



Department of Mechatronics & Industrial Engineering
Chittagong University of Engineering and
Technology

**REMOTELY CONTROLLABLE DRIVERLESS VEHICLE
EQUIPPED WITH COMPUTER VISION SYSTEM**

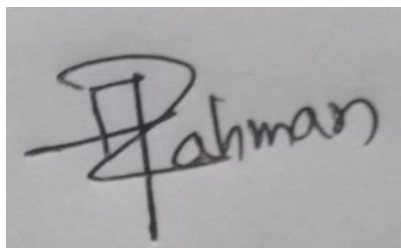
Masud Ahmed Rifat

ID:1509007

APRIL, 2021

A PROJECT SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF SCIENCE
IN
MECHATRONICS AND INDUSTRIAL ENGINEERING

SUPERVISED BY:

A handwritten signature in black ink on a light gray background. The signature is stylized, starting with a large 'R' and followed by the name 'ahman'.

Md Abdur Rahman

Lecturer

DEPARTMENT OF MECHATRONICS & INDUSTRIAL ENGINEERING
CUET

Declaration

This project is a presentation of our original research work ideas. Whenever contributions of others are involved, every effort has been made to indicate this clearly, with due reference to the literature, and acknowledgement of collaborative research and discussions.

This work was done under the guidance of Md Abdur Rahman, Lecturer at Chittagong University of Engineering and Technology, Chattogram.

Abstract

In this project a self-driving car is designed. This car is equipped with camera. Here we used histogram analysis to easily identify road. Using histogram method requires little memory and low processing power. A Raspberry Pi computer is sufficient. Some pre-processing is applied such as binary thresholding, perspective warping, Region of Interest. The visual system can work with any colored path. This car can drive itself on the road even if the road is curved.

Drive optimization is done by averaging method for smoothing out road following movement. In this novel method, use of histogram makes the process applicable in small computers. As small computers like Raspberry Pi have very small amount of computing capability, our system is capable to run the full process. The image processing method we proposed here is far better for efficient use of memory and processing power. We used histogram for its superiority in any color intensity variation.

It is also designed for remote control so surveillance works can be carried out. The system is designed as it is expandable and modular. The modular approach make it fit for small scale applications. The car is prototype and can be used for further experimentation and improvement. Our presented remote control is suitable for portable setups.

The remote control and road following method both can be applied in the same setup. This gives us flexible maneuverability. Whenever we need which setup, we can easily deploy it. Using python as programming language, makes it easier to debug and also gives us the flexibility of using huge 3rd party libraries. This car is an achievement for the advancement of future automobile technologies.

Acknowledgement

Above all, I would like to express my gratitude to my supervisor **Md Abdur Rahman** for accepting me into his research group and also express my heartfelt thanks to him for his guidance, encouragement and continuous support during my graduate studies. His enthusiasm for teaching and research offered challenging opportunities to expand my scientific knowledge and my growing interest in the world of Mechatronics.

Table of Contents

Declaration	3
Abstract	4
Acknowledgement.....	5
List of Figures	8
Chapter 01	9
Introduction	9
1.1 Introduction	9
1.2 Project Objective	10
1.3 Project Overview	10
Chapter 02	12
Literature Review	12
2.1 Related works.....	12
2.2 Scopes	13
2.3 Conclusion.....	13
Chapter 03	14
Methodology	14
3.1 Introduction	14
3.2 System Design.....	14
3.3 Hardware Components.....	19
Chapter 04	24
Results	24
Conclusion.....	26
References	27

Appendix	29
----------------	----

List of Figures

Fig. 1. Automation Level	10
Fig. 2. Project Flow	10
Fig. 3. Main components.....	14
Fig. 4. Image Processing Pipeline	15
Fig. 5. HSV color space	15
Fig. 6. Binary Thresholding	16
Fig. 7. Warp Function	16
Fig. 8. Right most is after warping.....	16
Fig. 9. Road Curvature Decision.....	17
Fig. 10. Pixel summation in Histogram for left curve.....	17
Fig. 11. Left side is road image. Right side is histogram and average base point.	18
Fig. 12. Decision Making Process Flowchart	18
Fig. 13. Remote Control Process.....	19
Fig. 14. Raspberry pi	19
Fig. 15. Pi Camera.....	20
Fig. 16. 18650 Battery.....	20
Fig. 17. XL4015 Power Supply Module	21
Fig. 18. Breadboard.....	21
Fig. 19. L293d Motor Driver IC.....	22
Fig. 20. Car Chassis	22
Fig. 21. Actual car photo.....	24

Chapter 01

Introduction

1.1 Introduction

A driverless vehicle is also known as self-driving car. It implements a hardware and software integration to enable a vehicle to sense its environment and moving safely without any human input. It combines a variety of sensors to perceive its surroundings. Advanced Control systems interpret sensor values and determines optimum navigation path.

There is a great amount of road accidents occurring in Bangladesh due to two way roads. Most of them are caused by face-to-face collision. As reckless drivers try to overtake, they jump into the wrong lane and this creates catastrophic accidents.

According to SAE International, there is six levels of automation. The levels are:

Level 0 is warning system but no automated control implemented.

Level 1 dictates shared control between human and computer system. Such as adaptive cruise control, lane keeping assistance, automatic emergency braking etc.

Level 2, automated system controls the car but the human needs to be always ready to take over in case of system failure.

Level 3 relaxes the human driver, as human can move his eyes away.

Level 4 assures that driver can go even sleep. This can be applied in specified limited locations.

Level 5 is fully reliable autonomous vehicle. There will be no need for driver.

Level of Automation is shown in following figure 1 [0]

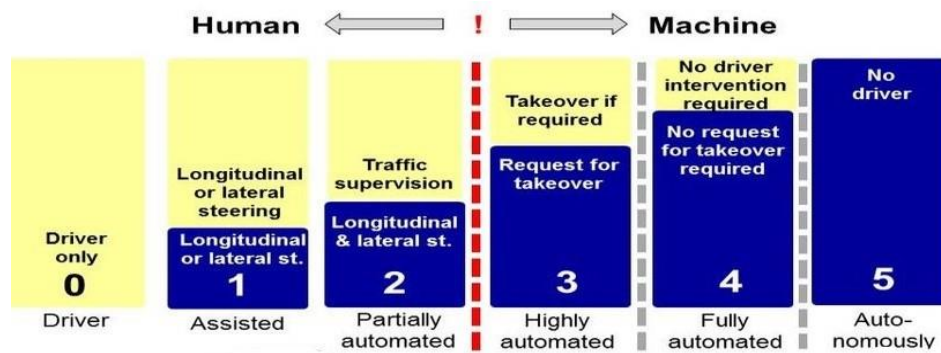


Fig. 1. Automation Level

For autonomous vehicle, remote control is an emergency feature. Whenever there is present such an environment, which the system is not designed for; then an external human operator should operate the vehicle. Also remote control is very useful in surveillance operations.

1.2 Project Objective

- Develop a prototype car with computer vision system.
- Integrate automatic lane detection.
- Develop Automatic lane following control.
- Develop remote control of the car.

The project flow is depicted in following figure 2

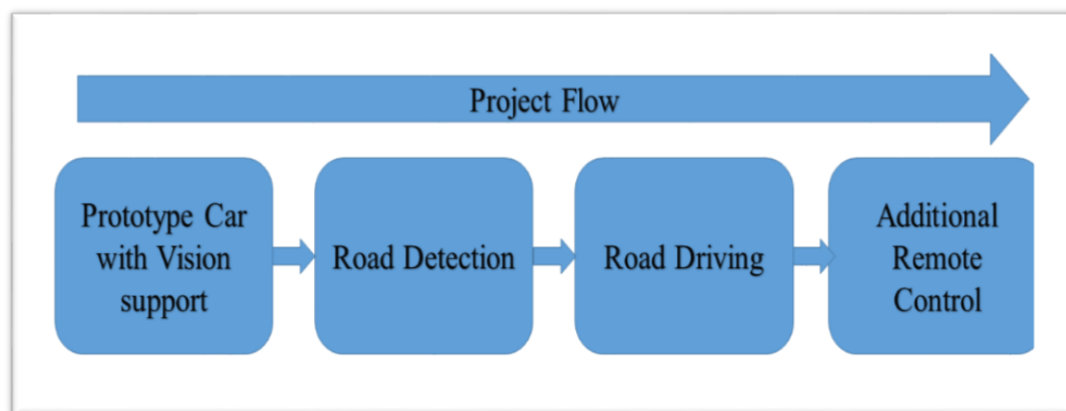


Fig. 2. Project Flow

1.3 Project Overview

This project improves road safety by incorporating automation in cars. Images from front camera are processed by on board computer to detect and drive the car on lane. This will increase safety and eliminate wrong lane collisions. Now a days cars are fully

manual, drivers need to take full control all the time and every moment. But with the help of this projects system, the car will be able to detect road and even drive through the lane. The computer takes visual snapshots and with processing detects road. It can drive the car along the lane.

As the system uses modular design, future improvements are possible by integrating new sensors and updating software.

Also the car has remote control feature. It can be remotely controlled. When remote control mode is activated, the driver operates the car using his remote from a safe and remote position.

Chapter 02

Literature Review

2.1 Related works

In 1975 Hummel [1] publishes a paper about how we can retrieve various data from raw grayscale images. He suggested various techniques for analyzing histograms that makes it easy to get features from the gray level of images.. Juan Pablo et al [2] proposed a system that exploits the characteristics of the gray level histogram of the road to detect lane markers. Each lane marker is then analyzed using a decision tree and finally the relations between lane markers are analyzed to create structures defining the lane boundaries.

Patiyuth et al [3] proposed a system that is mostly accurate in an environment where there is no objects or obstruction, the road is clearly seen and the light is not too bright. They used Hough transformation for lane line detection. Pritha Gupta et al [4] proposed an autonomous car that also features obstacle avoidance and remote control. They uses Hough line transformation. They also have an extra feature of region of interest. The region of interest is a very good way to reduce calculating area in the view. Only lower third of the view is used for finding lane. Narayan P. P. et al [5] uses ultrasonic sensor to avoid obstacles. Though they used Hough line transformation for finding lane lines and gave satisfactory results. They also compressed there video before streaming. Network bandwidth made a significant impact on video transmission.

R. Muthalagu et al [6] proposed a better way to detect lane or road markings. They emphasized on preprocessing of the image. Both Sobel operator and color thresholding is used. Sobel operator is related to canny edge detection, which is further used with Hough transformation. At the end they applied sliding window search technique. A third order spline is fitted over first identified points of maxima. The spline marks the first lane. There is a shortcoming in this method, it heavily depends on preprocessing

of image. This preprocessing is computationally expensive. Also if there is a steep curve then it might be missed from defined ROI.

Rami Watheq Yaseen et al [7] designed a miniature autonomous vehicle. They used kalman filter that is deterministic in nature. Here lane detection and tracking is done using that kalman filter. Also they used Finite state machine that cooperates among path planning, lane following, parking strategies. The modifications that they made for the lane detection consisted in considering the two regions of interest. It is thereby reduced the search space. The edge detection, it is based on applying the vertical Sobel operator. The vanishing point detection is done with intersecting the two relevant lane markings, without Random sample consensus.

2.2 Scopes

The scope of autonomous vehicle is vast. The modern world is giving importance on road safety and modernization. For this automation is necessary. Automated vehicles are emerging and biggest tech-giants are investing in this field. New experiments are being conducted by GOOGLE, APPLE, TESLA, BMW and other world leading companies. Both technology and automotive companies are incorporating new features like lane assist, parking assist, and collision avoidance system. So this field is emerging quickly. Starting with simple mathematical algorithms, there is a possibility of machine learning too.

2.3 Conclusion

As reviewing literatures we can conclude that mathematical laws and algorithms are continuously getting easy to apply on powerful new processors. Results are pretty good. As more powerful computing devices are available, big data related machine learning is also experimented. Both simple techniques like histogram and complex processing like artificial intelligence need to be integrated simultaneously to get the best result.

Chapter 03

Methodology

3.1 Introduction

There are two sections in this project. One is the autonomous driving and the other one is remote control. Autonomous driving is a multi-step procedure. The computer takes picture through the camera and process the image frame to find road. After separation of the road from the image background, calculates the curve of the road. If the road goes straight, then command the driver motors to go forward; if the road curves to left, then turns the wheel so the car goes left; if the road curves to right, then the left wheel turns more to make the car turn right. The process then loops again from taking image.

The remote control section is programmed separately. As there is a computer inside the car, a keyboard is set up as the remote. The keyboard is continuously monitored for key-press. Whenever there is a keypress on the selected keys, program runs the motor as programmed.

3.2 System Design

The system mainly consists of a central processing computer, input and output. This is shown in following figure 3

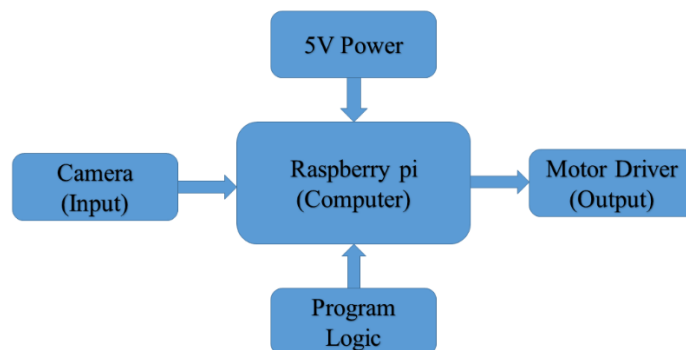


Fig. 3. Main components

The image processing part is crucial for this project. In this project the road is differently colored from the environment. The system detects the road by differentiating the color. The color is specified beforehand. The road can be any color but it should be distinctive

The image processing pipeline is summarized in following figure 4

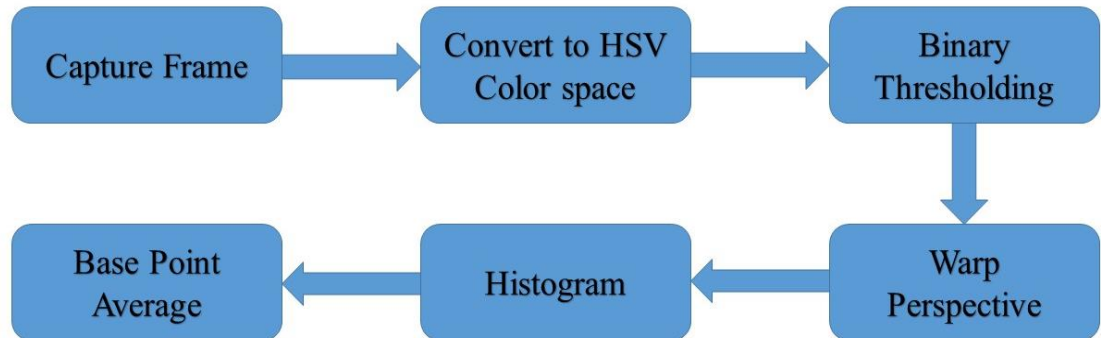


Fig. 4. Image Processing Pipeline

The frame we capture is in RGB format. To easily separate the desired color, we convert it to HSV color space. HSV space is showed in following figure 5 [8]

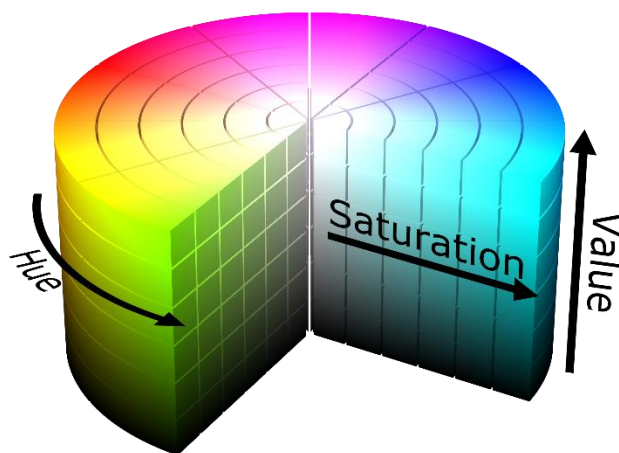


Fig. 5. HSV color space

Binary thresholding is done by masking with binary value. The pixels matched with our value is retained and any other pixels are zeroed. Threshold operation effect is shown in following figure 6



Fig. 6. Binary Thresholding

For better detection of road curvature angle and correct our camera perspective, we warp detected road. It is also called birds eye view. For example is shown this figure 7

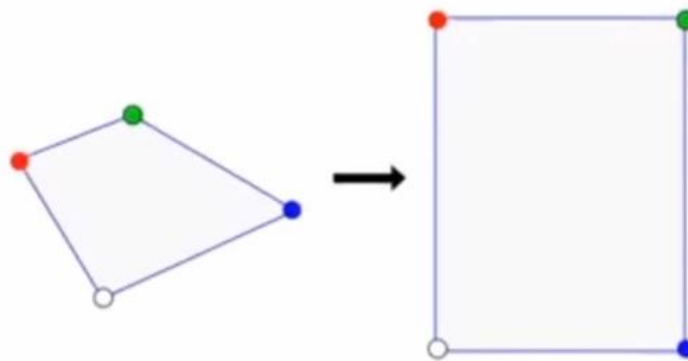


Fig. 7. Warp Function

After warping the image looks like following figure8

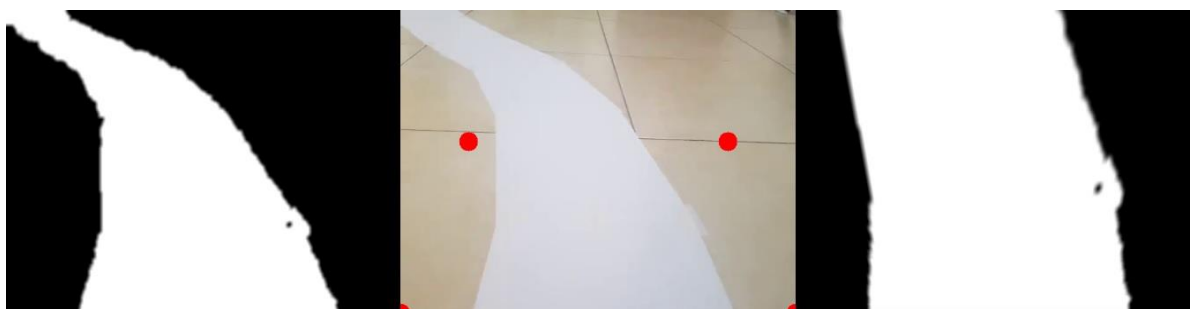


Fig. 8. Right most is after warping

Then histogram method gives us values depending on the weight of the pixels of the road. If the road is curved right then the sum of the pixel values is high on the right side. Movement decision criteria is shown in following figure 9 [9]

PIXEL SUMMATION

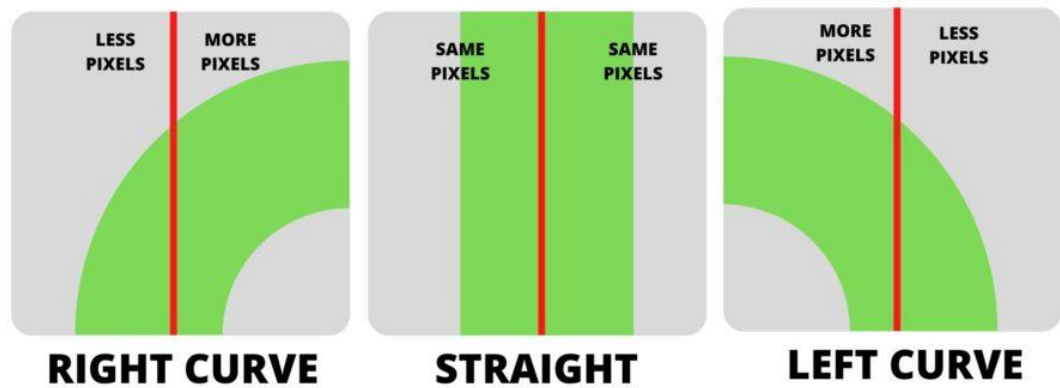


Fig. 9. Road Curvature Decision

On the histogram it looks like the following figure 10 [9]

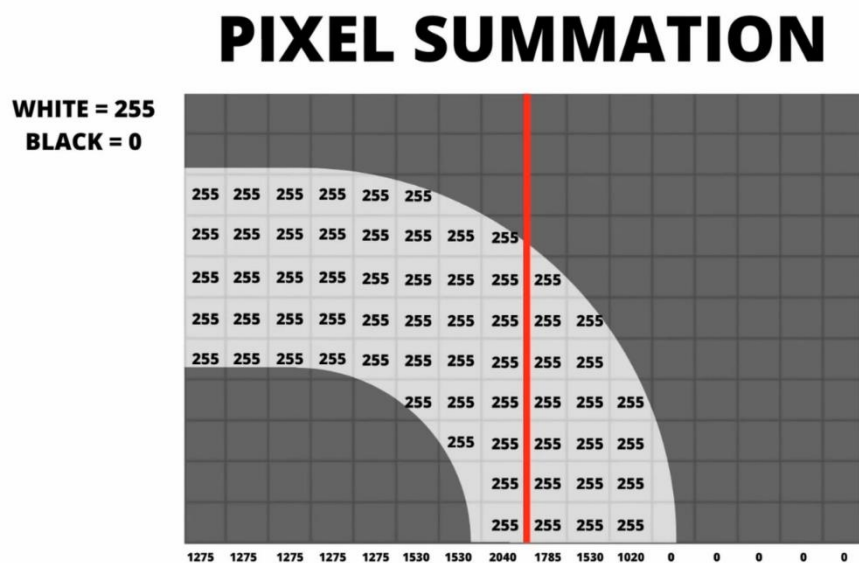


Fig. 10. Pixel summation in Histogram for left curve.

Actual results are shown in following figure 11



Fig. 11. Left side is road image. Right side is histogram and average base point.

The decision making process depends on the curve value. It is showed in following figure 12

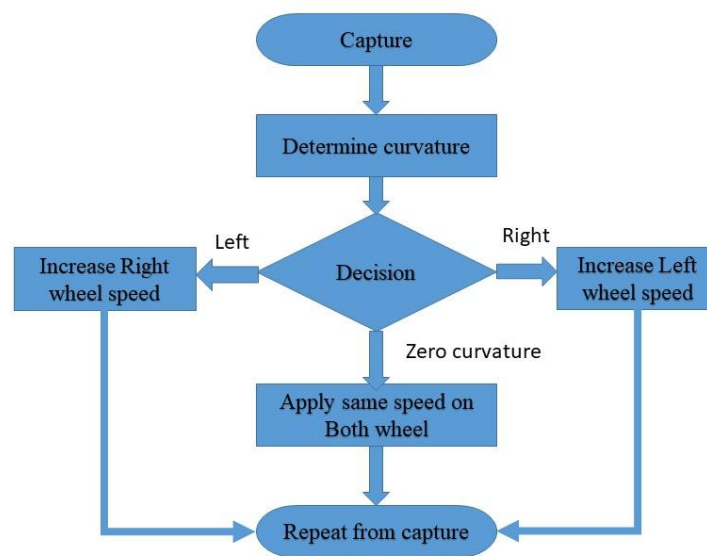


Fig. 12. Decision Making Process Flowchart

The remote control system is implemented separately for simplicity. This requires only two software modules. In a loop any keypress is continuously scanned and recorded. UP, RIGHT and LEFT button is set for matching. If any these three button is pressed, small movement is made via motor control. Keypress is continuously monitored. If no button is pressed down then the car is stopped. This is shown in following figure 13

It is a credit card sized, full-featured computer. It has on-board micro-processor and Wi-Fi. Its small size is preferable for portable use. It can process complex programs and algorithms. In our project it's complex computation capability is used to process images. Also remote communication is easily established through Wi-Fi.

b. Camera

Here Shown in figure 15 [11]

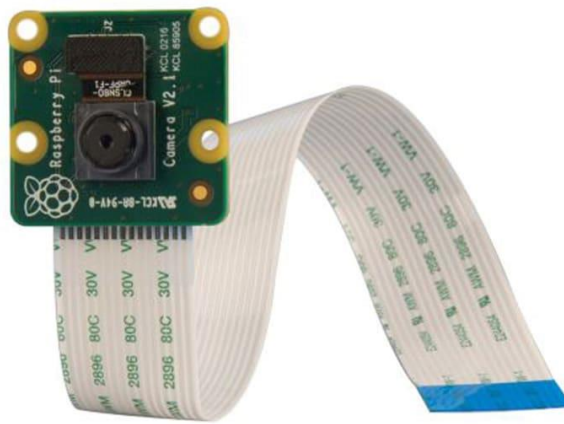


Fig. 15. Pi Camera

Camera Module uses a eight megapixel sensor made by sony. The Camera can take high-definition video, as well as stills. It uses 15cm ribbon cable. Connects to CSI port on the Raspberry Pi. It works with every models of Raspberry Pi 1, 2, 3 and 4. Accessed through the MMAL and V4L APIs. Having numerous libraries make it very much flexible in operation.

c. 18650 Li-ion Battery. Shown in figure 16 [12]



Fig. 16. 18650 Battery

The 18650 is one kind of lithium-ion battery. It is superior for large capacity, high current, portable size and low cost. Its nominal voltage is 3.7v and generally holds more than 2000mAh of charge.

d. Voltage Regulator

Shown in figure 17 [13]

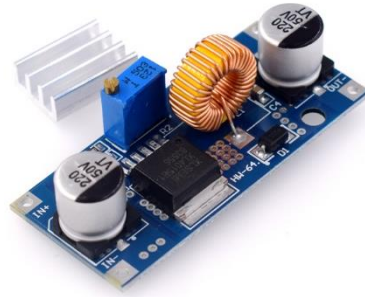


Fig. 17. XL4015 Power Supply Module

XL4015 dc-dc adjustable step down power supply module is 180 KHz fixed frequency PWM step-down (buck) DC/DC converter, capable of driving a 5A load with high efficiency, low ripple and excellent line and load regulation.

e. Breadboard

Shown in figure 18 [14]

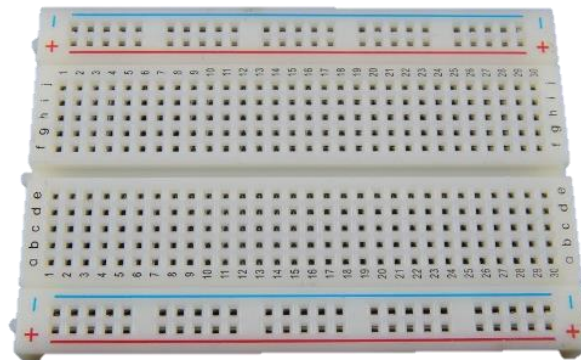


Fig. 18. Breadboard

Breadboard is useful in prototyping. It is easy to change connections for testing. It is generally used in prototype phase of development. When design is finalized, breadboard is replaced with Veroboard and permanent soldered connection.

f. L293d Motor driver IC

Shown in figure 18 [15]

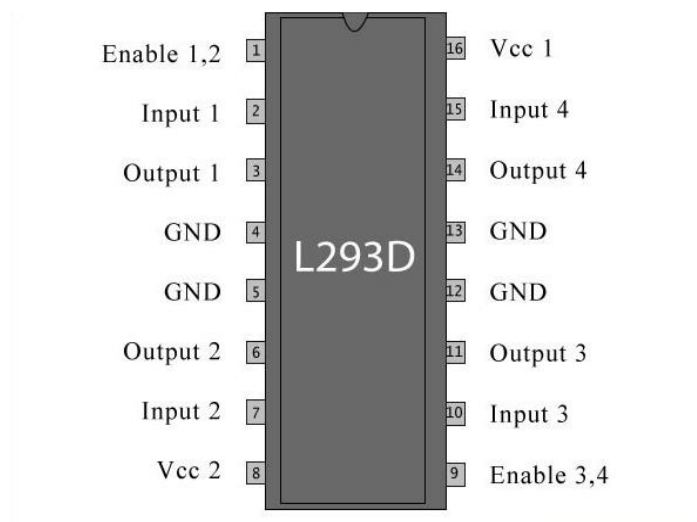


Fig. 19. L293d Motor Driver IC

Wheel motors require relatively high current. As Raspberry pi cannot handle high current and to protect it from reverse voltage, this motor driver IC is used.

g. Geared Motor, Wheel and custom-made Chassis

Shown in figure 18 [16]



Fig. 20. Car Chassis

The chassis includes two rear wheels which are powered and a castor ball acting as front wheel. The design is a tri wheeled simple vehicle. The chassis is laser cut.

Chapter 04

Results

The car is working properly on the custom made path in controlled environment. The road is differently colored than the environment. Actual car functional [Fig. 10]

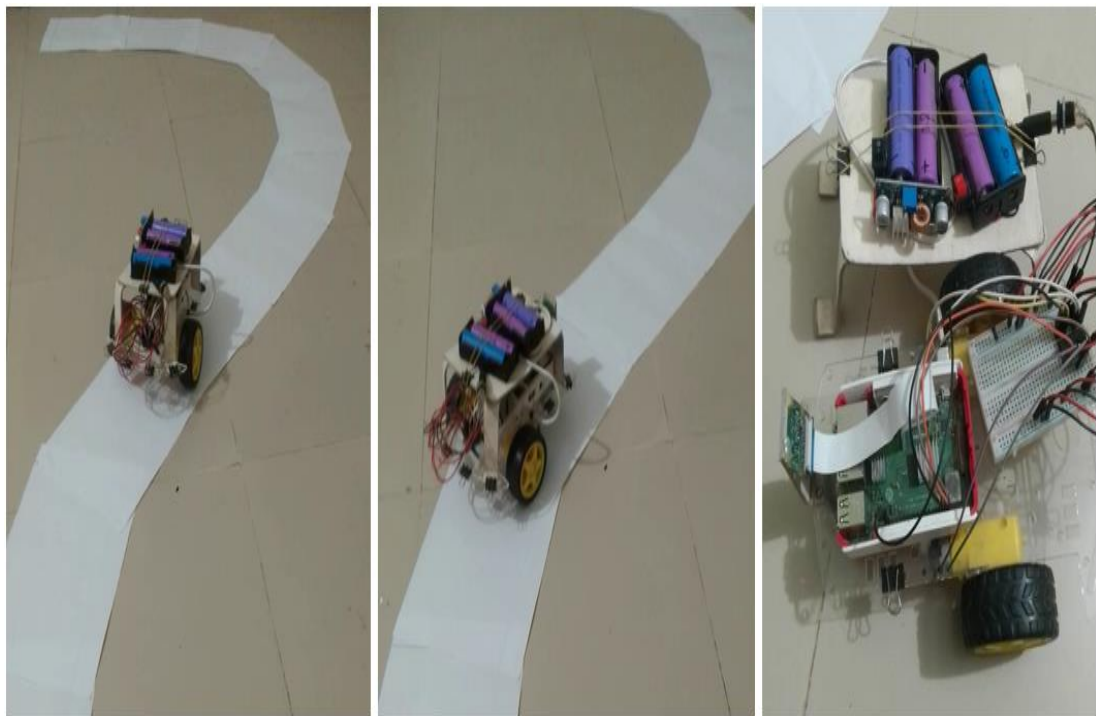


Fig. 21 Actual car photo

1. Automatic driving

The car follows the road. If the road take a curve it also follows the curve. We can achieve a high speed run when the road is mostly straight. White color is tested correctly. In a wide curve, the car gets off the road but it follows the curve. The car is not able to follow a very steep curve less than 45 degrees.

There are some limitations in this setup. The car's speed is reduced voluntarily to synchronize image processing time to motor actuation time. As raspberry pi does not have a powerful processor, it is slow to process images. Thus the overall process lags. But this is mitigated by multiple control segments in program code. Thus it is a good

approach for reliable driving on road and it secures the driver and passengers from fatal accidents. As it's drive system is designed as rear wheel drive, it is similar to most of the cars in Bangladesh. Rear wheel drive system made the car robust and the control system remains simple enough to maneuver in simple setups. Reducing four wheels to only two powered wheels makes car more energy efficient and cost effective. This design allows car manufacturers to limit their cost at minimum and makes the car affordable to all class of people in the country. Thus everyone will be able to afford a car by him/herself. This will enable our country to advance rapidly in conjunction with the rest of the world. Also energy efficiency means less fuel requirement and making the economy stronger in the long run. Energy efficiency keeps our planet mother earth livable for more coming years and supports our survivability.

2. Remote Control

A wireless keyboard connected with Raspberry pi is used as remote controller. As the preselected key is pressed, the computer process the scan and commands the motor to make a small move. Forward, backward, left turn and right turn is working correctly. As the processing is fast, movement seems continuous. The remote control system makes the system applicable for search and rescue type processes. This will enable police, army and fire brigade personnel to use it in emergency situations where human cannot go but needs active control over the process. The remote control design is not only limited to car system but it is also applicable to any type of moving vehicle. The remote control system can be adjusted to function with ship, aircraft or even crane controls. It can add as many directional control switch or valve to the remote for adjusting any number of direction needed for advanced controls.

Conclusion

Using PID controller will reduce jerking movement. This will make the movement smoother. More sensors and road sign recognition can be added. But this requires simultaneously combining all the data and decisions. In this case multi-threading is required. With its limitations, it achieves its goal of following the road. This car is capable of more image processing techniques. This is great for experimental exercises. Also it can be used in laboratory environments. Where we can implement virtual realistic augmentation. Our work's superiority lies in its simplicity and effectiveness.

The remote control scans for button press separately, thus cannot make both forward and side movement simultaneously. Also the controller does not provide information about variable speed. This requires controller with variable input support and more complex programming. There is a limitation of range. It is determined by the wireless technology used in keyboard controller. If internet based remote control is used, then there would be no limitation of range but then another limitation will be added. That is internet coverage. This can be mitigated by satellite communication.

Also the speed of the car can be increased by using powerful computer. Although the car is now operating with usable functionality. These are proposed for future works.

As automobile industry advances in Bangladesh, car technologies are going to be one of the highest grossing sector. Thus cooperation from government, public and private sector is highly recommended.

References

- [0] (Source: Hirz, Mario & Walzel, Bernhard. (2018). Sensor and object recognition technologies for self-driving cars. *Computer-Aided Design and Applications*. 15. 1-8. 10.1080/16864360.2017.1419638.)
- [1] Hummel, R., "Image enhancement by histogram transformation", ieht.rept, 1975.
- [2] J. P. Gonzalez and U. Ozguner, "Lane detection using histogram-based segmentation and decision trees," ITSC2000. 2000 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.00TH8493), Dearborn, MI, USA, 2000, pp. 346-351, doi: 10.1109/ITSC.2000.881084.
- [3] P. Pramkeaw and C. Thongchaisuratkrul, "Automated Driving System of Lane Detection Using Image Processing Techniques", *International Journal of the Computer, the Internet and Management* Vol.27 No.1 (January-April, 2019) pp. 58-63
- [4] Gurjashan Singh Pannu, Mohammad Dawud Ansari and Pritha Gupta. Article: Design and Implementation of Autonomous Car using Raspberry Pi. *International Journal of Computer Applications* 113(9):22-29, March 2015
- [5] N. P. Pawar and M. M. Patil "Driver Assistance System based on Raspberry Pi", *International Journal of Computer Applications* (0975 – 8887), Volume 95– No.16, June 2014
- [6] Raja Muthalagu, Anudeepsekhar Bolimera, V. Kalaichelvi, Lane detection technique based on perspective transformation and histogram analysis for self-driving cars, *Computers & Electrical Engineering*, Volume 85, 2020, 106653, ISSN 0045-7906, <https://doi.org/10.1016/j.compeleceng.2020.106653>. ([https:// www. Scienced i r e c t.com/science/ article/ pii/S0045790620305085](https://www.sciencedirect.com/science/article/pii/S0045790620305085))
- [7] B. Blaga, M. Deac, R. W. Y. Al-doori, M. Negru and R. Dănescu, "Miniature Autonomous Vehicle Development on Raspberry Pi," 2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing (ICCP), Cluj-Napoca, Romania, 2018, pp. 229-236, doi: 10.1109/ICCP.2018.8516589.

[8] Page URL:

https://commons.wikimedia.org/wiki/File:HSV_color_solid_cylinder.png, file URL: https://upload.wikimedia.org/wikipedia/commons/4/4e/HSV_color_solid_cylinder.png, attribution: SharkD, CC BY-SA 3.0 <<https://creativecommons.org/licenses/by-sa/3.0/>>, via Wikimedia Commons. Date taken: 14 April,2021

[9] Date taken: 14 April,2021. URL:<https://www.murtazahassan.com/courses/self-driving-car-using-raspberry-pi/lesson/finding-curve/>

[10] Date taken: 14 April,2021. URL:<https://grabcad.com/library/raspberry-pi-4-model-b-1>

[11] Date taken: 14 April,2021. URL:<https://www.raspberrypi.org/products/camera-module-v2/>

[12] Date taken: 14 April,2021. URL:<https://www.electronics.com.bd/Battery-5000mAh-Rechargeable-Li-Ion-18650>

[13] Date taken: 14 April,2021. URL:<https://www.electrodragon.com/product/dc-dc-step-down-adj-power-module-xl4015-4-38v-5a-96/>

[14] Date taken: 14 April,2021. URL:<https://roboindia.com/store/small-bread-board-self-adhesive-robo-india>

[15] Date taken: 14 April,2021.

URL:<https://sites.google.com/site/projectsriza/controlling-a-dc-motor-using-l293d-motor-driver-board>

[16] Date taken: 14 April,2021. URL:<https://www.probots.co.in/robot-chassis-2-wheel-drive-frame-with-motors-diy-car-kit-2wd.html>

Appendix

Here used codes are given bellow

Gpiozero_robot.py

```

"""
Motor pins are hardcoded in this module.
The car will take very small turn and correct direction. Then go
forward.
Test to find out usable minimum values for speed and duration.
Usable function:
    move()
    stopping()
"""

from gpiozero import Robot
import time

# ===== Minimum speed and duration values start
# todo: Need to find out by testing. Use minimum usable value
min_turn_speed = 0.4
turn_duration = 0.0

min_forward_speed = 0.5
forward_duration = 0.1
# ===== Minimum speed and duration values end

# ===== Motor pin definition
robot = Robot(left=(4, 14), right=(17, 18))

def move(curve):
    """
    Runs the motors to go forward or turn left/right
    @ min speed and for min duration.
    Then stops the car.

    input: curve value
    output: prints direction of movement
    """
    if curve > 0.0:
        robot.right(min_turn_speed)
        print("            right\n")
        time.sleep(turn_duration)

    elif curve < 0.0:

```

```

        robot.left(min_turn_speed
        time.sleep(turn_duration)
        print("left\n")

    elif curve == 0:
        robot.forward(min_forward_speed)
        print("        forward        \n")
        time.sleep(forward_duration)

    robot.stop()

def stopping():
    """stops the car"""
    robot.stop()

```

LaneDetectionModule.py

```

import cv2
import numpy as np
import utlis

curveList = []
avgVal = 10

def getLaneCurve(img, display=2):
    """
    display options:
        0: Nothing will be displayed in the window
        1: Show only resulted image (marked by detected lane)
        2: Shows full pipeline with six frames stacked from
different phases
    Total 5 steps
    """
    imgCopy = img.copy()
    imgResult = img.copy()

    # ===== STEP 1
    """ Produce mask; min and max hsv value range is hardcoded in
utlis module."""
    imgThres = utlis.thresholding(img)

    # ===== STEP 2
    """Region of interest will be warped and resized
to the size of original image"""
    hT, wT, c = img.shape # the order is Height, width and
channel/depth
    points = utlis.valTrackbars() # trackbar initialized with
values in main function

    # order of W and H can be arbitrary but must be consistent
across all functions
    # wT is width Target and hT is height Target

```

```

imgWarp = utlis.warpImg(imgThres, points, wT, hT)
imgWarpPoints = utlis.drawPoints(imgCopy, points)
# ===== STEP 3
middlePoint, imgHist = utlis.getHistogram(imgWarp,
display=True, minPer=0.5, region=4)
curveAveragePoint, imgHist = utlis.getHistogram(imgWarp,
display=True, minPer=0.9)
curveRaw = curveAveragePoint - middlePoint

# ===== STEP 4
curveList.append(curveRaw)
if len(curveList) > avgVal:
    curveList.pop(0)
curve = int(sum(curveList) / len(curveList))

# ===== STEP 5
if display == 0:
    # Do nothing
    pass
else:
    # if display = 0, then display related nothing will execute
    # in any other case, it will execute

    # Inverse warped image to get original unwrapped portion
    imgInvWarp = utlis.warpImg(imgWarp, points, wT, hT,
inv=True)
    imgInvWarp = cv2.cvtColor(imgInvWarp, cv2.COLOR_GRAY2BGR)
# convert to 3 channel to match in stacking
imgInvWarp[0:hT // 3, 0:wT] = 0, 0, 0
imgLaneColor = np.zeros_like(img)
imgLaneColor[:] = 0, 255, 0
imgLaneColor = cv2.bitwise_and(imgInvWarp, imgLaneColor)
imgResult = cv2.addWeighted(imgResult, 1, imgLaneColor, 1,
0)

    midY = 450
    cv2.putText(imgResult, str(curve), (wT // 2 - 80, 85),
cv2.FONT_HERSHEY_COMPLEX, 2, (255, 0, 255), 3)
    cv2.line(imgResult, (wT // 2, midY), (wT // 2 + (curve *
3), midY), (255, 0, 255), 5)
    cv2.line(imgResult, ((wT // 2 + (curve * 3)), midY - 25),
(wT // 2 + (curve * 3), midY + 25), (0, 255, 0), 5)
    for x in range(-30, 30):
        w = wT // 20
        cv2.line(imgResult, (w * x + int(curve // 50), midY -
10),
(w * x + int(curve // 50), midY + 10), (0, 0,
255), 2)
    # fps = cv2.getTickFrequency() / (cv2.getTickCount() -
timer);
    # cv2.putText(imgResult, 'FPS ' + str(int(fps)), (20, 40),
cv2.FONT_HERSHEY_SIMPLEX, 1, (230, 50, 50), 3);
    if display == 1:
        cv2.imshow('Result', imgResult)
    elif display == 2:
        imgStacked = utlis.stackImages(0.7, ([img,
imgWarpPoints, imgWarp],
[imgHist,
imgLaneColor, imgResult]))

```

```

        cv2.imshow('ImageStack', imgStacked)
    # ===== NORMALIZATION
    curve = curve / 100
    if curve > 1:
        curve = 1
    if curve < -1:
        curve = -1

    return curve

if __name__ == '__main__':
    # cap = cv2.VideoCapture('vid1.mp4')
    cap = cv2.VideoCapture(0)    # use camera
    intialTrackBarVals = [102, 160, 20, 240]
    utlis.initializeTrackbars(intialTrackBarVals)

    frameCounter = 0
    while True:
        # continuous loop
        frameCounter += 1
        if cap.get(cv2.CAP_PROP_FRAME_COUNT) == frameCounter:
            cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
            frameCounter = 0

        success, img = cap.read()
        img = cv2.resize(img, (480, 240))
        curve = getLaneCurve(img, display=2)
        print("curve = " + str(curve))
        # cv2.imshow('Vid',img)
        # cv2.waitKey(1)
        if cv2.waitKey(1) & 0xff == ord('q'):
            break

    cap.release()
    cv2.destroyAllWindows()

```

MainRobotLane.py

""" This file contains gpiozero and movement codes
which is related to Pi.
So can be run in Pi. But can not be run in windows.

Use different maxVal (aka max speed) for battery(0.7) and adapter(1)

```

"""
# ===== Imports Start
import cv2

from LaneDetectionModule import getLaneCurve
import WebcamModule
import utlis

import gpiozero_robot

```



```

# ===== Imports End
# ===== Main Function Start
def main():
    img = WebcamModule.getImg()
    curveVal = getLaneCurve(img, 0)

    # sen = 1.3 # SENSITIVITY # disabled for simplicity

    # lithium battery can deliver high current, so use value lower
    # than 1,
    # adapter delivers very low current(180mA), so use 1(full
    # available speed)
    maxVal = 0.7 # MAX SPEED
    if curveVal > maxVal:
        curveVal = maxVal
    if curveVal < -maxVal:
        curveVal = -maxVal
    print("curve value = " + str(curveVal))
    if curveVal > 0: # Deadzone, if in this -0.08 to 0.05
        then no turning
        # sen = 1.7
        if curveVal < 0.05:
            curveVal = 0
    else:
        if curveVal > -0.08:
            curveVal = 0

    gpiozero_robot.move(curveVal)

# ===== Main Function End

# ===== If this module is run
if __name__ == '__main__':
    intialTrackBarVals = [102, 160, 20, 240]
    utlis.initializeTrackbars(intialTrackBarVals)
    while True:
        main()
        """
        To preview, waitKey is necessary.
        bitwise AND will be evaluated first.
        ord() returns unicode number for character.
        """
        if cv2.waitKey(1) & 0xff == ord('q'):
            gpiozero_robot.stopping()
            cv2.destroyAllWindows()
            break

```

MotorModule.py

```

import RPi.GPIO as GPIO
from time import sleep
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

```

```

class Motor():
    def __init__(self, EnaA, In1A, In2A, EnaB, In1B, In2B):
        self.EnaA= EnaA
        self.In1A = In1A
        self.In2A = In2A
        self.EnaB= EnaB
        self.In1B = In1B
        self.In2B = In2B

GPIO.setup(self.EnaA,GPIO.OUT);GPIO.setup(self.In1A,GPIO.OUT);GPIO
O.setup(self.In2A,GPIO.OUT)

GPIO.setup(self.EnaB,GPIO.OUT);GPIO.setup(self.In1B,GPIO.OUT);GPI
O.setup(self.In2B,GPIO.OUT)
        self.pwmA = GPIO.PWM(self.EnaA, 100);
        self.pwmB = GPIO.PWM(self.EnaB, 100);
        self.pwmA.start(0);
        self.pwmB.start(0);
        self.mySpeed=0

    def move(self, speed=0.5, turn=0, t=0):
        speed *= 100
        turn *= 70          # todo: starnge value, need to check
video
        leftSpeed = speed-turn
        rightSpeed = speed+turn

        if leftSpeed>100: leftSpeed =100
        elif leftSpeed<-100: leftSpeed = -100
        if rightSpeed>100: rightSpeed =100
        elif rightSpeed<-100: rightSpeed = -100
        # print(leftSpeed,rightSpeed)
        self.pwmA.ChangeDutyCycle(abs(leftSpeed))
        self.pwmB.ChangeDutyCycle(abs(rightSpeed))
        if
leftSpeed>0:GPIO.output(self.In1A,GPIO.HIGH);GPIO.output(self.In2
A,GPIO.LOW)

else:GPIO.output(self.In1A,GPIO.LOW);GPIO.output(self.In2A,GPIO.H
IGH)
        if
rightSpeed>0:GPIO.output(self.In1B,GPIO.HIGH);GPIO.output(self.In
2B,GPIO.LOW)

else:GPIO.output(self.In1B,GPIO.LOW);GPIO.output(self.In2B,GPIO.H
IGH)
        sleep(t)

    def stop(self,t=0):
        self.pwmA.ChangeDutyCycle(0);
        self.pwmB.ChangeDutyCycle(0);
        self.mySpeed=0
        sleep(t)

def main():
    motor.move(0.5, 0, 2)
    motor.stop(2)

```

```

        motor.move(-0.5, 0, 2)
        motor.stop(2)
        motor.move(0, 0.5, 2)
        motor.stop(2)
        motor.move(0, -0.5, 2)
        motor.stop(2)

if __name__ == '__main__':
    motor= Motor(2,3,4,17,22,27)
    main()

```

utils.py

```

""" Utils.py
This module includes all extra functions used in
LaneDetectionModule.py .
There are 8 functions in it.
"""

```

```

import cv2
import numpy as np

```

```

def thresholding(img):
    """Gives mask for specific color
    Chosen color is hardcoded.
    input: original RGB image
    output: scalar(single channel) mask, (1 for matched pixels, 0
for otherwise.)
    """
    imgHsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    lowerWhite = np.array([80, 0, 0])
    upperWhite = np.array([255, 160, 255])
    maskWhite = cv2.inRange(imgHsv, lowerWhite, upperWhite)
    return maskWhite

```

```

def warpImg(img, points, w, h, inv=False):
    """change to birds eye view
    input: img      = src
           points   = the points which will be warped
           w        = final width
           h        = final height
    point is specified as first width then height, like (w,h)=(0,0)
    Order:top-left, top-right, bottom-left, bottom-right
    """
    pts1 = np.float32(points)          # src points
    pts2 = np.float32([[0, 0], [w, 0], [0, h], [w, h]])  # dst
    points
    if inv:
        matrix = cv2.getPerspectiveTransform(pts2, pts1)
    else:
        matrix = cv2.getPerspectiveTransform(pts1, pts2)
    imgWarp = cv2.warpPerspective(img, matrix, (w, h))
    return imgWarp

```

```

def nothing(a):
    """This null function is required for trackbar function"""
    pass

def initializeTrackbars(intialTracbarVals, wT=480, hT=240):
    """
    wT is width Target and hT is Height target
    Point is specified as first width then height, like (w,h)=(0,0)
    Order:top-left, top-right, bottom-left, bottom-right
    """
    cv2.namedWindow("Trackbars")
    cv2.resizeWindow("Trackbars", 360, 240)
    cv2.createTrackbar("Width Top", "Trackbars",
intialTracbarVals[0], wT // 2, nothing)
    cv2.createTrackbar("Height Top", "Trackbars",
intialTracbarVals[1], hT, nothing)
    cv2.createTrackbar("Width Bottom", "Trackbars",
intialTracbarVals[2], wT // 2, nothing)
    cv2.createTrackbar("Height Bottom", "Trackbars",
intialTracbarVals[3], hT, nothing)

def valTrackbars(wT=480, hT=240):
    """
    point is specified as first width then height, like (w,h)=(0,0)
    Order:top-left, top-right, bottom-left, bottom-right
    """
    widthTop = cv2.getTrackbarPos("Width Top", "Trackbars")
    heightTop = cv2.getTrackbarPos("Height Top", "Trackbars")
    widthBottom = cv2.getTrackbarPos("Width Bottom", "Trackbars")
    heightBottom = cv2.getTrackbarPos("Height Bottom",
"Trackbars")
    points = np.float32([(widthTop, heightTop), (wT - widthTop,
heightTop),
                        (widthBottom, heightBottom), (wT -
widthBottom, heightBottom)])
    return points

def drawPoints(img, points):
    """Used to draw four points to indicate Region of Interest"""
    for x in range(4):
        cv2.circle(img, (int(points[x][0]), int(points[x][1])),
15, (0, 0, 255), cv2.FILLED)
    return img

def getHistogram(img, minPer=0.1, display=False, region=1):
    if region == 1:
        histValues = np.sum(img, axis=0)
    else:
        histValues = np.sum(img[img.shape[0] // region:, :],
axis=0)

    # print(histValues)
    maxValue = np.max(histValues)

```

```

minValue = minPer * maxValue

indexArray = np.where(histValues >= minValue)
basePoint = int(np.average(indexArray))
# print(basePoint)

if display:
    imgHist = np.zeros((img.shape[0], img.shape[1], 3),
np.uint8)
    for x, intensity in enumerate(histValues):
        cv2.line(imgHist, (x, img.shape[0]), (x, img.shape[0]
- intensity // 255 // region), (255, 0, 255), 1)
        cv2.circle(imgHist, (basePoint, img.shape[0]), 20, (0,
255, 255), cv2.FILLED)
    return basePoint, imgHist

return basePoint

def stackImages(scale, imgArray):
    rows = len(imgArray)
    cols = len(imgArray[0])
    rowsAvailable = isinstance(imgArray[0], list)
    width = imgArray[0][0].shape[1]
    height = imgArray[0][0].shape[0]
    if rowsAvailable:
        for x in range(0, rows):
            for y in range(0, cols):
                if imgArray[x][y].shape[:2] ==
imgArray[0][0].shape[:2]:
                    imgArray[x][y] = cv2.resize(imgArray[x][y],
(0, 0), None, scale, scale)
                else:
                    imgArray[x][y] = cv2.resize(imgArray[x][y],
(imgArray[0][0].shape[1], imgArray[0][0].shape[0]),
None, scale,
scale)
                    if len(imgArray[x][y].shape) == 2: imgArray[x][y]
= cv2.cvtColor(imgArray[x][y], cv2.COLOR_GRAY2BGR)
                    imageBlank = np.zeros((height, width, 3), np.uint8)
                    hor = [imageBlank] * rows
                    hor_con = [imageBlank] * rows
                    for x in range(0, rows):
                        hor[x] = np.hstack(imgArray[x])
                    ver = np.vstack(hor)
            else:
                for x in range(0, rows):
                    if imgArray[x].shape[:2] == imgArray[0].shape[:2]:
                        imgArray[x] = cv2.resize(imgArray[x], (0, 0),
None, scale, scale)
                    else:
                        imgArray[x] = cv2.resize(imgArray[x],
(imgArray[0].shape[1], imgArray[0].shape[0]), None, scale, scale)
                        if len(imgArray[x].shape) == 2: imgArray[x] =
cv2.cvtColor(imgArray[x], cv2.COLOR_GRAY2BGR)
                        hor = np.hstack(imgArray)
                        ver = hor
    return ver

```

WebcamModule.py

```
import cv2

cap = cv2.VideoCapture(0)

def getImg(display= False, size=[480,240]):
    _, img = cap.read()
    img = cv2.resize(img, (size[0], size[1]))
    if display:
        cv2.imshow('IMG', img)
    return img

if __name__ == '__main__':
    while True:
        img = getImg(True)
```