**HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY AND EDUCATION**

**FACULTY FOR HIGH QUALITY TRAINING**



**GRADUATION THESIS**

**CONSTRUCT OF LANE DEPARTURE WARNING SYSTEM USING RASPBERRY PI**

**ADVISOR: NGUYEN DANG QUANG, M.S**

**STUDENT:**

1. **NGUYEN TUAN KIET 14110348**
2. **NGUYEN QUOC TINH 14110422**

**ACADEMIC BATCH: 2014 – 2018**

**MAJOR: INFORMATION TECHNOLOGY**

Ho Chi Minh City, July 2018

 **SOCIALIST REPUBLIC OF VIETNAM**

**Independence – Freedom – Happiness**

**----\*\*\*----**

**Ho Chi Minh City, March 23rd 2018**

**MISSION OF THESIS**

Student’s name: Nguyen Tuan Kiet 14110348

Nguyen Quoc Tinh 14110422

Major: Information Technology Class: 14110CL1

Advisor: NGUYEN DANG QUANG, M.S Phone number: 01662218622

Delivery date: 12/03/2018 Submission date: 20/06/2018

1. **Title of thesis:** Construct of lane departure warning system using Raspberry Pi.
2. **Main content:**

Fundamental

* Introduce about Raspberry Pi 3 and accessories.
* How to work with the devices.
* Introduction to OpenCV library.
* Image processing Basics, Canny edge detection algorithm, Hough transform method.

Construct of lane departure warning system using Raspberry Pi 3

* Using OpenCV library, Image processing Basics, Canny edge detection algorithm, Hough transform method to detect lane.
* Using blink lights to warning the driver.

1. **Product:**

* A device which uses Raspberry Pi 3 capable of detecting lane markings on the road surface.
* Warning the driver of unintended lane departures.

|  |  |
| --- | --- |
| Program Chair | Advisor |

 **SOCIALIST REPUBLIC OF VIETNAM**

**Independence – Freedom – Happiness**

**----\*\*\*----**

**ADVISOR’S COMMENT**

Student’s name: Nguyen Tuan Kiet 14110348

Nguyen Quoc Tinh 14110422

Major: Information Technology

Title of thesis: Construct of lane departure warning system using Raspberry Pi.

Advisor’s name: NGUYEN DANG QUANG, M.S.

**COMMENTS**

1. The content of thesis and the workload allocated:

1. Advantages:

1. Improvement point:

1. Project defense approval:

1. Evaluation mark:

1. Grade: (in words: )

|  |  |
| --- | --- |
|  | Ho Chi Minh City, June 21, 2018  Advisor  (*Name & signature*) |

 **SOCIALIST REPUBLIC OF VIETNAM**

**Independence – Freedom – Happiness**

**----\*\*\*----**

**REVIEWER’S COMMENT**

Student’s name: Nguyen Tuan Kiet 14110348

Nguyen Quoc Tinh 14110422

Major: Information Technology

Title of thesis: Construct of lane departure warning system using Raspberry Pi.

Reviewer’s name: TRAN TIEN DUC, M.S.

**COMMENTS**

1. The content of thesis and the workload allocated:

1. Advantages:

1. Improvement point:

1. Project defense approval:

1. Evaluation mark:

1. Grade: (in words: )

|  |  |
| --- | --- |
|  | Ho Chi Minh City, June 20, 2018  Reviewer  (*Name & signature*) |

**TABLE OF CONTENTS**

[ACKNOWLEDGMENT i](#_Toc518583042)

[ABSTRACT ii](#_Toc518583043)

[LIST OF ABBREVIATIONS iii](#_Toc518583044)

[LIST OF FIGURES iv](#_Toc518583045)

[Chapter 1: INTRODUCTION 1](#_Toc518583046)

[1.1. Statement of the Problem 1](#_Toc518583047)

[1.2. Purpose of the Study 1](#_Toc518583048)

[1.3. Background of the Study 2](#_Toc518583049)

[1.4. Objectives of the Study 4](#_Toc518583050)

[1.5. The scope of the Study 5](#_Toc518583051)

[1.6. Methodology 5](#_Toc518583052)

[1.7. Structure of the Thesis 5](#_Toc518583053)

[Chapter 2: THEORY 6](#_Toc518583054)

[2.1. Image processing Basics 6](#_Toc518583055)

[2.1.1. Pixels 6](#_Toc518583056)

[2.1.2. RGB color space and grayscale 7](#_Toc518583057)

[2.1.3. Filtering 8](#_Toc518583058)

[2.2. Canny Edge detection 11](#_Toc518583059)

[2.3. Hough transform method 15](#_Toc518583060)

[2.4. Raspberry Pi 3 and Raspbian 19](#_Toc518583061)

[2.4.1. Raspberry Pi 3 19](#_Toc518583062)

[2.4.2. Raspbian 21](#_Toc518583063)

[Chapter 3: METHOD 22](#_Toc518583064)

[3.1. Model outline 22](#_Toc518583065)

[3.2. Preprocessing 22](#_Toc518583066)

[3.2.1. Convert image to its grayscale image 23](#_Toc518583067)

[3.2.2. Region of Interest (ROI)  Initialization 24](#_Toc518583068)

[3.3. Lane marking detection 25](#_Toc518583069)

[3.3.1. Color selection 25](#_Toc518583070)

[3.3.2. Canny edge detection 26](#_Toc518583071)

[3.4. Line detector 26](#_Toc518583072)

[3.5. Control LEDs with Raspberry Pi 3 28](#_Toc518583073)

[Chapter 4: LANE DEPARTURE WARNING SYSTEM 29](#_Toc518583074)

[4.1. Lane mark parameters 29](#_Toc518583075)

[4.2. Detection Lane Departure 30](#_Toc518583076)

[4.3. Experimental results 31](#_Toc518583077)

[Chapter 5: CONCLUSION 32](#_Toc518583078)

[5.1. Conclusion: 32](#_Toc518583079)

[5.2. Development: 32](#_Toc518583080)

[REFERENCES 33](#_Toc518583081)

ACKNOWLEDGMENT

Firstly of all, we would like to sincerely thank you for the teachers of the Ho Chi Minh City University of Technology and Education provided me with useful knowledge to complete the graduation thesis.

Secondly, we would like to express my sincere gratitude to my advisor M.S. Nguyen Dang Quang for the continuous support of my thesis, for this patience, motivation, and immense knowledge. His guidance helped us in all the time of research and writing of this thesis. We would not have imagined having a better advisor and mentor.

Thirdly, we would like to thank my reviewer M.S. Tran Tien Duc for his insightful comments and encouragement, but also for the hard question which incented me to improve our thesis from various perspectives.

Last but not the least, we would like to thank our family: our parents and to our brothers and sisters for supporting spiritually throughout writing this thesis.

We wish you good health to all!

Nguyen Tuan Kiet.

Nguyen Quoc Tinh.

ABSTRACT

The main purpose of this thesis is to develop a Lane Departure Warning System (LDWS). As part of Automatic Driver Assistance System (ADAS), LDWS exists to alert the driver when the car is diverting from its original lane. The LDWS requires 3 major operations. First, the captured image is divided into two parts as a road part and a non -road part by using the camera geometry information. Then, the inverse perspective mapping is applied to avoid the disadvantage of perspective effect. Next, a gradient method is used to filter lane marks and Canny edge detection is applied. Additionally, Hough transform method is used for lane marks detection. Finally, the driver is warned according to right or left lane departure by using detected lane marks’ angles.

Raspberry Pi is the primary platform that this system will be implemented on. It has various Input/Output (I/O) ports that allow the developer to utilize to connect peripheral-device and many modules and component have been designed for it, such as the Camera Module which will be used for a various purpose such as capturing live images of our system.

There are several problems which need to be solved by the system. First of all, the road condition such as weather and low light condition that will cause the system not able to detect the lane should be addressed with several image processing technique. Other than that, this system should be portable and compact enough to be installed on the rear of a windshield mirror and implemented on a vehicle.

LIST OF ABBREVIATIONS

LDWS Lane Departure Warning System

ADAS Automatic Driver Assistance System

NHTSA National Highway Traffic Safety Administration

IIHS Insurance Institute for the Safety of the United States Highway

LIST OF FIGURES

[Figure 1.1: Lane Departure Warning 2](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583093)

[Figure 1.2: Technology Lane alert on Chevrolet 4](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583094)

[Figure 2.1: Pixels 6](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583095)

[Figure 2.2: RGB color space 7](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583096)

[Figure 2.3: Remove noise and detail from the image with Gaussian filtering 10](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583097)

[Figure 2.4: Remove salt and pepper noise from the image with median filtering 11](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583098)

[Figure 2.5: Original 12](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583099)

[Figure 2.6: Smoothed Smoothed 12](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583100)

[Figure 2.7: Gradient magnitudes 13](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583101)

[Figure 2.8: Non-maximum suppression. Edge-pixels are only preserved where the gradient has local maxima 14](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583102)

[Figure 2.9: Double Thresholding 14](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583103)

[Figure 2.10: Edge tracking by hysteresis 15](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583104)

[Figure 2.11: Final edges 15](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583105)

[Figure 2.12: Lines through a point in the Cartesian domain 16](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583106)

[Figure 2.13: A straight line in (m,c) domain 17](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583107)

[Figure 2.14: Points on the same line 17](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583108)

[Figure 2.15: The mapping of P1 and P2 from Cartesian space to the (m, c) space. 18](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583109)

[Figure 2.16: The representation of a line in the (x, y) space using (ri, θi ) 18](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583110)

[Figure 2.17: Five point in Cartesian coordinate system 19](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583111)

[Figure 2.18: Their sinusoid in Polar coordinate system 19](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583112)

[Figure 2.19: Raspberry Pi 3 20](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583113)

[Figure 2.20: Raspbian OS 21](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583114)

[Figure 3.1: Flowchart of the model outline 22](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583115)

[Figure 3.2: Convert RGB image to Grayscale image 23](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583116)

[Figure 3.3: Image smoothing 24](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583117)

[Figure 3.4: ROI Initialization 24](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583118)

[Figure 3.5: Histogram of image 25](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583119)

[Figure 3.6: Select white lines in image 25](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583120)

[Figure 3.7: Canny edge detection 26](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583121)

[Figure 3.8: Images after Hough Transformation 27](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583122)

[Figure 3.9: Detected lane marks in input images 27](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583123)

[Figure 3.10: Connect LED with Raspberry Pi 3 28](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583124)

[Figure 4.1: Ideal case of lane departure 29](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583125)

[Figure 4.2: Left lane departure 29](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583126)

[Figure 4.3: Right lane departure 30](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583127)

[Figure 4.4: Graphical representation of lane departure cases, a) ideal case, b) left lane departure, c) right lane departure. 30](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583128)

[Figure 4.5: Sample images of the lane detection. 31](file:///C:\Users\quoct\Google%20Drive\DATN\DATN\Docs\LaneDepartureWarningDoc_2.5.docx#_Toc518583129)

# INTRODUCTION

## Statement of the Problem

The growing number of vehicles on road and consequently, the rising traffic accidents have been noticing recently all over the world. According to the World Health Organization (WHO), about 1.2 million people get killed in traffic accidents each year worldwide, while the number of injured is estimated to be between 20 and 50 million. Most of these accidents occur due to driver inattention.

One of the most developed trends of research in automotive solution is passenger's safety. Thus, public research groups, automotive manufacturers, and suppliers, as well as other research institutions are developing the next generation of the driver – assistant system that enables vehicles to have safer reactions and to decrease road injuries and deaths.

Driver warning systems which make the driver realize a dangerous situation are one of the functions of the driver – assistant. The driver is warned against potential hazards which are determined by various sensor systems such as radar and camera.

The operation of the system can be split into 3 phases. Firstly, the sense and perception of the World are observed. Secondly, the decision on what to do is made based on real-time perception. And finally, the action will be carried out based on the decision made. Among those, the perception and land detection a the most challenging.

The system in this thesis is about lane detection and lane departure warning system.

## Purpose of the Study

When people drive, they use their eyes to figure out how fast to go, where the lane lines are and where to turn. A car doesn’t have eyes. But, in a self - driving car, we can use cameras and other sensors to achieve a similar function. So, let’s think about what those cameras are seeing as it is driven down the road. The human can easily see where the lane lines are but for a self-driven car, it must be taught to do that.

People can find lane lines on the road fairly easily, even in a wide variety of conditions. Unless there is snow covering the ground, extremely heavy rainfall, the road is very dirty or in disrepair, we can mostly tell where we are supposed to go, assuming the lines are actually marked. But, computers, on the other hand, do not find this easy. Shadows, glare, small changes in the color of the road, slight obstruction of the line…all things that people can generally handle, but a computer may struggle mightily with.

The purpose of the study is how to solve that problem and warn the driver of unintended lane departures. OpenCV and Canny Edge detection is one of the most popular computer vision techniques to serve it.

At the end of this thesis, we aim to have a device which uses Raspberry Pi 3 and Raspberry Pi Camera Module capable of:

1. Detecting lane markings on the road surface with various weather conditions and different road types.
2. Warning the driver of unintended lane departures.

## Background of the Study

In road-transport terminology, a lane departure warning system is a mechanism designed to warn the driver when the vehicle begins to move out of its lane on freeways and arterial roads. These systems are designed to minimize accidents by addressing the main causes of collisions: driver error, distractions, and drowsiness. In 2009 the U.S. National Highway Traffic Safety Administration (NHTSA) began studying whether to mandate lane departure warning systems and frontal collision warning systems on automobiles.



Figure .: Lane Departure Warning

(Source: https://xehay.vn/uploads/images/2017/8/04/xehay-Lane-Departure-Warning-250817-1.jpg)

There are two main types of system:

* The system which warns the driver (Lane departure warning, LDW) if the vehicle is leaving its lane (visual, audible, and/or vibration warnings).
* The system which warns the driver and, if no action is taken, automatically takes steps to ensure the vehicle stay in its lane (lane keeping system, LKS).

The first production lane departure warning system in Europe was developed by the United States company Iteris for Mercedes Actros commercial trucks. The system debuted in 2000 and is now available on most trucks sold in Europe.

In 2002, the Iteris system became available on Freightliner Trucks North American vehicle. In both these system, the driver is warned of unintentional lane departures by an audible rumble strip sound generated on the side of the vehicle drifting out of the lane. No warnings are generated if, before crossing the lane, an active turn signal is given by the driver.

*Abroad:*

The lane departure warning system is one of the most expensive safety features on modern cars. According to the Insurance Institute for the Safety of the United States Highway (IIHS), this feature helps prevent up to 85,000 accidents each year in this country.

IIHS deputy director of research, Jessica Cicchino, recently conducted a study on the effectiveness of the lane departure warning system by analyzing crash date from police reports. And the recorded the numbers are notable.

Research has shown active safety system cars, the rate of self-inflicted injury, accidental stabbing from the hip or craniofacial injury was reduce to 11%. In addition, it reduced the incidence of injuries in accidents by 21%. And most notably, the lane warning system has helped prevent 85,000 accidents and at least 55,000 injuries in 2015.

According to Cicchino, these are the first examples that show us, Lane departure warning or active safety systems are important and they help prevent serious accidents and help save more lives.

According to previous studies, the lane warning system is believed to reduce up to 50% of accidents. However, IIHS argues that the bottom line is that failure to achieve this number is because many US drivers have the habit of turning off the lane warning system, though they know it is a safety feature.

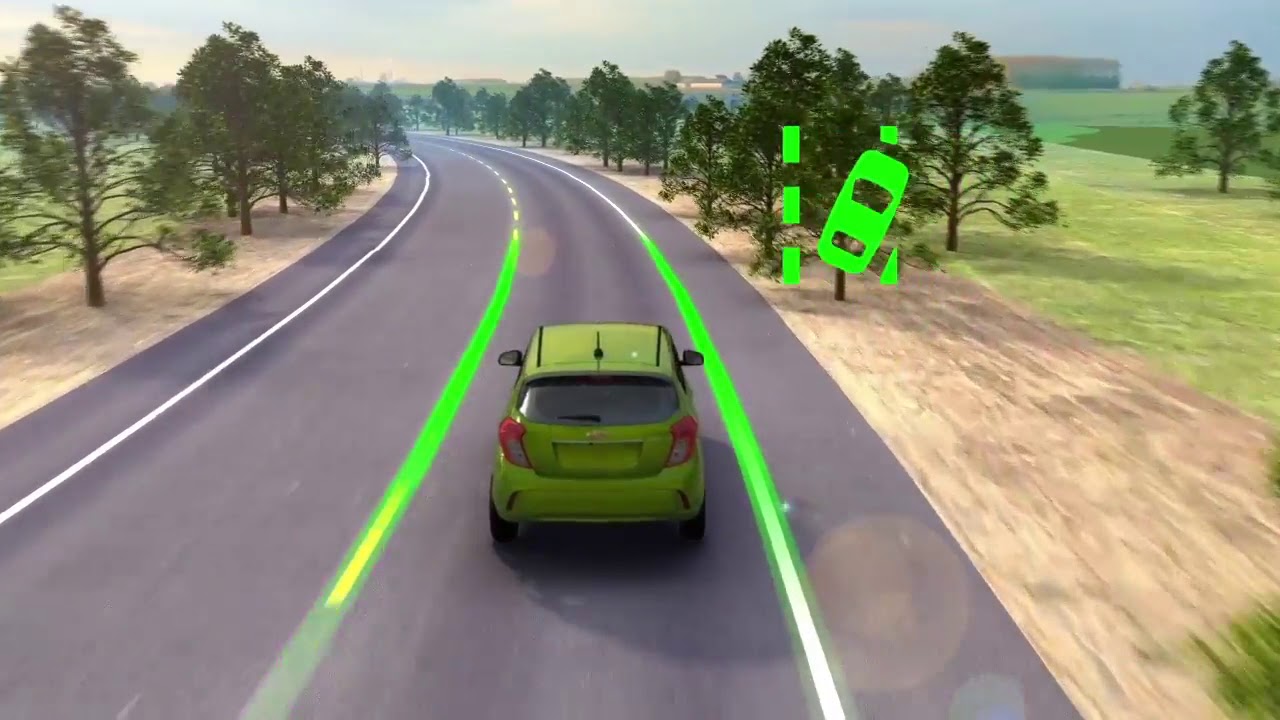


Figure .: Technology Lane alert on Chevrolet

(Source: https://i.ytimg.com/vi/7rfa1rEWLQ8/maxresdefault.jpg)

## Objectives of the Study

To complete this study we have four main objectives:

1. Learn about Raspberry Pi 3.
2. Learn about image processing basic, Canny Edge detection, Hough transform method, and OpenCV library.
3. Detecting lane markings on the road surface base on OpenCV, Canny Edge detection and Hough transform method.
4. Warning driver according to right or left lane departure by using detected lane marks’ angles.

## The scope of the Study

Subject: single-board computers Raspberry Pi 3, and data capture from camera images to detect lane.

The scope of the study: Ho Chi Minh city, represented by Ho Chi Minh City University of Technology and Education.

## Methodology

Chapter 2, our team applies the methods of analyzing and synthesizing theories, studying different documents and theories, exploring topics and researches done at home and abroad. Proceed with reference and then select lane detection algorithm and Raspberry Pi apply to the project.

## Structure of the Thesis

Chapter 1: Introduction

Chapter 2: Theory

Chapter 3: Method

Chapter 4: Lane departure warning system

Chapter 5: Conclusion

# THEORY

In this chapter, a thorough explanation of theory needed to understand this thesis will be presented. To start off, some basic digital image processing concepts vital to this work is described. After that more purpose-specific theory such as Canny Edge detection and Hough transform method.

## Image processing Basics

### Pixels

Short for **Pic**ture **El**ement, a pixel is a single point in a graphic image. Graphics monitors display pictures by dividing the display screen into thousands (or millions) of pixels, arranged in rows and columns. The pixels are so close together that they appear connected [1].

Each pixel represents a small part of the image with a solid color. To extract this information it is common standard to divide the information into three color levels: red, green and blue. For each color, the amount of that color in the pixel is described as a number. This means in practice that each frame can be represented by a [m; n; 3] matrix, where m is the width and n is the height of the image in pixels, and each of the layers in the third dimension represents the colors.

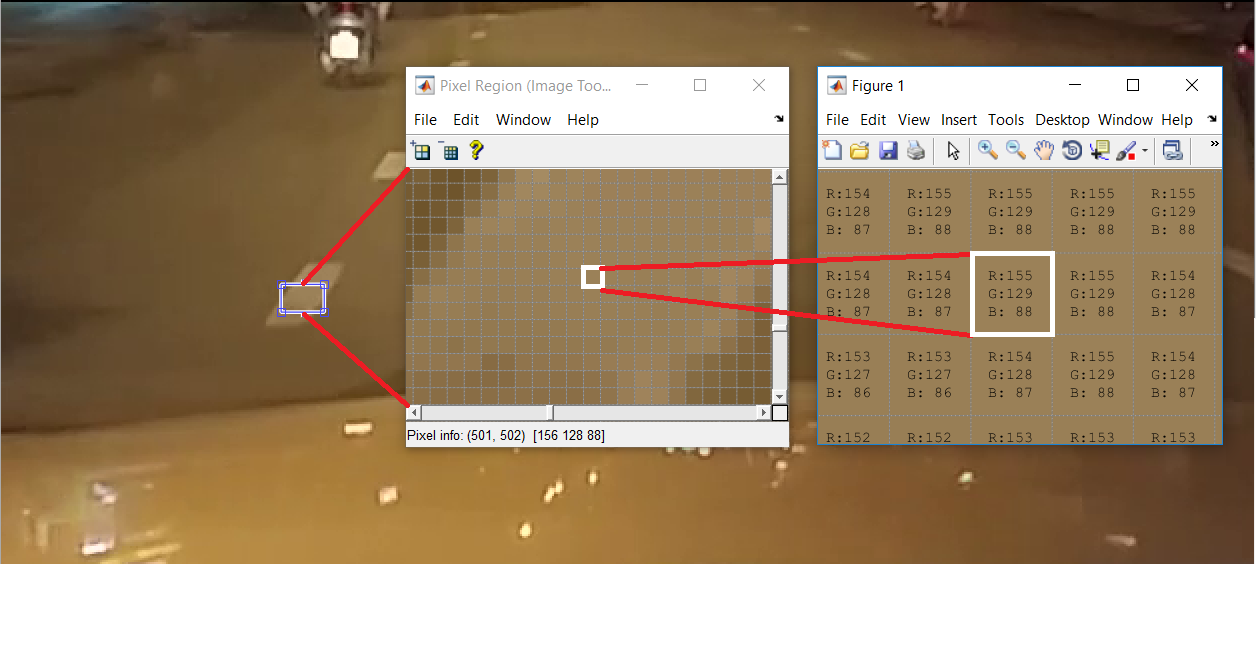


Figure .: Pixels

### RGB color space and grayscale

Humans perceive color through wavelength-sensitive sensory cells called cones. There are three different types of cones, each with a different sensitivity to electromagnetic radiation (light) of a different wavelength. One type of cone is mainly sensitive to red light, one to green light, and one to blue light. By emitting a controlled combination of these three basic colors (red, green and blue), and hence stimulate the three types of cones at will, we are able to generate almost any perceivable color. This is the reasoning behind why color images are often stored as three separate image matrices; one storing the amount of red (R) in each pixel, one the amount of green (G) and one the amount of blue (B). We call such color images as stored in an RGB color space.

A color space is a set of colors also called gamut. The gamut of a monitor represents all the colors it is able to display. RGB color space or RGB color system construct all the colors from the combination of the Red, Green and Blue colors. The red, green and blue use 8 bits each, which have integer values from 0 to 255. This makes 256\*256\*256 = 16777216 possible colors [2].

The possibilities for mixing the three primary colors together can be represented as a three dimensional coordinate plane with the values for R (red), G (green) and B (blue) on each axis. This coordinate plane yields a cube called the RGB color space:

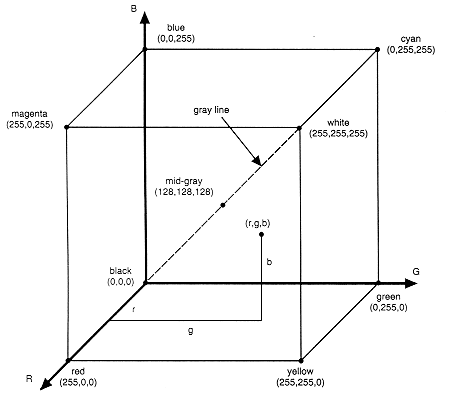


Figure .: RGB color space

(Source: http://www.viz.tamu.edu/faculty/parke/ends375f03/notes/sec1\_4.html)

In grayscale images, however, we do not differentiate how much we emit of the different colors, we emit the same amount in each channel. What we can differentiate is the total amount of emitted light for each pixel; little light gives dark pixels and much light is perceived as bright pixels.

When converting an RGB image to grayscale, we have to take the RGB values for each pixel and make as output a single value reflecting the brightness of that pixel. One such approach is to take the average of the contribution from each channel: (R+B+C)/3. However, since the perceived brightness is often dominated by the green component, a different, more “human-oriented”, a method is to take a weighted average, e.g.: Equation 2.1.

|  |  |
| --- | --- |
|  | Equation 2.1 |

Where I is intensity value of the grayscale image. R, G, and B are the intensity value of each pixel.

### Filtering

Images are often corrupted by impulse noise due to transmission errors, malfunctioning pixel elements in the camera sensors, faulty memory locations, and timing errors in analog-to-digital conversion. In most applications, denoising the image is fundamental to subsequent image processing operations. The goal of noise removal is to suppress the noise while preserving image details. A variety of techniques have been proposed to remove impulse noise. Noise is perturbations of the pixel values. Noise arises in the sensor or the imaging process. Noise may degrade the visual interpretation. Noise may be removed by filtering. Of course, noise cannot be perfectly removed. Noise removal reconstructs the correct pixel values. Generic filters such as the mean filters, order statistics filters are used to remove the noise in an image [3].

The filter which removes unwanted things. In order to that image noises, we can use the filter such as:

* Linear Image Smoothing Filters: One method to remove noise is by convolving the original image with a mask that represents a low-pass filter or smoothing operation. For example, the Gaussian mask comprises elements determined by a Gaussian function. This convolution brings the value of each pixel into closer harmony with the values of its neighbors. In general, a smoothing filter sets each pixel to the average value, or a weighted average, of itself and its nearby neighbors; the Gaussian filter is just one possible set of weights.
* Nonlinear Image Filters: A median filter is an example of a non-linear filter and, if properly designed, is very good at preserving image detail.

#### Gaussian filtering

The Gaussian filter is a filter commonly used in image processing for smoothing, reducing noise, and computing derivatives of an image. It is a convolution-based filter that uses a Gaussian matrix as its underlying kernel.

The Gaussian filter is linear, meaning it replaces each pixel by a linear combination of its neighbors (in this case with weights specified by a Gaussian matrix). It is also local, meaning it produces output pixel values based only upon the pixel values in its neighborhood as determined by the convolution kernel [4].

When working with images we need to use the two-dimensional Gaussian function. This is simply the product of two 1D Gaussian functions and is given by **Error! Reference source not found.**:

Where σ is the standard deviation of the distribution. The distribution is assumed to have a mean of 0. We need to discretize the continuous Gaussian functions to store it as discrete pixels.

The kernel of a Gaussian filter with a standard deviation of σ = 1.4 is shown in **Error! Reference source not found.**:

|  |  |
| --- | --- |
|  | Equation 2.2 |

|  |  |
| --- | --- |
|  | Equation 2.3 |

The Gaussian function is used in numerous research areas:

* It defines a probability distribution for noise or data.
* It is a smoothing operator.
* It is used in mathematics.

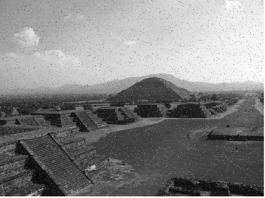


Figure .3: Remove noise and detail from the image with Gaussian filtering

#### Median filtering

Median filtering is a non-linear filtering technique that is well known for the ability to remove impulsive-type noise while preserving sharp edges. The median filter is an order statistics filter. Also, Mean filter is used to remove the impulse noise. Mean filter replaces the mean of the pixels values but it does not preserve image details. Some details are removes with the mean filter. But in the median filter, we do not replace the pixel value with the mean of neighboring pixel values, we replace with the median of those values. The median is calculated by first sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value [3].

To run a median filter:

1. Consider each pixel in the image.
2. Sort the neighboring pixels into order based upon their intensities.
3. Replace the original value of the pixel with the median value from the list.

In the median filter, the pixel value of a point p is replaced by the median of the pixel value of 8-neighborhood of a point p. The operation of this filter can be expressed as Equation 2.4.

|  |  |
| --- | --- |
|  | Equation 2.4 |

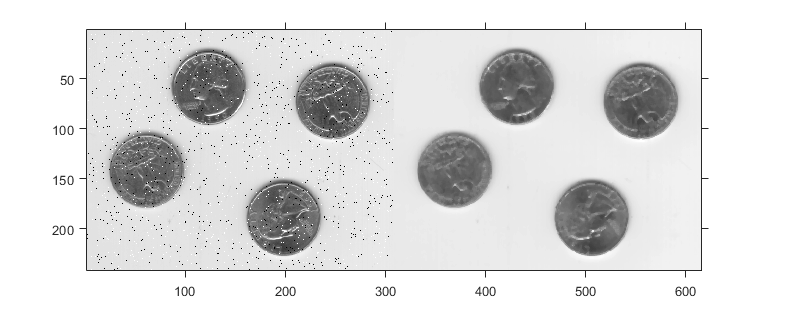


Figure .4: Remove salt and pepper noise from the image with median filtering

## Canny Edge detection

There are many methods of edge detection. However, Sobel, Roberts Cross, and Prewitt are the most used methods in the edge detection. Prewitt and Sobel are composed of horizontal and vertical directional 3x3 masks. The major limitation of these operators cannot find the edges properly in the high noise environment. Therefore, a more advanced technique was release - Canny edge detection method, developed by John F. Canny in 1986, become a standard and widely used in many types of research until nowadays.

There are multiple steps to implement the Canny Edge detector [5].

1. Smoothing

A Gaussian filter is used to smooth the image to remove noise in an image. The effect of smoothing the test image with this filter is shown in Figure 2.6 from the original grayscale image is shown in Figure 2.5.



Figure .5: Original



Figure .6: Smoothed Smoothed

1. Finding gradients

The Canny algorithm basically finds edges where the grayscale intensity of the image changes the most. Gradients at each pixel in the smoothed image are determined by applying what is known as the Sobel-operator. The first step is to approximate the gradient in the x- and y-direction respectively by applying for the kernels show in Equation 2.5.

|  |  |
| --- | --- |
|  | Equation 2.5 |

The magnitude or Edge strength of the gradient is then approximated using the formula:

|  |  |
| --- | --- |
|  | Equation 2.6 |

Where GX and GY are the gradients in the x - and y - directions respectively.

However, the edges are typically broad and thus do not indicate exactly where the edges are. To make it possible to determine this, the direction of the edges must be determined and stored as shown in Equation 2.7.

|  |  |
| --- | --- |
|  | Equation 2.7 |

The gradient magnitudes in the smoothed image shown in Figure 2.7



Figure .7: Gradient magnitudes

1. Non-maximum suppression

After getting gradient magnitude and direction, the next step - Non-maximum suppression is applied in which the algorithm removes pixels that aren’t part of an edge. The algorithm is for each pixel in the gradient image:

* Round the gradient direction 𝜃 to nearest 45°
* Compare the edge strength of the current pixel with the edge strength of the pixel in the positive and negative gradient direction.
* If the edge strength of the current pixel is largest, preserve the value of the edge strength. If not, suppress the value.

1. Double thresholding



Figure .8: Non-maximum suppression. Edge-pixels are only preserved where the gradient has local  
maxima

This step, potential edges are determined by thresholding. Canny edge detection algorithm uses double thresholding. Edge pixels stronger than the high threshold are marked as strong, edge pixels weaker than the low threshold are suppressed and edge pixels between the two thresholds are marked as weak.

In Figure 2.9, strong edges are white, while weak edges are grey. Edges with a strength below both thresholds are suppressed.



Figure .9: Double Thresholding

1. Edge tracking by hysteresis

Edge tracking can be implemented by Binary Large Object analysis (BLOB-analysis). The edge pixels are divided into connected BLOB’s using 8-connected neighborhood. BLOB’s containing at least one strong edge pixel are then preserved.

The effect of edge tracking by hysteresis on the test image is shown in Figure 2.10, strong edges in white, weak edges connected to strong edges in blue and other weak edges in red.

Final edges are determined by suppressing all edges that are not connected to a very strong edge and shown in Figure 2.11.

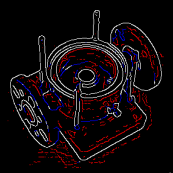


Figure .10: Edge tracking by hysteresis

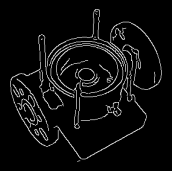


Figure .11: Final edges

## Hough transform method

Knowledge about the lines in an image is useful in many applications, e.g. in Computer vision. To manually extract the line information from an image can be very tiring and time-consuming especially if there are many lines in the image. An automatic method is preferable but is not as trivial as edge detection since one has to determine which edge point belongs to which line if any. The Hough-transform makes this separation possible and is the method we have used in our program.

The Hough transform is a feature extraction technique used in image analysis, computer vision, and digital image processing. It can be used to detect lines, circles or, other parametric curves. This technique was developed by Paul Hough in 1962 and it provides an effective approach to determining the boundaries of the object of interest [6]. It transforms between the Cartesian space and a parameter space in which a straight line (or other boundary formulation) can be defined.

The simplest case of Hough transform is detecting straight lines. Let’s consider the case where we have straight lines in an image. We first note that for every point (xi, yi) in that image, all the straight lines passing through that point satisfy Equation 2.8 for varying values of line slope and intercept (m, c), see Figure 2.12 [7].

|  |  |
| --- | --- |
|  | Equation 2.8 |

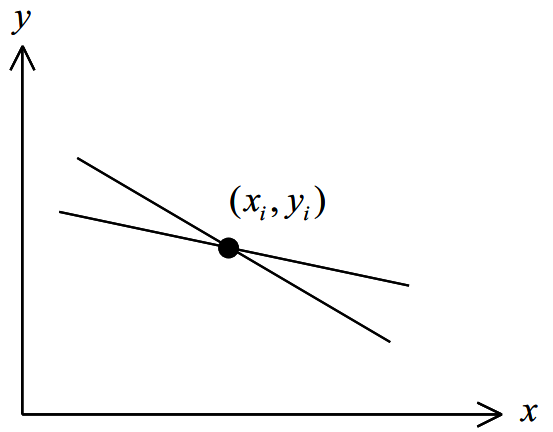


Figure .12: Lines through a point in the Cartesian domain

Now if we reverse our variables and look instead at the values of (m,c) as a function of the image point coordinates (xi, yi), then Equation 2.8 becomes:

|  |  |
| --- | --- |
|  | Equation 2.9 |

Equation 2.9 describes a straight line on a graph of c against m as shown in Figure 2.13.

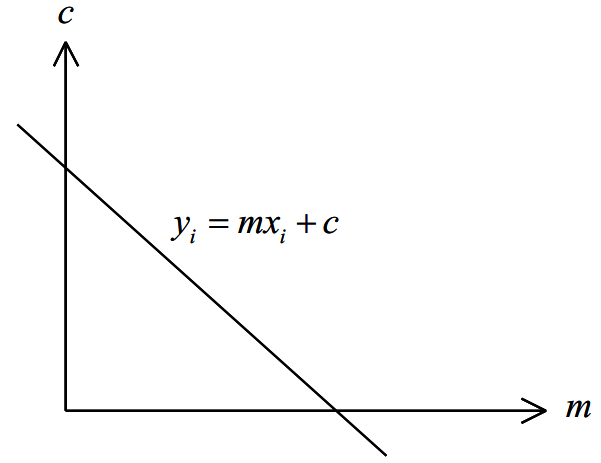


Figure .13: A straight line in (m,c) domain

At this point, it is easy to see that each different line through the point (xi, yi) corresponds to one of the points on the line in the (m, c) space.

Now, consider two pixels P1 and P2, which lie on the same line in the (x, y) space. For each pixel, we can represent all the possible lines through it by a single line in the (m, c) space. Thus a line in the (x, y) space that passes through both pixels must lie on the intersection of the two lines in the (m, c) space, which represent the two pixels. This means that all pixels which lie on the same line in the (x, y) space are represented by lines which all pass through a single point in the (m,c) space [7], see Figure 2.14 and Figure 2.15.

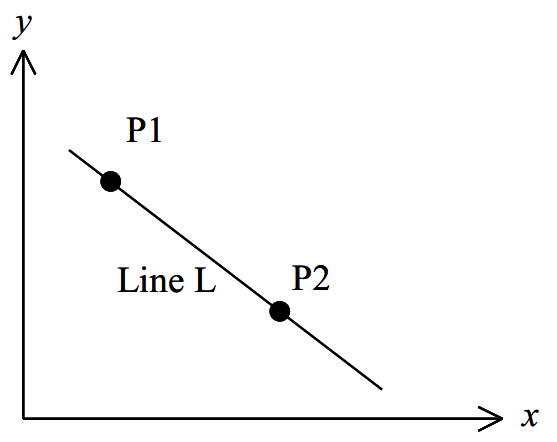


Figure .14: Points on the same line

The advantage of the Hough transform is that the pixels lying on one line need not all be contiguous. This can be very useful when trying to detect lines with short breaks in them due to noise, or when objects are partially occluded.

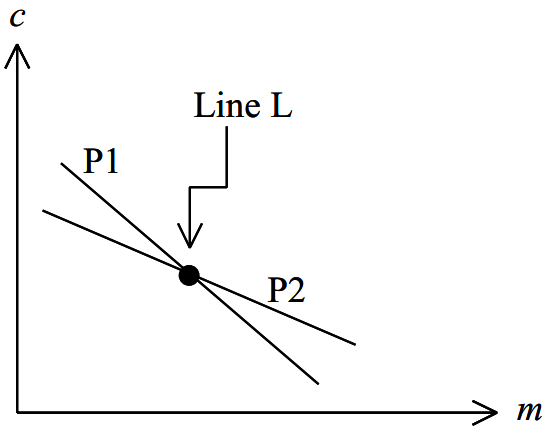


Figure .15: The mapping of P1 and P2 from Cartesian space to the (m, c) space.

As for the disadvantages of the Hough transform, one is that it can give misleading results when objects happen to be aligned by chance. This clearly shows another disadvantage which is that the detected lines are infinite lines described by their (m,c) values, rather than finite lines with defined endpoints [7].

To avoid the problem of infinite m values which occurs when vertical lines exist in the image, the alternative formulation shown in Equation 2.10 can be used to describe a line, see Figure 2.16.

|  |  |
| --- | --- |
|  | Equation 2.10 |

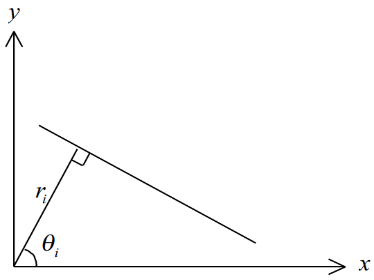


Figure .16: The representation of a line in the (x, y) space using (ri, θi )

This, however, means that a point in (x, y) space is now represented by a curve in (r, θ) space rather than a straight line. Where r is the length of a line perpendicular to this line, starting from the origin and θ is the orientation angle of ρ with respect to the x-axis. The conversion between these two systems can be easily done. Points in a Cartesian coordinate system which corresponds to a sinusoid in the Polar coordinate system is shown in Figure 2.17 and Figure 2.18 respectively.

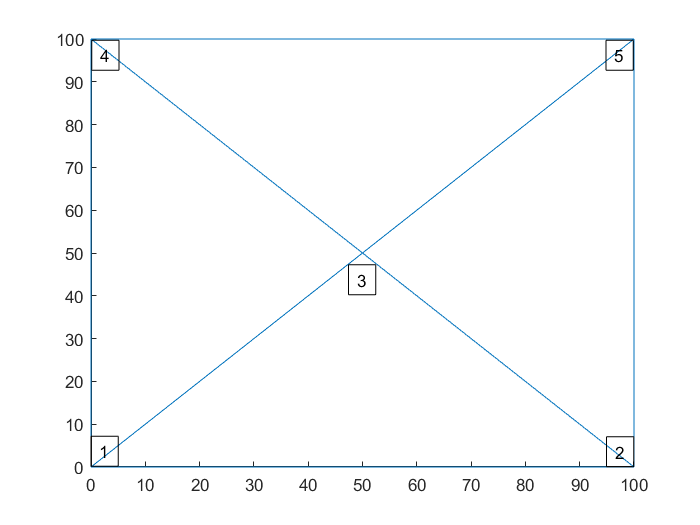


Figure .: Five point in Cartesian coordinate system

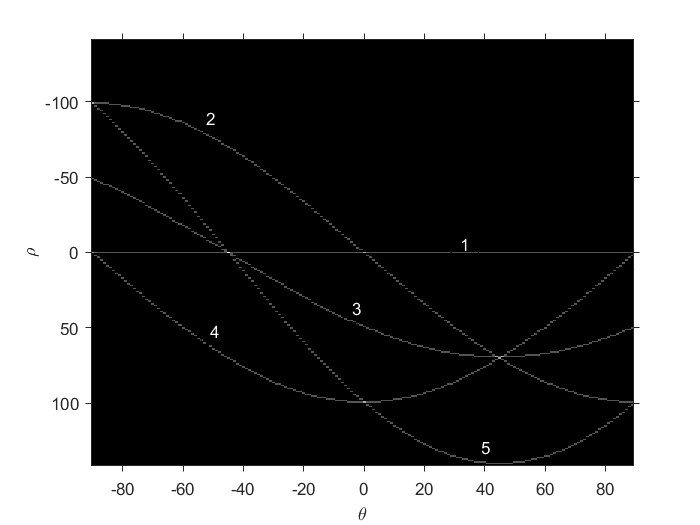


Figure .: Their sinusoid in Polar coordinate system

## Raspberry Pi 3 and Raspbian

### Raspberry Pi 3

The Raspberry Pi is a series of credit card-sized single-board computers developed in the United Kingdom by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science in schools and developing countries.

Raspberry Pi is manufactured by 3 OEMs: Sony, Qsida, Egoman. Distributed by: Newark element14 (Premier Farnell), RS Components and Egoman.

On 29 February 2012, the first generation – Raspberry Pi 1 Model B was released with 256MB of RAM, followed by the simpler and cheaper model A. After four years, Raspberry Pi 3 was released with more powerful configurations, more features, many multiple improvement support wifi and Bluetooth available on the board.

Configuration Information Raspberry Pi 3 [8]:

* Broadcom BCM2837 chipset running at 1.2 GHz
* 64-bit quad-core ARM Cortex-A53
* 802.11 b/g/n Wireless LAN
* Bluetooth 4.1 (Classic & Low Energy)
* Dual-core Videocore IV® Multimedia co-processor
* 1 GB LPDDR2 memory
* Supports all the latest ARM GNU/Linux distributions and Windows 10 IoT
* MicroUSB connector for 2.5 A power supply
* 1 x 10/100 Ethernet port
* 1 x HDMI video/audio connector
* 1 x RCA video/audio connector
* 4 x USB 2.0 ports
* 40 GPIO pins
* Chip antenna
* DSI display connector
* MicroSD card slot
* Dimensions 85 x 56 x 17 mm



Figure 2.19: Raspberry Pi 3

*(Source:https://cf1.s3.souqcdn.com/item/2016/07/04/11/04/66/79/item\_XL\_11046679\_15157672.jpg)*

### Raspbian

Raspbian is a free operating system based on Debian optimized for the Raspberry Pi hardware. Raspbian comes with over 35,000 packages: pre-compiled software bundles in a nice format for easy installation on your Raspberry Pi.

The Raspbian OS with Raspberry Pi has been updated to support 64-bit computing, with other platforms such as RetroPie and KODI must wait for updates.

Raspbian is a community project active development, with improving the stability and performance.

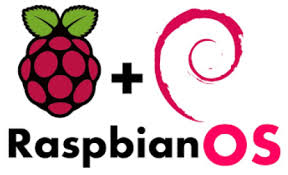


Figure 2.20: Raspbian OS

*(Source:http://misapuntesde.com/images/raspbian.png)*

# METHOD

In this chapter, the methodology of this thesis work will be described in detail. Throughout the thesis work, Python has been used, along with corresponding versions of OpenCV library and some packages.

## Model outline

The different functions of the model are implemented as separate algorithms to make the model both easy to extend and to understand. An outline of the model implementation can be seen in Figure 3.1.

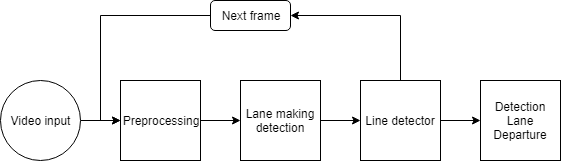


Figure .: Flowchart of the model outline

## Preprocessing

The input data is an RGB-based color image sequence taken from a moving vehicle. A color camera is mounted inside the vehicle at the front-view along the central line. It takes the images in front of the vehicle including road, vehicles on the road, roadside. Roads can be marked by well-defined solid lines, segmented lines.

Preprocessing step is an important part of the system, which helps us to reduce computational time and improve the results of the proposed algorithm. It includes two stages. First, we convert image to its grayscale image. Second, we base on vanishing point to limit ROI selection. We show more details as follows.

### Convert image to its grayscale image

The first module in the main model is a preprocess where the video is processed to speed up the computations in the algorithms. The video was converted to 450x253 pixels in size to reduce the computational domain. Next, we have to convert RGB image to its grayscale image to speed up the method to be applied. Thus, the computational complexity of algorithm can be reduced significantly and the algorithm does less sensitive to environmental conditions



Figure .: Convert RGB image to Grayscale image

Noise is a problem for computer vision as a problem for most systems. Developed algorithms should have noise tolerance or the image should be purified from noise. Aiming at lane marking detection, image smoothing can be done by applying two main filters, Median Filter and Gaussian filter or both, to blur the noisy details. Median filter replaces every pixel value with the median of its neighboring entries. Gaussian filter scans all image with the mask you specify its matrix size, calculates the weighted average of the matrix for each step of scanning, even though mean filter calculates just the average. In our application, we use both Median and Gaussian filter.

The Gaussian takes a kernel size parameter which we will need to play with to find one that works best. We tried 3, 5, 9, 11, 15, 17 (they must be positive and odd) and check the edge detection (see the next section) result. The bigger the kernel size value is, the more blurry the image becomes. However, the bigger kernel size value requires more time to process. Thus, we have chosen 11 because the effect is similar to the values 11, 15, 17.

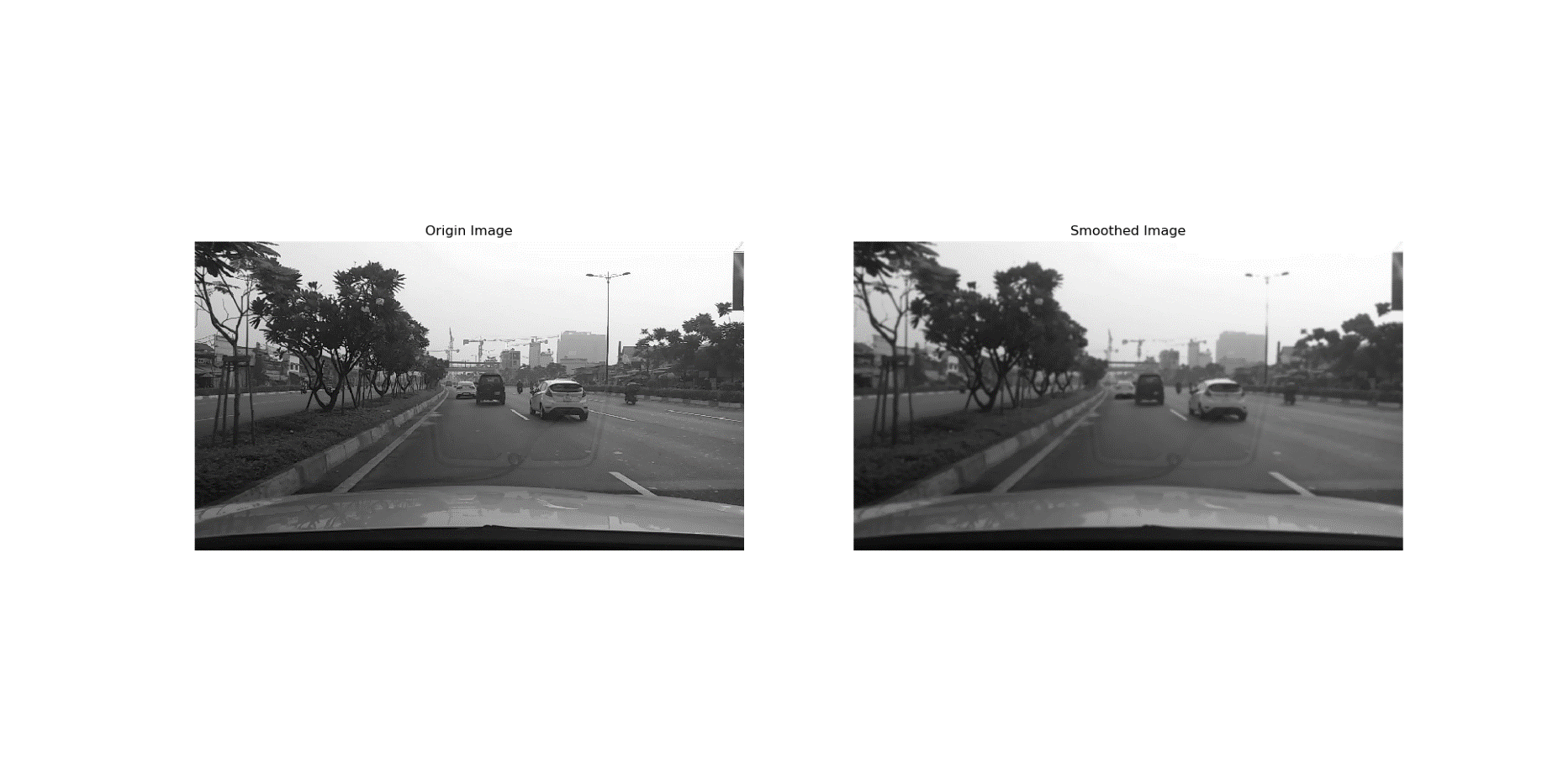


Figure .: Image smoothing

### Region of Interest (ROI)  Initialization

The ROI selection is extracted by cropping the original image. The area which has a high potential of being lane marks is selected [9]. The ROI is defined when we capture the input image from the video or camera as seen in Figure 3.4 by considering the height and tilt angle of the camera which has placed to the vehicle. Cropping also enhances both speed and the accuracy of lane detection system. The speed improvement comes from the reduction in the size of the image to be processed while the accurate improvement comes from the elimination of objects outside the ROI that may have features similar to lane marking.

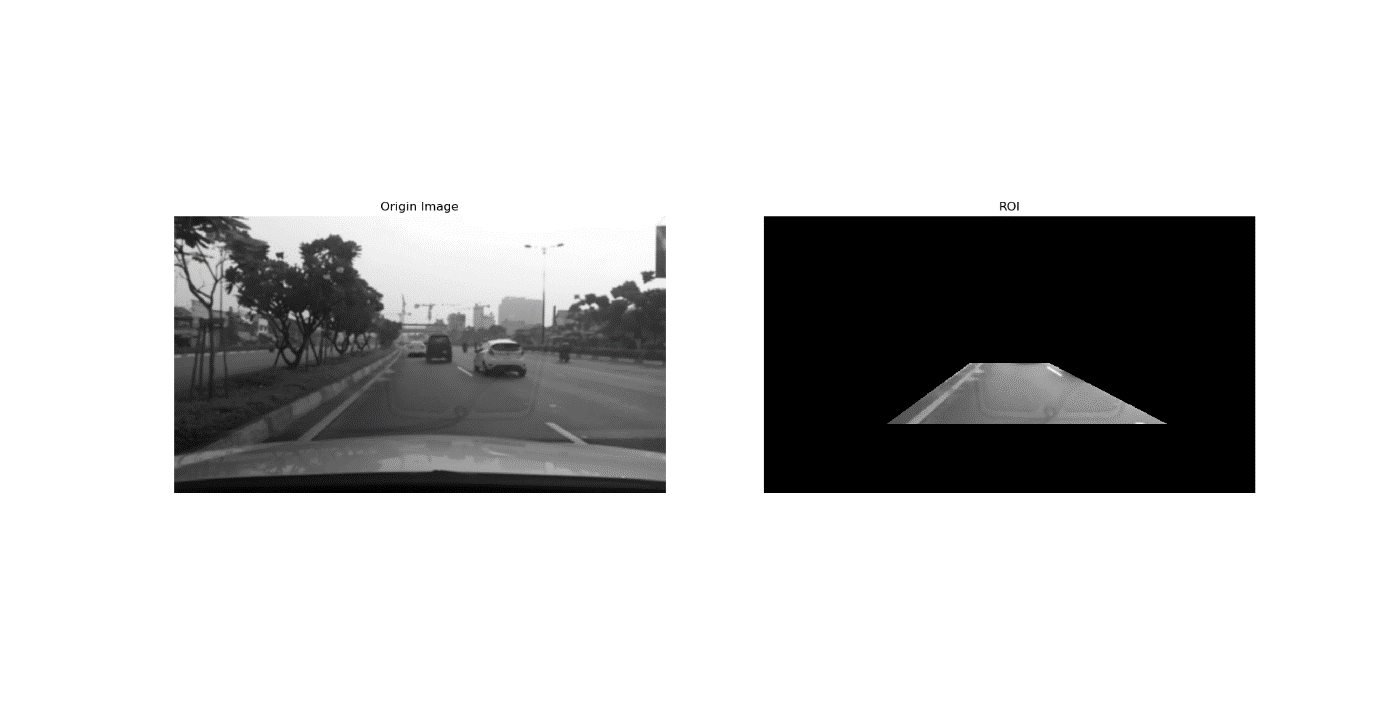


Figure .: ROI Initialization

## Lane marking detection

From the observed image, lane marking has intensity value brighter than intensity value of road surface. There are many methods using this characteristic to detect lane. In this section, we use color selection and Canny edge detection.

### Color selection

Normally, lanes are white or yellow in RGB color space. However, we have converted RGB image to grayscale image. Thus, the lane is a while, which has integer values from 95 to 200 can be seen in Figure 3.5. We build a filter to select those white lines.

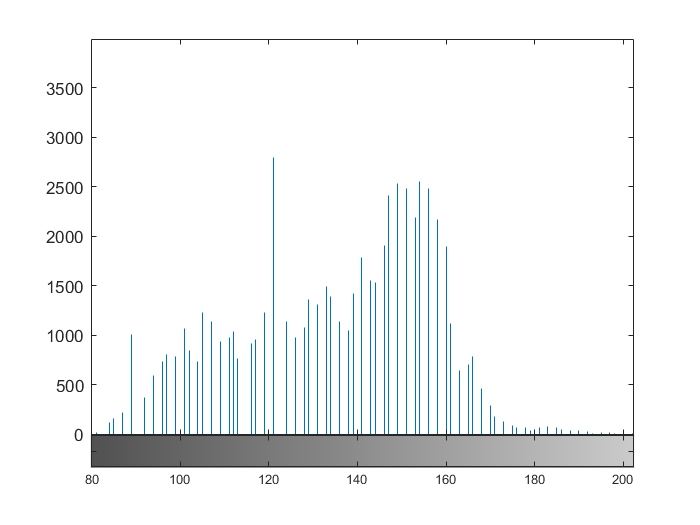


Figure .: Histogram of image

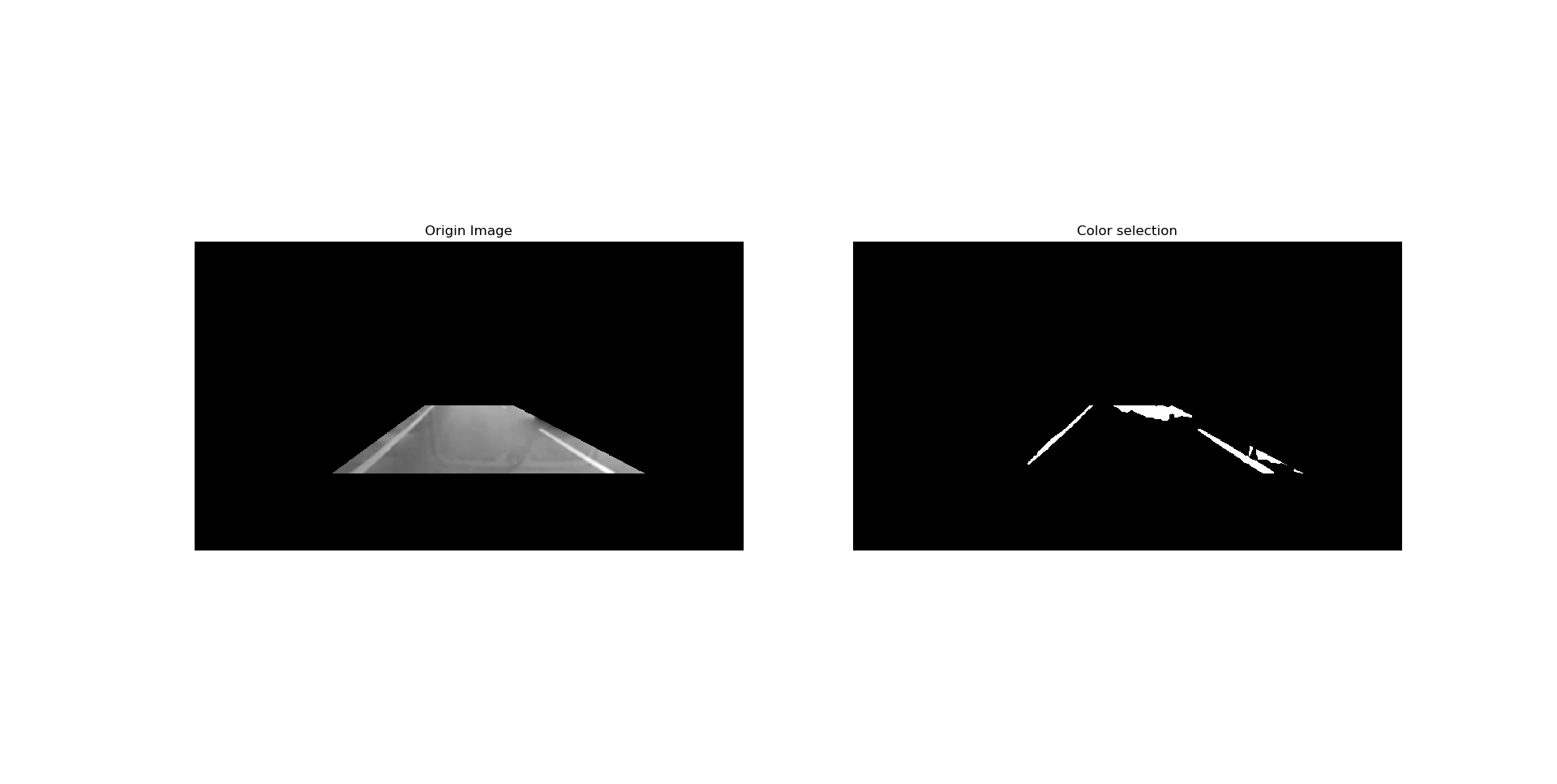


Figure .: Select white lines in image

### Canny edge detection

Canny edge detection method takes two threshold values. According to the OpenCV documentation, the double thresholds are used as follows:

* If a pixel gradient is higher than the upper threshold, the pixel is accepted as an edge.
* If a pixel gradient value is below the lower threshold, then it is rejected.
* If the pixel gradient is between the two thresholds, then it will be accepted only if it is connected to a pixel that is above the upper threshold.
* Canny recommended an upper: lower ratio between 2:1 and 3:1.

These two threshold values are empirically determined. Basically, we will need to define them by trials and errors. We first set the low threshold to zero and then adjust the high threshold. If high threshold is too high, we find no edges. If high threshold is too low, we find too many edges. After many trials, we have chosen 20 and 40 respectively for low threshold and high threshold.

## Line detector



Figure .: Canny edge detection

The line detector used is a standard Hough transform with a restricted search space. The Hough transform is used in a variety of related methods for shape detection. For implementation on an image, more often than not, the Hough Transform is performed after edge detection has been done.

In reality, we can reject any line that falls outside a certain region. For example, a horizontal line is probably not the lane boundary and can be rejected. We only detect the straight lines with restriction of . Next, we search in the Hough space for the long straight lines. There are many straight lines detected by Hough Transform, now we search in the Hough space to find the long straight lines, which are lane marking candidates. The lane markings include more edge pixels than other lines in the image. The resultant images after Hough transformation is applied to images in our application are shown in Figure 3.8.



Figure .: Images after Hough Transformation

Hough transform, as shown in Figure 3.8, can produce multiple results for each lane mark. We need to come up with an averaged line for that. Also, some lane lines are only partially recognized. We need to extrapolate the line to cover full lane line length. We want two lane lines: one for the left and the other for the right. The left lane should have a positive slope, and the right lane should have a negative slope. Therefore, we will collect positive slope lines and negative slope lines separately and take averages. Input image results are shown in Figure 3.9.



Figure .: Detected lane marks in input images

## Control LEDs with Raspberry Pi 3

Connect LED to GPIO pins on Raspberry Pi 3 using Python

The first, we need to note the polarity of the LED. There are 3 ways to identify the LEDs:

* Option 1: Based on the foot of the LED, the longer leg is usually connected to the high voltage source, short legs connected to the low voltage source (GND).
* Option 2: Based on internal pole LED, very small – high pressure, very large – low pressure (GND).
* Option 3: Based on defect on the LED body, no high-pressure, low-pressure (GND).

Next, we will connect the long leg to resistor to connect the corresponding pin GPIO 17 for led red and GPIO 19 for led white. The short pin connected to pin GPIO Ground is GND (as shown in footer diagram).

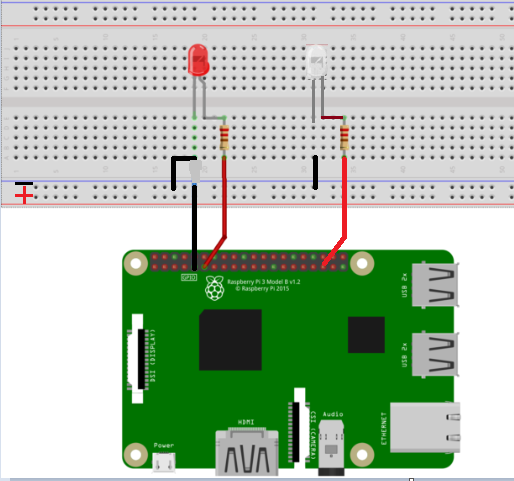


Figure .: Connect LED with Raspberry Pi 3

# LANE DEPARTURE WARNING SYSTEM

In this chapter, we build a Lane departure warning system on Raspberry Pi 3 model B.

## Lane mark parameters

While the vehicle is on a normal course, two-lane marks appear in Figure 4.1. Left and right lane marks are symmetrical with respect to each other along a vertical line, and their angles with respect to the horizontal axis are equal. If the vehicle is making lane departure, one lane mark will appear more vertically than the other one. If the vehicle is making left lane departure, left lane mark is more vertical than right one Figure 4.2 or if the vehicle is making right lane departure, right lane mark is more vertical than left one Figure 4.3. We can easily detect that the vehicle is making right or left lane departure by using this feature [9].

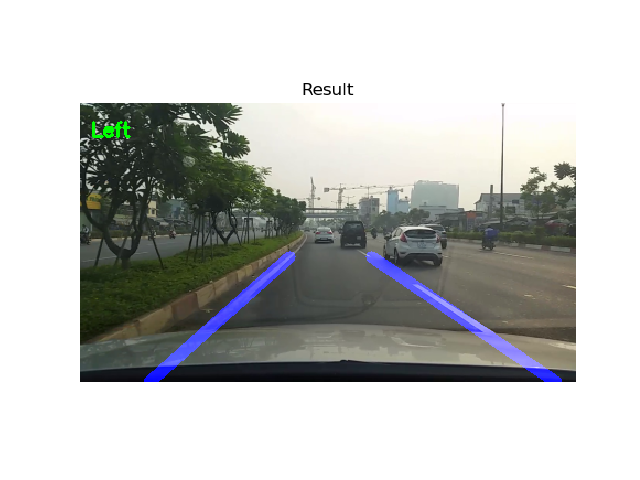


Figure .: Ideal case of lane departure

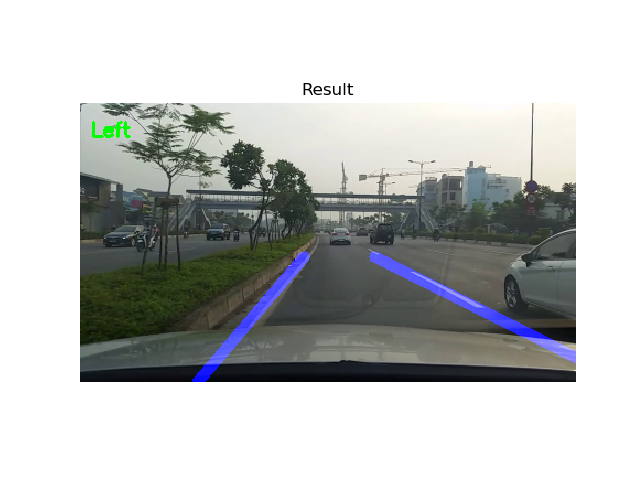


Figure .: Left lane departure



Figure .: Right lane departure

## Detection Lane Departure

It can be decided which lane mark is more vertical by looking at their angles with respect to the horizontal axis. Greater angle value with respect to x-axis will indicate more vertical line between two-lane marks. Both angles are in the ideal situation as seen in Figure 4.4a and there is no any lane departure. If one of the angles is greater than the other, lane departure starts at the direction of which lane mark’s angle is great [9] as seen in Figure 4.4b and Figure 4.4c. When the vehicle turns right, red LED will turn on. In contrast, when the car turn left white LED will turn on.

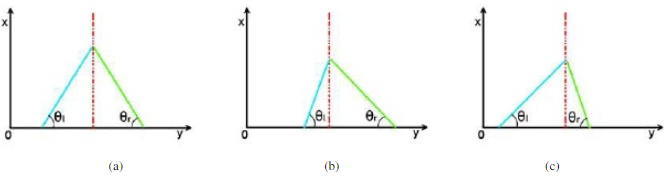


Figure .: Graphical representation of lane departure cases, a) ideal case, b) left lane departure, c) right lane departure.

(Source: https://www.academia.edu/2823/Real\_Time\_Lane\_Departure\_Warning)

## Experimental results

The proposed lane departure warning algorithm has been implemented on Raspberry Pi 3 Model B by using OpenCV library with Python programming language. The images have been captured from video have 1280x720 image resolution. The processing time of proposed algorithm was about 0.1~0.2s per frame. Sample images is show in Figure 4.5

Figure .: Sample images of the lane detection.



# CONCLUSION

This chapter contains conclusions drawn from this thesis work along with suggestions for future work.

## Conclusion:

Comparing with the Objective we had set, we conclude:

* The main part of this system consists of lane detection, lane detection has been eased by avoiding the disadvantage of perspective effect by using inverse perspective mapping.
* The proposed algorithm is more adaptive to the various road.
* Using blink lights to warn the driver.

## Development:

In future:

* Lane departure warning system can integrate with the vehicle.
* Apply Deep Learning into the thesis.

REFERENCES

1. R. Gonzalez and R. Woods (2010). “Digital Image Processing”. third. International edition. *Pearson Education*.
2. Arnaud Frich. “Color spaces”. URL: <https://www.color-management-guide.com/color-spaces.html>.
3. Rakesh M.R, Ajeya B, and Mohan A.R (2013). “Hybrid Median Filter for Impulse Noise Removal of an Image in Image Restoration”. pp 5117 – 5120.
4. Stephen Wolfram. "GuassianFilter". URL: <http://reference.wolfram.com/language/ref/GaussianFilter.html>.
5. Thomas B. Moeslund (2009). "Canny Edge Detection". pp 2 - 5.
6. Christopher M. Brown (1982). “Bias and Noise in the Hough Transform I: Theory”. pp 3 – 5.
7. Ghassan Hamarneh, Karin Althoff, and Rafeef Abu-Gharbieh (1999). “Automatic Line Detection”. pp 7 – 10.
8. Raspberry Pi Foundation. “Raspberry Pi 3 Model B+”. URL: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus>.
9. Kadir İLERİ and Salih GÖRGÜNOĞLU (2016). “Real Time Lane Departure Warning System for Drivers”. pp 543 - 544