# Warning Traffic System

Wanchalerm Chaithip*, Supasin Nguantad, Sarun Olankranok and Sirikan Chucherd

Center of Excellence in AI and Emerging Technology, School of Information Technology, Mae Fah Luang University, Chiang Rai, Thailand

*Corresponding author. E-mail: Wanchalermchaithip@gmail.com

## ABSTRACT

Nowadays, many people use the road to travel daily. Resulting in many types of vehicles on the road since we have to share the road with many people how can we be sure that it is safe? Traffic conditions and vehicle driver decisions are also one of the factors in accidents. Therefore, we decided to build a device that helps us to know the current traffic conditions and is aiding in our road usage decisions. Our equipment will consist of a device for vehicle detection, a camera for detecting all vehicles, and a Raspberry Pi used for processing the number of vehicles entered and displayed via a display. Consisting of LED. If there are a lot of vehicles, there will be a warning to road users on our display. We test our system by looking at traffic conditions at different times. Shows the times when there is bad traffic. It was created to help make decisions for road users.

*Keywords: Raspberry Pi, OpenCV, Vehicle detection.*

## INTRODUCTION

Mae Fah Luang University access road is the main road; therefore, all students must use in-traffic and this road is also a path for traveling to the student dormitory. Thus, this road has bus commuters throughout the day. Frequently, there is often a delay in leaving the university because many students, professors, and staff leave the university at the same time from traffic jams and accidents.

Nowadays there are many devices used to detect objects. Street Mark Detection System. A self-driving utilizes many methods such as radar, GPS, camera, etc. This system was designed using Raspberry Pi and webcam to process the image (Sumardi et al., 2018). Road and lane detection from different lane marks, the width of the marks is changed, Image clarity issues, low visibility due to heavy rain (Aharon Bar Hillel et al., 2012). Car detection cross-domain (day-to-night) using training datasets without annotations compared to the training with only the available annotated data, the detection performance is increasing more than 10 % (Vinicius F. Arruda et al., 2019). In the real-time traffic video monitoring using the algorithm to reduce the missed detection and error detection rate information and using the multi-objective particle swarm optimization algorithm to extract the vehicle boundaries (Shijun Yu et al., 2018). Face detection, OpenCV and Raspberry Pi3

Article history:
Received 12 March 2020; Received in revised from 6 October 2020;
Accepted 11 November 2020; Available online 11 November 2020

was used together to detect motion and face. The output storage into the cloud. It is an effective method for detecting human beings in both normal and blurred images (Zoltán Balogh et al., 2019).

Although the above methods get good results some images cannot be detected, training file is not covered with the things that need to be detected, large and expensive equipment, Therefore, vehicle detection is still an interesting research topic. In this paper, we offer real-time vehicle detection using different devices. By using the camera of the Raspberry Pi 3 in vehicle detection, OpenCV library and use the python language to run the system. Each device is managed to send data to each other over the internet. This paper is structured as follows. Section 2 describes the proposed algorithm and some details of the materials. The results and conclusion will be in sections 3 and 4 respectively.

## METHODOLOGY

### Technology and Tools

Raspberry Pi foundation is a UK-based open-source that attempts to put the performance in computing and digital making under the control of individuals everywhere throughout the world. We do this so that more people can harness the performance in computing and digital technologies for work, to solve issues that matter to them and to express themselves creatively.

**Figure 1** Raspberry Pi 3 Model B

RPi IR-CUT Camera is a camera module for the Raspberry Pi board that has a 5-million- pixel resolution camera for shooting both during the day and at night or in low light using the same OV5647 sensor used in the official Raspberry Pi's camera. It can rotate the front lens element to adjust the focus distance. Connect via Camera Interface (CSI) port.

Relay Module that can be used as a switch to turn on / off the device. It can control devices that use both AC and DC power not more than 250V 10A. The 1 channel 5V relay module works with Active HIGH. The electric trigger in pins ranges from 3-5 volts.

**Figure 2** RPi IR-CUT Camera

**Figure 3** Relay Module

Cascade Trainer GUI is a program that can be utilized to train, test and improve course classifier models. It utilizes a graphical interface to set the parameters and make it simple to utilize the OpenCV tool for training and testing classifiers.

**Figure 4** Cascade Trainer GUI

**Block diagram**

The block diagram as shown in Figure 5, we use Raspberry Pi 3 Model B to collect the data from RPi IR- CUT Camera. Raspberry Pi 3 Model B (Sever) will send the data to Raspberry Pi 3 Model B (Client). And the Client will take the data from the Server to analyze and control the operation of the LED.

OpenCV is an open-source library of computer vision applications. We can use it for free and develop it in real-time. Moreover, it mainly focuses on image processing, video capture and analysis including features, for example, face recognition and vehicle detection (Bradski, G. R., & Kaehler, A., 2011)

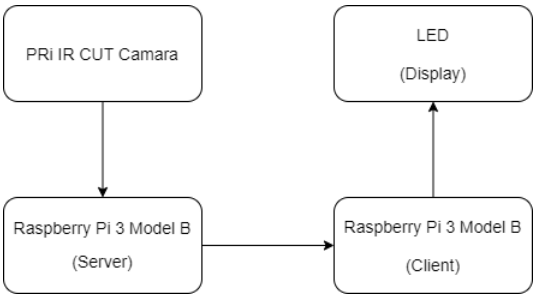


**Figure 5** Block Diagram

Image processing is the utilization of computer algorithms to achieve image processing on digital images. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing (Digital image processing., 2019).

**Flowchart**

Flowchart of the system: Send and receive value between server and client as shown in Figure 6.

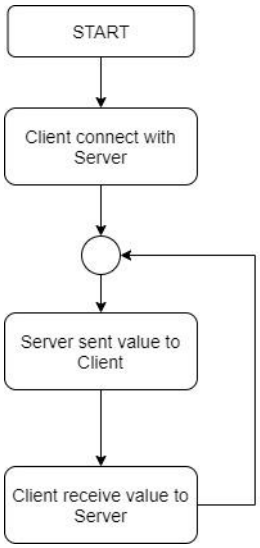


**Figure 6** Flowchart of the system

The following steps describe the system diagram. Step 1 to 3 explain the operation of server, while step 4 to 6 represent the steps of client part.

1. Sever try to connect with client
2. If connect with client success camera will open. If not try to connect with client again
3. Detect vehicle if detected server will send x int (1) to client
4. Client receive x value from server
5. If x<4 relay sent logic 1 to Green LED, if 3<x<7 relay sent logic 1 to Orange LED, if x>6 relay sent logic 1 to Red LED
6. If timer>60s client will set x = 0 and timer = 0
7. Start timer and receive x value again

## Model

We designed the box dimension by the size of 45x24x14 cm. The relay module and Raspberry Pi were set at the back of the box. For LED, we placed three different 64-LED colors (green, yellow and red) set in the box to represent the warning sign. We created a box that has a lot of LEDs because we want people to be able to see easily. Colors of LED are red for many vehicles, yellow for medium vehicles and green for fewer vehicles as shown in Figure 7.

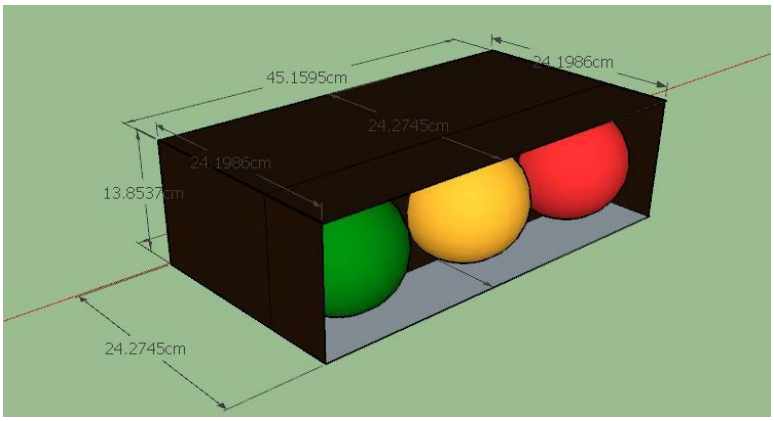


**Figure 7** Model

## Circuit diagram

From the server circuit diagram, Raspberry Pi is connecting with an RPi IR-CUT Camera as shown in Figure 8.

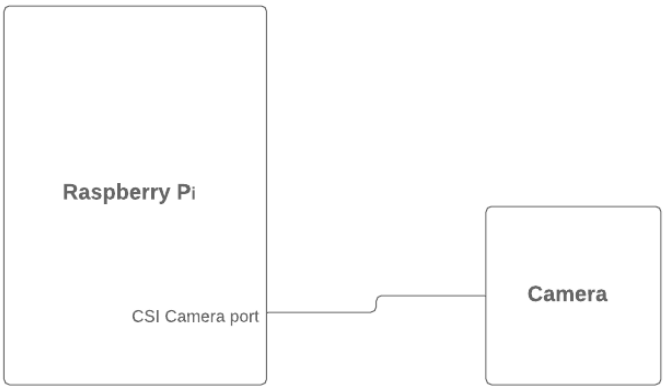


**Figure 8** Server circuit diagram

From the client circuit diagram, Raspberry Pi is connected with two relays by supplying 5V to the relay to operate. Then it connects with three batteries to supply voltage to all of LED as shown in Figure 9.

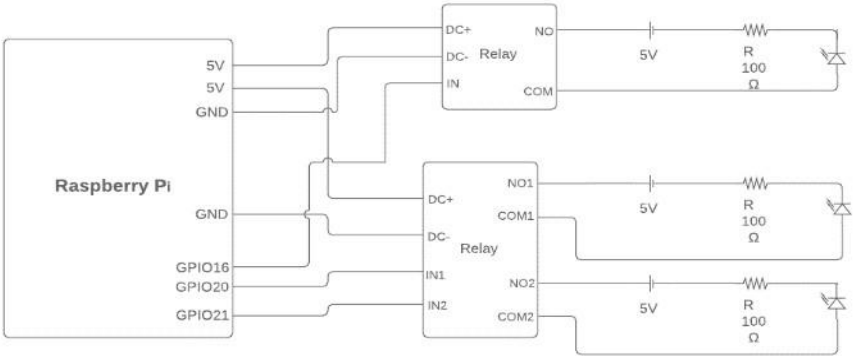


**Figure 9** Client circuit diagram

## Output

We created text "count: " to show the total number of vehicles that can be detected. And create a blue line to detect the car when the car passes this line and the value will go up to text "count: " as shown in Figure 10.

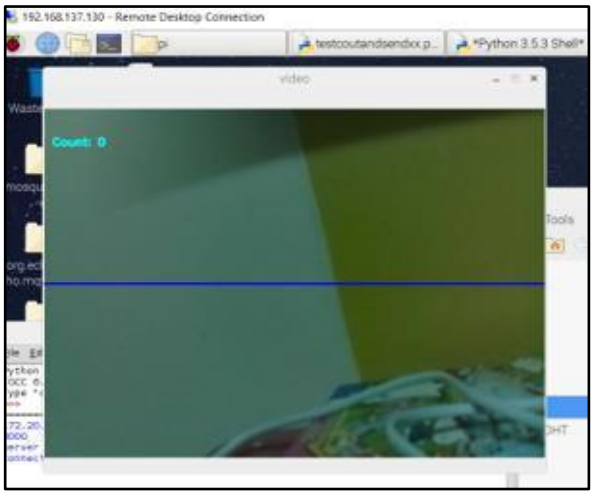


**Figure 10** Detection display

We created the LED dashboard with electronic equipment. It shows a green, orange and red light to tell several cars to drive out of university and installs with a Raspberry Pi. It worked as a client to receive server values from another Raspberry Pi as shown in Figure 11.

**Figure 11** Display

For the algorithm to detect vehicle we use the code from Erdem Arslan (Erdemarslan/opencv_motion.py., 2018). And develop into code of this paper as shown in Figure 12.

```
import socket
import time
import schedule
import RPi.GPIO as GPIO
pin1 = 20
pin2 = 16
pin3 = 21
GPIO.setmode(GPIO.BCM)
GPIO.setup(pin1, GPIO.OUT)
GPIO.setup(pin2, GPIO.OUT)
GPIO.setup(pin3, GPIO.OUT)
GPIO.setwarnings(False)
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host ="172.20.10.3"
port =8000
s.connect((host,port))

i=0
totall=0
data=0
start = time.clock()
while i<99:

    t0=time.clock()
    data = s.recv(1024).decode()
    totall=int(totall)+int(data)
    print(totall)
    elapsed = (time.clock() - start)
    print("check condition")
    print(elapsed)
    if (totall<3):
        GPIO.output(pin1, GPIO.HIGH)
    elif (3>totall<6) :
        GPIO.output(pin2, GPIO.HIGH)
    else :
        GPIO.output(pin3, GPIO.HIGH)

    if (elapsed>60):
        print("elapsed>60")
        start=time.clock()
        totall=0
        GPIO.output(pin1, GPIO.LOW)
        GPIO.output(pin2, GPIO.LOW)
        GPIO.output(pin3, GPIO.LOW)
```

**Figure 12** Example of coding in the client

## RESULTS

In this project, the system has been implemented for detection the vehicle passing the line using the image processing with OpenCV system. Finally, the system would detect each vehicle with a Deep learning algorithm.

### Experimental detection results by Cascade Trainer GUI

We took the clip from the actual location to test the Cascade Trainer GUI program because we experimented on a computer with more process faster than Raspberry Pi, resulting in relatively high accuracy, few mistakes.


**Figure 13** Test by Cascade Trainer GUI

**TABLE 1** The vehicle detection by Cascade Trainer GUI

| Type | Condition | Number of vehicle | Percentage (%) |
|---|---|---|---|
| Accurate | The camera can detect vehicle. | 15 | 83.33% |
| Error | The camera can detect the vehicle, but it also detects other objects. | 3 | 16.67% |
| | Total no. of vehicle | 18 | 100% |

Table 1 shows the experiment of vehicle detection. We use the cascade trainer GUI program to test detection. The analysis gives the following results. First, the number of samples for accurate is 15 samples depend on the condition "The camera can detect vehicle". The result rate representing 83.33%. Second, the number of samples for error is 3 samples depend on the condition "The camera can detect the vehicle, but the camera detect other objects.". The result rate representing 16.67%.

**Experimental detection results by practicality**

We experimented from real locations by putting the project in the corner that we trained in the Cascade Trainer GUI program. From the experiment, it can be seen that the Raspberry Pi has heavy work and high heat as shown in Figure 14. Resulting in slow project work detects less than an experiment in the Cascade Trainer GUI. Figure 15 and 16, the camera can detect and count the number of the motorcycles and cars respectively.

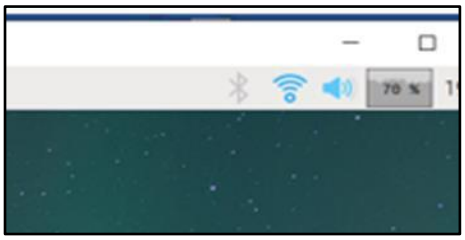


**Figure 14** Percentage of Raspberry Pi processing

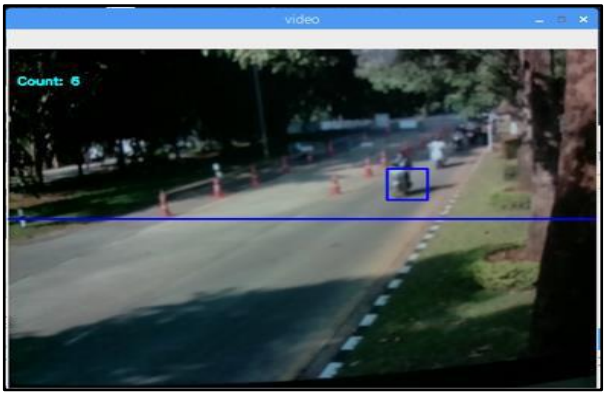


**Figure 15** Motorcycle detection

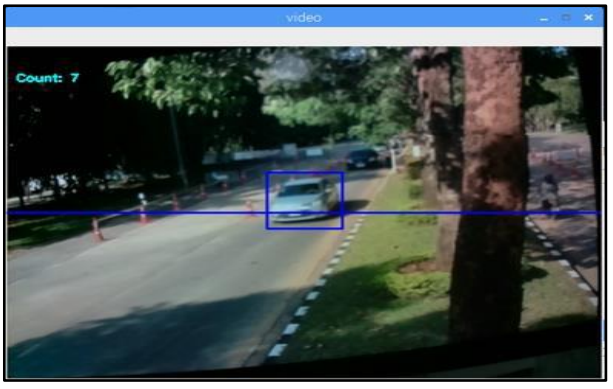


**Figure 16** Car detection

| Testing period | No. of vehicle | No. of detected vehicle | Counting (If detect and pass line at the same time) | Accuracy of detection (%) | Accuracy of counting (%) |
|---|---|---|---|---|---|
| 9.00-9.10 a.m. | 47 | 32 | 17 | 68.09% | 36.17% |
| 12.30-12.40 p.m. | 126 | 76 | 35 | 60.32% | 27.78% |
| 3.00-3.10 p.m. | 188 | 131 | 42 | 69.68% | 23.20% |

**TABLE 2** The vehicle detection by practicality

Table 2 shows the experimental result of vehicle detection. The experiment has been designed during rush hours (morning, noon and afternoon period). The total number of vehicle manually counted by human are 47, 126 and 188 respectively. The vehicle that detected by our system are 32, 76 and 131. The accurate result rate representing 68.09% 60.32% and 69.68%. Second, the number of counting for the experiment is 17, 35 and 42. The accurate result rate representing 36.17%,27.78%, and 23.20% respectively.

## CONCLUSION

In this paper, we presented the experimental of the vehicle detection drive out from Mae Fah Luang University. We designed the hardware and software to implement our experiments in practicality conditions. From the experimental result, we found that some vehicles were detected but the counting accuracy is low. Various factors such as light and shadow of vehicles or trees may cause the difficult detection results. The number and the varieties of the training and testing dataset of vehicle may affect to the accuracy. The processing speed of Raspberry Pi is another problem with real-time situation. However, our proposed vehicle detection is helpful for traffic warning especially at the crossroads. Drivers can look at the LED display and decide whether they should cross the road or not.

## ACKNOWLEDGMENTS

This research was supported by the Center of Excellence in AI and Emerging Technology, School of Information Technology, Mae Fah Luang University.

**REFERENCES**

Sumardi, S., Taufiqurrahman, M., & Riyadi, M. A. (2018). Street Mark Detection Using Raspberry PI for Self-Driving System. TELKOMNIKA (Telecommunication Computing Electronics and Control). ***16*(2)**, 629. doi: 10.12928/telkomnika.v16i1.4509

Hillel, A. B., Lerner, R., Levi, D., & Raz, G. (2012). Recent progress in road and lane detection: a survey. Machine Vision and Applications. ***25*(3)**, 727–745. doi: 10.1007/s00138-011-0404-2

Arruda, V. F., Paixao, T. M., Berriel, R. F., Souza, A.F.D., Badue, C., Sebe, N., & Oliveira- Santos, T. (2019). Cross- Domain Car Detection Using Unsupervised Image- to- Image Translation: From Day to Night. 2019 International Joint Conference on Neural Networks (IJCNN). doi: 10.1109/ijcnn.2019.8852008

Yu, S., & Deng, S. (2018). Adaptive vehicle extraction in real-time traffic video monitoring based on the fusion of multi- objective particle swarm optimization algorithm. EURASIP Journal on Image and Video Processing, 2018(1). doi: 10.1186/s13640- 018-0381-8

Motion Detection and Face Recognition using Raspberry Pi, as a Part of, the Internet of Things. (2019). Acta Polytechnica Hungarica. ***16*(3)**. doi: 10.12700/aph.16.3.2019.3.9

Bradski, G. R., & Kaehler, A. (2011). Learning OpenCV: Beijing: OReilly.

Digital image processing. (2019, November 14). Retrievedfromhttps://en.wikipedia.org/wiki/Digital_image_processi

Erdemarslan/opencv_motion.py. (2018). Retrieved from https://gist.github.com/erdemarslan