the surrounding environment.

Street mark detection system was developed using python programming language on a mini computer raspberry pi. The algorithm and flowchart for detection system is shown in Figure 2, consisting of the following tasks:

- a. Take pictures
- b. Separate components of HSV
- c. Thresholding
- d. Total the results thresholding HSV color space
- e. object detection
- f. Calculate the coordinates of the object
- g. Send the data via the serial

Figure 3 shows five windows for HSV color filtering in street mark detection that are Hue Filter as shown in Figure 3.a, Saturation Filter as shown in Figure 3.b, Value Filter as shown in Figure 3.c which feature trackbar to set variable threshold minimum and maximum of each component filtering. The windows also show the results of each filter for hue, saturation, or value. Window in Figure 3.d shows results from the amount of the three processes hue,

saturation, and value to the results of the final filter that determines the HSV object is detected or not.

Once the object can be detected from HSV filter, then it performs the calculation of the coordinates of the detected object. Coordinate data then will be sent to the another controller (microcontroller) via serial communication. Figure 3.e shows the result of street mark tracked from the original image which was taken by the camera with the addition of a square shape at the origin of the object being detected in calculated coordinates.

The car prototype was designed with camera which have window view on yellow area as in Figure 4. The prototype has blank spot in front of it, up to 23.3 cm and has 40 degrees of horizontal visibility. According to [11] [12], the captured image is obtained from light reflection to camera lens that is proportional with the distance. Therefore, the point coordinates is detected by obtaining the center of detection area with geometry method. The coordinate point (x,y) is obtained by calculating the center of gravity in the object detections with  $X0=M10/M00$  and  $y0=M01/M00$  [13]. Some results of the object detection and COG point are shown in Figure 5. The image is streamed directly, and the COG is simultaneously calculated. As a result, the self driving car can adjust the coordinate of street to follow it.

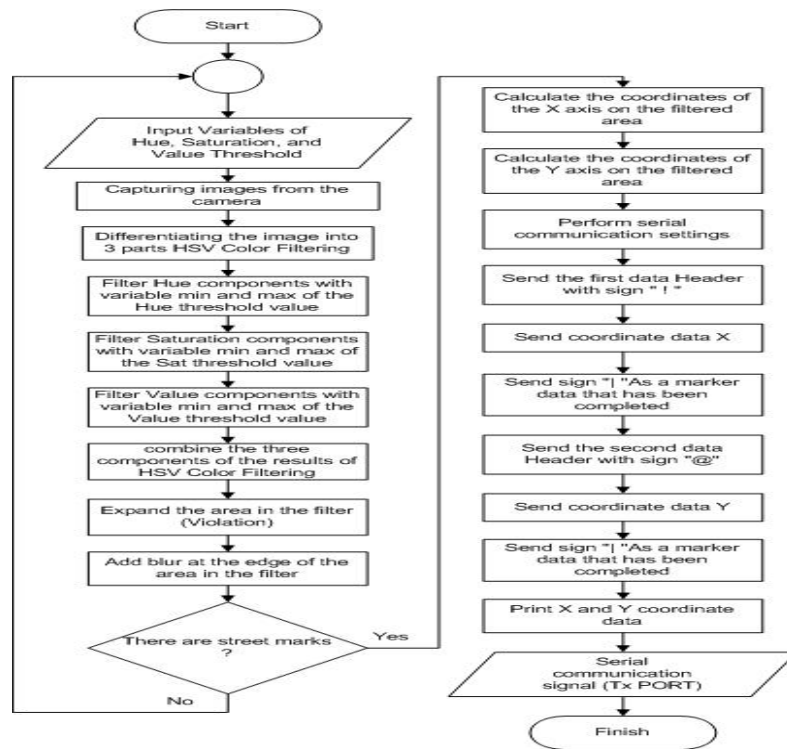


Figure 2. Flowchart of street mark detection system

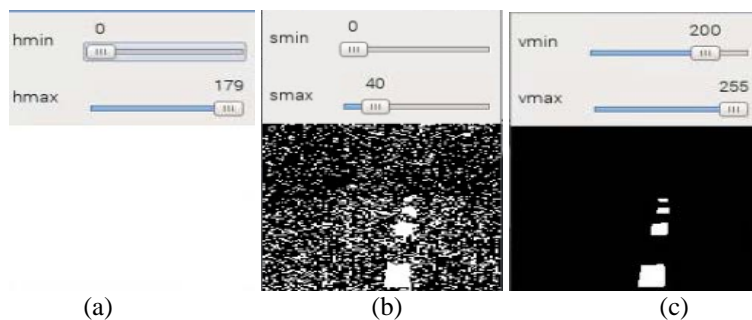


Figure 3. The interface of color filtering process in street mark detection System

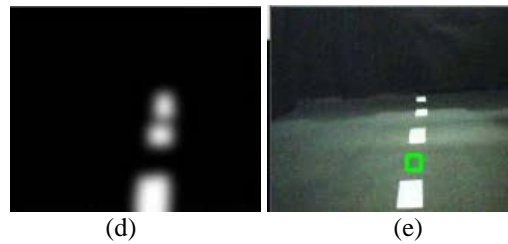


Figure 3. The interface of color filtering process in street mark detection System

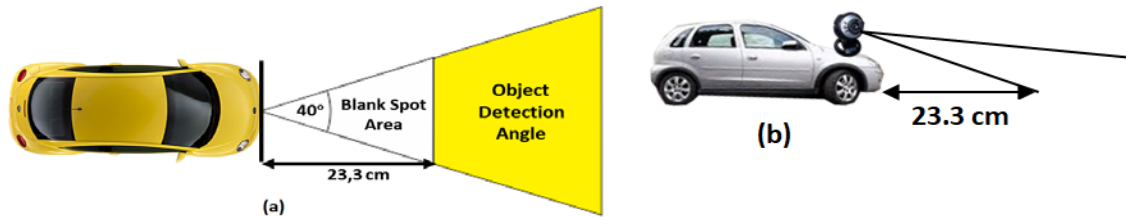


Figure 4. Detection areas in autonomous car prototype: (a) top view, (b) side view

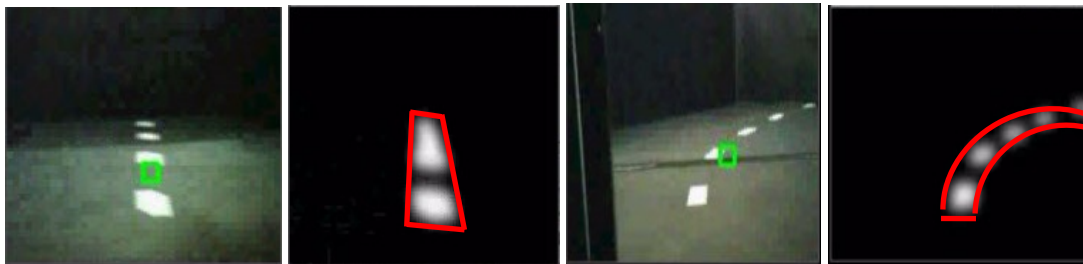


Figure 5. Detection areas and COG coordinates

### 3. Results and Analysis

#### 3.1. Color Filtering Test

The digital image processing was built on Raspberry pi 2 based on python programming language and OpenCV library. To obtain the capabilities of color filter process, testing of color filtering was performed. The test was purported to detect the presence of white street mark from black background. The testing was done by calibrating the threshold of color filtering component (i.e hue, saturation, and value) in the image color space to the HSV color distribution method. This test also pay attention to the influence of the intensity of light or illumination provided at the time of testing color filtering. Illumination received by the camera affects the outcome of images that can change the color space coordinates so that the threshold filtering colors used might not be correct.

The image data is based on the input camera and color-filtered with HSV method. Trackbar is designed to be used in the calibration process of threshold filtering components of hue, saturation, and value. Data width of 8 bits was applied with value of 0 as minimum value and value of 255 as the maximum value that represents the coordinates based on the hue, saturation, and value of a color. In the OpenCV library, saturation and value is represented by 0 as the minimum and 255 as the maximum. The maximum value of saturation is the clear colors of red, green, or blue. Conversely, for smaller saturation value, the color is faded to white of the original color of red, green, or blue. Meanwhile, for the greater detected color value, the colors are bright, while small value turn to black.

The filter components of hue represent the coordinates of the original color of red, green, or blue to blend them in accordance with the 3D color space diagram. A mix of red,

green, and blue (RGB) values are represented in degree circle of 0-360 degrees. The range of hue value is represented by the value 0 as a minimum value and a maximum value 179. Trackbar is used in process color filtering. The variable of threshold hue filtering has a range of values from 0-179. The variable of saturation filtering has a range value of 0-255. Filtering and variable threshold value has a value range of 0-255.

#### a. Hue Filtering

The testing process is done by changing the hue filtering threshold value on the trackbar. Testing is done by finding the threshold value of the minimum to maximum and from maximum to minimum.

Table 1. The Result of Hue Threshold Calibration

Calibration	Min.Value	Max. Value	White Object	Black Object
Minimal to maximal	0	74	Not Detected	Detected
Maximal to minimal	78	179	Detected	Not Detected

Table 1 reveals the threshold filtering component for color hue. It can be seen that the detection of black and white in hue component is not significant. This is because black and white are not basic colors, but a mixture of red, green, or blue. Black and white are valid on the value of any hue.

#### b. Saturation Filtering

Saturation testing process filtering is done by changing the threshold value on the trackbar. Testing is done by finding the threshold value of the minimum to maximum and from maximum to minimum. is input from a video camera in real time on the testing process filter saturation. From the testing of threshold filtering for component color saturation as seen in Table 2, the test shows that images of black color can be detected at maximum saturation threshold values and white color detection with minimal saturation threshold value.

Table 2. The Result of Saturation Threshold Calibration

Calibration	Saturation (min)	Saturation (max)	White Object	Black Object
Minimal to maximal	0	30	Detected	Not Detected
Maximal to Minimal	25	255	Not Detected	Detected

#### c. Value Filtering

The testing process value filtering is done by changing the threshold value on the trackbar. Testing is done by finding the threshold value of the minimum to maximum and from maximum to minimum. From the testing of threshold value for component color filtering as shown in Table 3, it is found that black color can be detected at a minimum value threshold value and the white color detection threshold value maximum value.

Table 3. The Result of Value Threshold Calibration

Calibration	Value of min	Value of max	White Object	Black Object
Minimal to maximal	0	203	Not Detected	Detected
Maximal to minimal	200	255	Detected	Not Detected

### 3.2. Test of object coordinate detection

Test of object coordinate detection is performed to check the ability of the system in detecting the presence of objects in the form of street mark based on the coordinates of the camera image filter. Coordinates are obtained from central point on the object of filtered area marked by a green square shape. Coordinate values of the x-axis and y-axis is 0 or minimal in the upper left corner in the image received by the camera. The test data corresponding to coordinate detection is shown in Table 4, while the reference coordinate axes x and y axis (0,0)

is the upper left pixel of the image. This is the image modeled into a matrix of pixels possessed. The resolution used is 240x240 so that the maximum value of the x-axis is 240. In testing the value of the x-axis and y-axis maximum value is only about 160 for detecting a form of data taken is the center of the object. While the process of detection of the pixel area is limited to a minimum value to avoid detection of noise.

Table 4. The Result of coordinates object detection calibration

Calibration	X Value	Y Value	X position	Y position
1	94	86	Center	Center
2	167	70	Right	Center
3	7	73	Left	Center
4	98	4	Center	Top
5	89	139	Center	Bottom
6	4	6	Left	Top
7	166	129	Right	Bottom

#### 4. Conclusion

The street mark detection has been successfully built in Raspberry-pi 2 and can detect street marks and determining the tracking coordinate with COG method. Processing rate of this algorithm which runs on 900MHz quad-core ARM Cortex A7 is 136.48 ms or 7.2 FPS. The results of HSV threshold in calibration process is hue max=179, hue min=0, saturation max=30, saturation min=0, Value max=255, and the value min=200. The threshold value can detect a street marked with the coordinates with a range of 4-167 on the x axis and 4-139 on the y-axis. Set point of coordinates were obtained with calculation of COG point of filtered detected area in (X,Y). The street mark detection can provide prediction of the street mark in front either in straight lane or in turning lane.

#### References

- [1] S. Devitt, S. Flannery, G. Locraft, A. Wood, K. Weiss, and A. Schenker. *Autonomous Cars Self-Driving the New Auto Industry Paradigm*, Morgan Stanley, 2013:1-109
- [2] H. Cho, Y. Seo, B. V. K. V. Kumar, and R. R. Rajkumar. *A Multi-Sensor Fusion System for Moving Object Detection and Tracking in Urban Driving Environments*, IEEE International Conference on Robotics & Automation, Hongkong, 2014: 1836–1843.
- [3] Mohamed, Iqbal. *Self-driving Lego Mindstorms Robot*, Proc. Python in science Conf. (SCIPY), 2012.
- [4] Ujjainiya, Lohit, M. K. Chakravarthi. *Raspberry-Pi Based Cost Effective Vehicle Collision Avoidance System Using Image Processing*, *ARPJ. Eng. Appl. Sci*, 2015; 10(7)
- [5] P. Zhao, J. Chen, Y. Song, X. Tao, T. Xu, and T. Mei. *Design of a Control System for an Autonomous Vehicle Based on Adaptive-PID*, *Int. J. Adv. Robotic Syst.*, INTECH, Rijeka, 2012;9(44)
- [6] Tekalp, A. Murat. *Digital video processing*. Prentice Hall Press, 2015.
- [7] J. M. Alvarez and A. M. Lopez. *Road detection based on illuminant invariance*, *IEEE Trans. Intelligent Transportation Systems*, 2010.
- [8] T. Kuhn and J. Fritsch. *Visio-spatial road boundary detection for unmarked urban and rural roads*, in *IEEE Intelligent Vehicles Symposium Proceedings*, 2014: 1251–1256.
- [9] D. Ponsa, J. Serrat, A. M. López. *On-board image-based vehicle detection and tracking*, *Trans. Inst. Meas. Control*, 2011; 33(7) 783–805
- [10] S. F. X. Bayerl, T. Luettel, H. Wuensche. *Following Dirt Roads at Night-Time: Sensors and Features for Lane Recognition and Tracking*, *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015; 117–122.
- [11] Stein, Gideon P., Ofer Mano, and A. Shashua. *Vision-based ACC with a Single Camera: Bounds on Range and Range Rate Accuracy*, *IEEE IV2003 Intelligent Vehicles Symposium*, USA, 2003.
- [12] Buczkowski, M. and Stasinski, R., *Automatic Lane Detection*, PWT 2012, Poznan, 2012
- [13] Ardeshir, A. *Image Registration: Principles, Tools and Methods*, Springer, London, 2012.