Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 02, February-2020

Self-Driving Car Model using Raspberry Pi

Ms. Pratibha I Golabhavi
Department of Electrical Engineering
University Visvesvaraya College of Engineering,
Bengaluru, India

B P Harish
Department of Electrical Engineering
University Visvesvaraya College of Engineering,
Bengaluru, India

***Abstract* - Self-driving cars are autonomous vehicles that can drive by themselves without any human interference and has the potential to mark the technological revolution of the next decade. This work presents the development of a low-cost prototype of a miniature self-driving car model using simple and easily available technologies. The objective of the work is to avoid accidents caused due to driver faults. In this prototype, Raspberry Pi controller and H-bridge drives two DC motors to realize vehicle automation. Technologies such as GPS to access location, sonar sensors for obstacle detection and avoidance, image processing for pedestrian detection, computer vision for processing images and machine learning for intelligent systems have been deployed. If this car meets with an accident, it is detected with an accelerometer and an alert message and live location are sent to the registered mobile number and E-mail of a contact person using Geocoding through Twilio application. These self-driving cars are proposed as accident-free and on-time urban transportation solution and in military applications.**

Keywords—Raspberry Pi, Lane detection, Traffic light detection, Convolutional Neural Network, Single Shot Detector, MobileNet, Accident Alert, Accelerometer.

I. INTRODUCTION

Road accidents are a global challenge due to which nearly 1.25 million people die in road accidents globally every year, or on an average 3287 deaths per day and 20-50 million people get injured and disabled for life annually, as reported by World Health Organization (WHO). International Road Federation (IRF), Geneva report says that India has the highest number of road deaths and ranking first while accounting for 10% of global road accidents. It is found that 78% of road accidents are due to driver's fault or human error in driving and it is interesting to address this human issue with deployment of technology in the form of "Driverless Car".

With rapid advancements in technology, scientists are proposing new ideas to build "self-driving car" in order to ensure accident-free transportation. Companies like Google, Uber and Tesla are leading the global initiative in the design and manufacture of an autonomous car. This helps in significant savings in time as we can work even as the car is driving by itself in city traffic during rush hours to the office. A CNN based SSD-MobileNet technique is proposed to detect and track the real world objects. This prototype has achieved excellent results of detection and tracking of the trained objects with 99% accuracy at 98.2% confidence level, which helps the robot to avoid accidents [1]. An obstacle detection and avoidance robot is presented based on Hough transform algorithm of object detection using Java programming and controlled by a Microcontroller. The robot navigates on its own, avoids obstacles, detects and picks up a different colour balls [2]. A mathematical model is proposed to explore the characteristics of a two-wheel self-balancing robot and control its behaviour on a planar surface and on an inclined plane [3]. An accident alert system for motor cycles is proposed which sends out the accident information to emergency services to initiate rescue operation and save lives. The device includes a small embedded unit mounted at the safest place of the vehicle and an emergency switch for use during a medical emergency [4]. A lane detection system based on Raspberry Pi and Arduino controller is proposed with two subsystems: object detection and image processing. Image processing is used for lane detection with the camera mounted on the prototype capturing the image and image data is extracted to generate the necessary command for the obstacle detection system. This system has a disadvantage of not detecting lane accurately as the model goes little out of lane [5]. A real time traffic sign recognition system is implemented using traffic sign detection and traffic light recognition algorithms. While a u-Eye camera is used to capture images, the system suffers from instability with changes in ambient light [6]. An accident detection system is proposed to monitor the vehicle constantly and detect if the vehicle is in normal posture or has fallen down. When the vehicle fall is detected, the body condition or heartbeat rate of the driver is checked. If any abnormality is found, the message is sent to the nearest hospital [7].

In this direction, a prototype of driverless car is developed which senses the ever-changing traffic conditions on road and takes appropriate decisions for navigation dynamically, without any human input.

The remainder of the paper is organized as follows: Section II presents the design of architecture of self-driving car with its hardware and software organization. Section III describes the subsystems of the self-driving car and their function. While Section IV presents results and discussion, Section V concludes with some pointers to future work.

II. DESIGN OF ARCHITECTURE OF SELF-DRIVING CAR

The block diagram of the self-driving car system is shown in Fig. 1.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 02, February-2020

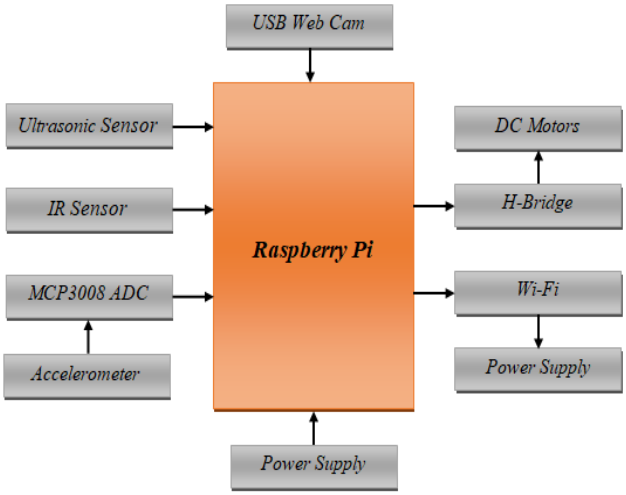

Fig. 1: Block Diagram of the Self-Driving Car model

A set of sensors like ultrasonic sensor, IR sensors, and the L293D H-bridge motor driver are connected to Raspberry Pi 3 controller through its General Purpose Input Output (GPIO) pins. Web camera is connected to USB port of Raspberry Pi board. The analog output of accelerometer sensor is converted to digital using IC MCP3008 Analog to Digital converter and then fed to GPIOs of Raspberry Pi 3. In order to build a hardware model of self-driving car, a chassis is selected as the base on which all boards are mounted and 4 wheels are attached - 2 wheels in front and 2 wheels in back, to the chassis. Front wheels are powered with two dc motors running at 30 RPM. H-bridge driver circuit controls the movement of these motors in clockwise or anticlockwise direction upon receiving control signals from Raspberry Pi controller. Ultrasonic sensor in the front and IR sensors on left and right sides of the car detect obstacles around the car and measure the distance. IR sensor module in the front is responsible for lane following. Accelerometer sensor mounted on top of the car is connected to the controller through IC MCP3008 ADC, with its x-axis value set to detect the tilt of the driverless car. USB webcam is used for image processing along with Python library to detect real-time objects and to follow traffic rules. All the programs, written in Python language, to implement image processing algorithms are dumped into Raspberry Pi 3. Android mobile phone connected to the controller board through Wi-Fi is used as desktop to enable the mobile act as input device to give commands and run the programs. Using geocoder, Raspberry Pi sends out message and location link to the phone number and E-mail written in the code through Twilio.

The Android OS based mobile phone of the user is setup to view the desktop of Raspberry Pi and control the car anytime and anywhere, using mobile hotspot and the programming is done in Python using Integrated Development and Learning Environment (IDLE). The mobile applications VNC Viewer, Mobile SSH and Network Scanner are deployed for remote control of user mobile by Raspberry Pi server on a client-server based model.

All functionalities of the self-driving car like Lane Detection and Following System (LDFS), Traffic Light Detection System (TLDS), Real-Time Object Detection System (RTODS), Accident Alert System are selected by keying in a code on the user mobile.

III. SUB-SYSTEMS OF SELF-DRIVING CAR

A. Lane Detection and Following System (LDFS)

In order to achieve the main objective of lane detection and tracking, a self-driving car should be able to detect, track and differentiate various roads for its proper movement on road. The LDFS consists of 3 IR sensors (IR1, IR2 and IR3) mounted on the self-driving car and connected to the Raspberry Pi controller to detect the position of the car relative to the yellow line marked at the center of the road. The process flow of LDFS is as shown in the Fig. 2.

In the proposed self-driving car, a lane on the road is designed with a yellow line drawn at its middle for the car to follow. When IR1 and IR3 are low, black lane is detected and when IR2 is at high, yellow line is detected. The mobile of the user is programmed to act as the control device to start and control the motion of the car. The user enters the car and keys in '1' on his mobile and all the sensors on the car get activated and the motor turns on. Once IR2 detects yellow color line car starts moving forward. When the car tends to shift to right, IR1 and IR2 both detect yellow line and the car moves to the left until IR2 alone senses yellow. When the car tends to shift to left, IR2 and IR3 detect yellow line and the car moves to the right until IR2 alone senses yellow. Thus the car is in a self-correcting mode to be at the center of the lane and keeps moving forward as long as IR2 alone is detecting yellow.

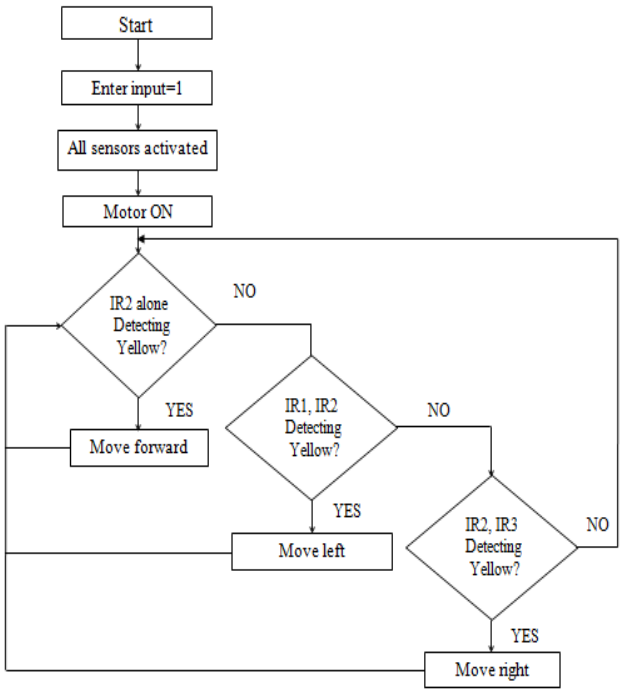

Fig. 2: Process Flow of LDFS

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 02, February-2020

A. *Traffic Light Detection System (TLDS)*

To detect and track red, yellow and green colors of the traffic light system, fundamentals of computer vision are used. The process flow diagram of Traffic Light Detection System (TLDS), shown in Fig. 3, illustrates how self-driving car recognizes traffic light and responds to it.

When user enters input 2 in his mobile, all sensors get activated, web camera starts capturing the video, motor turns on, and the car starts moving forward. Frames captured by a camera from continuous video streaming is sent to Raspberry Pi controller. The controller processes the traffic lights, based on image processing and traffic light detection algorithm. If any color is detected, then the color is identified. If color detected is red, car stops until it detects yellow or green. When yellow is detected, car moves bit forward and stops until green is detected. When green color is detected, the car moves forward guided by computer vision algorithm, illustrated in Fig. 4.

The input frames obtained from the camera is in BGR format which is converted from BGR color space to corresponding Hue Saturation Value (HSV). In OpenCV, range of values for Hue (representing color) is 0 - 179, range of values for Saturation (representing intensity/purity of color) it is 0 - 255, range of numbers for value (representing brightness of color) is 0 - 255. The range of colors to be tracked is defined as per the requirement and then morphological transformations are carried out on the color to remove noise present in the binary image. Then, contouring of the colors is done with a rectangular bounded line called 'contour' to differentiate between each color.

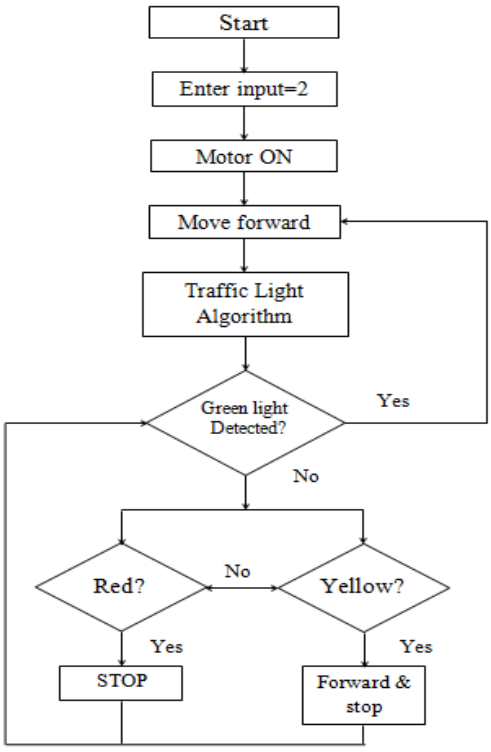

Fig. 3: Process Flow of TLDS

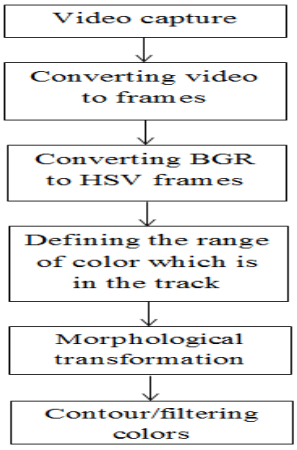

Fig. 4: Traffic Light Detection Algorithm

B. *Real- Time Object Detection System (RTODS)*

Object detection is at the core of many applications such as autonomous cars, security, surveillance and industrial applications. Selection of a right object detection method is critical depending on the nature of the problem to be solved. Single Shot Detector (SSD) is a good choice as it is able to run on a video and achieves a good tradeoff between speed and accuracy.

Real-Time Object Detection System (RTODS) is activated including all the sensors and camera by entering input = 3 in the mobile application. Raspberry Pi controller receives captured images from the webcam as input. On the received image, Raspberry Pi controller runs real-time object detection algorithm and sends the control signal to H-bridge which in turn drives the motor. Car starts moving forward, if any animal/person/objects are detected, car stops for 10 seconds and check for the presence of objects, if no object is detected, the car starts moving forward.

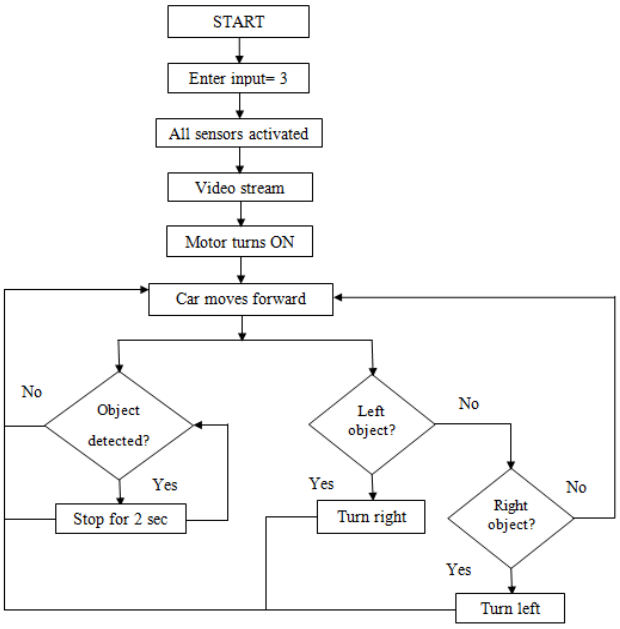

Fig. 5: Process Flow of RTODS

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 02, February-2020

The process flow of the real-time object detection system is shown in the Fig. 5. If the IR sensor mounted on the left detects the object, car should move right and vice versa. The ultrasonic sensor installed on the front detects and overcomes small objects and calculates its distance from the object dynamically. Real-Time Object Detection is implemented on Raspberry Pi using OpenCV Deep Neural Network (DNN) and MobileNet-SSD algorithm [1]. The algorithm is implemented in Python using OpenCV.

1. Data-sets
2. Feeding the image to the network
3. Feature Maps
4. Hard Negative Mining
5. Data Augmentation
6. Non-Maximum Suppression (NMS)

Training datasets and test datasets with ground truth bounding boxes like Caffe-datasets and assigned class labels are required for MobileNet-SSD algorithm. To feed image into the network, the image is converted into a blob, a pre-processed image that serves as the input. Pre-processing techniques like cropping, resizing, and color swapping are applied. Features maps represent the dominant features of the image at different scales. Therefore, running Multi-Box on multiple feature maps increases the size of any object to be detected. During training, it is followed to keep a ratio of negative to positive of around 3:1, instead of using all negative predictions to ensure that the network also needs to learn what constitutes an incorrect detection. The non-maximum suppression technique is applied to all bounding boxes and the top predictions are preserved. This ensures that the most likely predictions of object are retained by the network, while the noisy ones are removed.

C. Accident Alert System

An Accident Alert System is designed and developed to generate the location information of car when it meets with an accident like vehicle rollover or head-on collision. The process flow of Accident Alert System is shown in Fig. 6.

The accelerometer is mounted on top of the self- driving car to sense tilt in complete 3-axis. Accelerometer is fixed with a value in all 3- axes. The X-axis value is fixed at 500-530 for GY-61 ADXL335 accelerometer and its axis gets disturbed, when the self-driving car experiences accidents due to street conditions, run-off-road collisions, collisions with fallen debris, rollovers and collisions with animals. It is considered as an accident when X-axis value goes beyond 500-530. In that case, the accelerometer sends out analog voltage value to the IC MCP3008 ADC which converts analog voltage to corresponding digital number. The controller receives the digital value and sends out signal to H-bridge to stop the self- driving car and processes the data to get the live location of the place of accident using Geocoding and sends out a text message - "Accident is detected" and the location to the E-mail

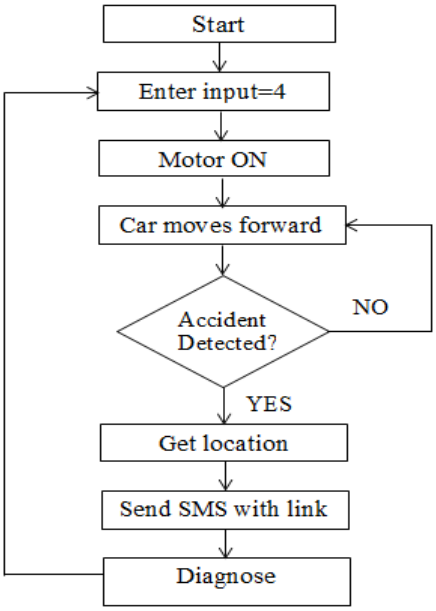

Fig. 6: The process flow of Accident Alert System

And phone number of the contact person through Twilio messaging application.

Accelerometer gives values in terms of gravity 'g'. Voltage at the midpoint 0 g (Earth gravity) is half of the supply voltage. Accelerometer operates at supply voltage, Vs= 3.3 V; Analog voltage value of 0-3.3 V corresponds to 0-1023 in digital format.

Sensitivity= 330 mV/g.

At 0g, $Vs = Vs/2 = 3.3/2 = 1.65V$ ……..…… (1)

The digital value is generated as:

Digital value = (Analog voltage obtained*Resolution) /3.3

= 0.5*(1023/3.3) = 155

For analog voltage of 1.65 V, digital value is 512.

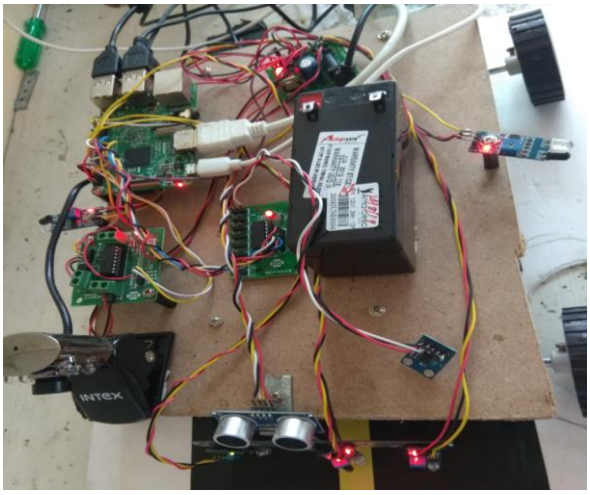

Fig. 7: Prototype of the self-driving car.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 9 Issue 02, February-2020

IV. RESULTS & DISCUSSION

A prototype of the self-driving car is developed and is shown in Fig. 7. The lane is designed and the lane detection and following system of the self-driving car is demonstrated and the results of Lane Detection and Following System are tabulated in Table 1. It is noted that logic 1 represents sensor detecting yellow and logic 0 represents sensor detecting black. The self-driving car is found to be successfully detecting traffic lights, and responding to it as follows: For red - car stops; for yellow - the car moves bit forward and stops; for green car moves forward. It is demonstrated that the self-driving car stops after detecting real-time objects, detects and overcomes the obstacles surrounding to it and measures the distance. When the self- driving car meets with an accident, the X-axis of an accelerometer deviates from a fixed value and an alert message is sent to the registered mobile number of the contact person through Twilio account successfully and the results are as shown in the Fig. 8.

Table 1: Results obtained from Lane Detection and Following System

IR1	IR2	IR3	MOVEMENT OF ROBOT
0	0	0	Reverse
0	0	1	Right
0	1	0	Forward
0	1	1	Right
1	0	0	left
1	0	1	reverse
1	1	0	left
1	1	1	reverse

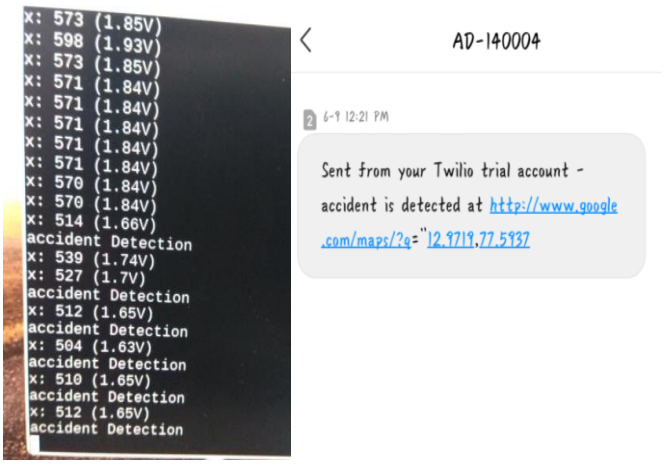

Fig. 7: Results obtained from Accident Alert System

V. CONCLUSIONS & FUTURE SCOPE

A. *Conclusions*

A low cost prototype of a Self-Driving Car model is designed developed and all functionalities are successfully demonstrated. The car is able to follow lane efficiently using IR sensor module and the traffic colors are detected and decisions are made by the car using image processing techniques to follow real-time traffic rules. Car is able to differentiate between real-time objects and is responding to the given instructions precisely and is detecting and overcoming obstacles successfully. Accident Alert System is designed and an alert message is sent to the mobile of a user in the event of an accident.

B. *Future scope*

While the lane following function of the proposed self-driving car model can be implemented using image processing, GPS can be used to track the live location of the self-driving car continuously for safety.

REFERENCES

[1] Chandan G, Ayush Jain, Harsh Jain and Mohana, “Real-time object detection and tracking using deep learning and OpenCV,” III International Conference on Inventive Research in Computing Applications (ICIRCA 2018).

[2] Karti Balasubramanian and Arunkumar, “Object recognition and obstacle avoidance robot,” Chinese Control Design & Conference (CCDC 2009).

[3] Pannaga R.M. and B.P. Harish, “Modelling and implementation of two-wheel self-balance robot,” International Journal of Electrical, Electronics and Computer Science Engineering, Vol. 4, Issue 6, pp. 33-40, December 2017.

[4] Fahim Bin Basheer, Jinu J Alias and Mohammad Favas C, “Design of accident detection and alert system for motor cycles,” International Conference on Automobile Engineering (IEEE 2013).

[5] Tiple Anjali Hemant, Tiple Hemanth P and Gauravsagar Hanumanth, “Prototype of autonomous car,” 2018 International Journal of Engineering Research in Electronics and Communication (IJERECE 2018).

[6] Tiago Moura, Antonio Valente and Vitorphilipe, “The traffic sign recognition for autonomous robot,” 2014 International Conference on Autonomous Robot System .and Competitions (ICARSC 2014).

[7] Nikky Kattukaran, Arun George, Mithun Haridas T P and Balakrishna M, “Intelligent accident detection and alert system for emergency medical assistance,” 2017 International Conference on Computer Communication and Informatics (ICCCI 2017).