

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Santhibastawad Road, Machhe

Belagavi - 590018, Karnataka, India



A PROJECT REPORT

ON

“Automated driving car using RPI board and CNN algorithm”

Submitted in the partial fulfilment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

For the Academic Year 2018-2019

Submitted by

Nithyanand S N

Prashanth holla

Sharan C

Suveda N

1JS15IS045

1JS15IS048

1JS15IS064

1JS15IS077

Under the Guidance of

Mrs. Apsara M B

Assistant Professor, Dept. of ISE, JSSATE



2018-2019

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

JSS ACADEMY OF TECHNICAL EDUCATION

JSS Campus, Dr.Vishnuvardhan Road, Bengaluru-560060

JSS MAHAVIDYAPEETHA, MYSURU
JSS ACADEMY OF TECHNICAL EDUCATION
JSS Campus, Dr.Vishnuvardhan Road, Bengaluru-560060

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING



CERTIFICATE

This is to certify that Project work Phase II entitled “**Automated driving car using RPI board and CNN algorithm**” is a bonafide work carried out by **Prashant Holla[1JS15IS048]**, **Suveda N [1JS15IS077]**, **Sharan C [1JS15IS064]**, **Nithyanand S N [1JS15IS045]** in partial fulfilment for the award of degree of Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University Belagavi during the year 2018-2019.

Signature of the Guide

Mrs Apsara M B
Assistant Professor
Dept. of ISE
JSSATE, Bengaluru

Signature of the HOD

Dr. Dayananda P
Assoc. Prof. & Head
Dept. of ISE
JSSATE, Bengaluru

Signature of the Principal

Dr. Mrityunjaya V Latte
Principal
JSSATE, Bengaluru

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible. So with gratitude, we acknowledge all those whose guidance and encouragement crowned my effort with success.

First and foremost we would like to thank his **Holiness Jagadguru Sri Shivarathri Deshikendra Mahaswamiji** and **Dr. Mrityunjaya V Latte**, Principal, JSSATE, Bangalore for providing an opportunity to carry out the Project Work as a part of our curriculum in the partial fulfilment of the degree course.

We express our sincere gratitude for our beloved Head of the department, **Dr. Dayananda P**, for his co-operation and encouragement at all the moments of our approach.

It is our pleasant duty to place on record our deepest sense of gratitude to our respected guide **Mrs. Apsara M B**, Assistant Professor for the constant encouragement, valuable help and assistance in every possible way.

We are thankful to the Project Coordinators **Mrs. Sowmya K N**, Assistant Professor and **Mrs. Sudha P R** Assistant Professor, for their continuous co-operation and support.

We would like to thank all ISE department teachers and non-teaching staff for providing us with their valuable guidance and for being there at all stages of our work.

Prashant Holla [1JS15IS048]
Suveda N [1JS15IS077]
Sharan C [1JS15IS064]
Nithyanand S N [1JS15IS045]

ABSTRACT

Self-driving vehicles will provide more than a luxury in this era. In this paper we are implementing an upcoming technology in Machine learning and neural networks called Autonomous cars or self-driven cars. It provides relief to humans from travelling stress and helps reduce traffic congestion and accidents. This paper produces a successful model that navigates through the traffic and tackles all the obstacles placed in its path. This self drive car is implemented using Machine Learning algorithms and Convolutional neural networks in a way that provides complete automation to the driver. This car performs the following functions: self-parks, avoids traffic congestion, follows lane markings, avoids obstacles, detects vehicles and moves accordingly based on its distance from other vehicles. Previously, designs were made in such a way that all these functions were implemented in separate vehicles. But this car design integrates all these functions in a single model which increases its effectiveness, efficiency and reduces the cost of building it separately.

Table of Contents

Sl. No	Content	Page. No
1	Introduction	1
1.1	Objectives	3
1.2	Challenges	5
1.3	Environmental factors	6
1.4	Basics of CNN	7
1.5	Applications of CNN	8
2	Literature Survey	9
3	System Analysis	22
4	System Requirement Specification	23
5	Software environment	25
5.1	Open-CV	25
5.2	Open CV python	27
6	Methodology	30
7	Implementation	36
8	Conclusion	54
9	Publication	56

List of Figures

Sl. No	Figure Name	Page. No
1.1	First autonomous car	2
1.2	CNN architecture	7
1.3	Object detection	8
4.1	Raspberry Pi	23
4.2	Camera	24
6.1	Block diagram	30
6.2	CNN layers	31
6.3	Detection of road surface	32
6.4	Ultrasonic sensor	33
6.5	Working of sensor	33
6.6	Commands for movement	34
6.7	Flowchart of ultrasonic sensor working	35
6.8	Pin diagram	35
7.1	Haar cascade features	37
7.2	Face features detection	38
7.3	Haar cascade design	39
7.4	Training stages	40
7.5	Sensor code	42
7.6	Motor circuitry	43
8.1	Winning Certificate	56
8.2	Participation Certificate	57

CHAPTER 1

INTRODUCTION

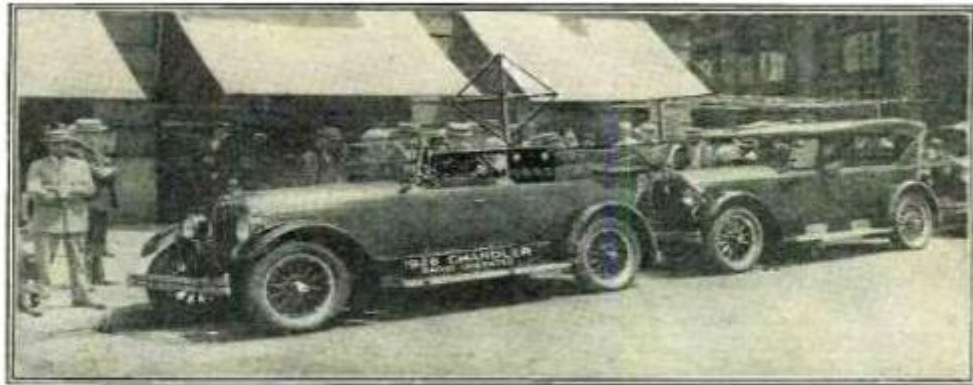
Driving has become an integral part of people's life, since they rush around trying to get their errands done. The accidents and errors encountered while driving due to various factors like use of cell phones, traffic problems and carelessness are few of the major issues to be tackled. Since the number of accidents is increasing rapidly over time, necessary measures have to be taken in order to overcome these human errors and help mankind. To overcome this problem, self-driving cars or autonomous car was introduced which drives to the destination given without any human help.

Autonomous cars or self driven cars provide humans with a luxury that lessen the commuting burden. It will eliminate accidents caused due to negligence of drivers, intoxicance and various other factors. It provides comfort to those who have to commute long distances without facing the hurdles of traffic.

"Linrrican Wonder" was the first autonomous car. It was introduced in 1925, a driver-less radio-controlled car. On the streets of New York from Broadway to Fifth Avenue, was its first journey. Later on, in 1926, this car transmitted radio impulses through antennas and sent to the circuit breakers which operated small electric motors that direct every movement of the car. In reality, with the help of Carnegie Mellon University and ALV projects in 1984 and later in 1987 by Budeswer University Munich's EUREKA Prometheus Project and Mercedes-Benz, the first actual autonomous car which needed no assistance of the drivers was developed.

Autonomous car aims at travelling from one point to the destination without any human interfering and to mainly avoid the obstacles present in front of it or near it. It helps in improving traffic conditions, provides relief to humans from daily travels, increases productivity of customers who would otherwise waste their time and energy in travelling back and forth, provides safety to those who are intoxicated and tired.

1920's



The radio-operated automobile, *Linrrican Wonder*, 1925

NHTSA

• Mobility. Safety. Customers. Economy. Responsible Steward •

3

Fig 1.1 First autonomous car “Linrrican Wonder”

To understand exactly what an autonomous car does, The National Highway Traffic Safety Administration (NHTSA) explains various levels:

➤ **1st DEGREE AUTOMATION (completely manual):** The driver has complete control over all functions of the car like brake, throttle, steering, etc.

➤ **2nd DEGREE AUTOMATION (Function specific Automation):** The driver has the control of the car but little control over few automatic functions like dynamic break during emergencies, cruise control, and lane keeping.

➤ **3rd DEGREE AUTOMATION (Combined Function Automation):** The driver is freed of two primary control functions and one or two functions are automated.

➤ **4th DEGREE AUTOMATION (Limited Self Driving Automation):** The driver has control over few small functions where the rest of the functions are automatic.

- **5th DEGREE AUTOMATION (Full Self Driving Automation):** The car is entirely automatic and has control over all the functions and can completely drive it.

At this stage, Level 3 and Level 4 automobiles are being designed. However it comes with its own challenges which need to be overcome in order to fully depend on this self driving car.

1.1 OBJECTIVES

The objective of our project is to navigate to the given destination, avoiding environmental obstacles, avoid traffic congestion, increasing safety, greater independence, more productivity, environmental gains, parking spaces, lower cost and to handle: Traffic sign, self-driving on the track, and front collision avoidance.

- **Navigate to given destination:** Self-driving cars help us to find the best possible route to our desired destination. It also finds a way to reach our destination as fast as possible. It finds routes where there is less traffic congestion to reach our destination as quick as possible.
- **Avoiding environmental obstacles:** This is one of the problems of self-driving cars, but this problem does not occur all the time. It avoids most of the obstacles that is possible by it.
- **Avoiding traffic congestion:** It helps in avoiding traffic congestion by reducing the number of collisions with other cars by maintaining a safe and consistent distance.
- **Increasing safety:** Sometimes a person maybe distracted or in some emotional distress and if he is driving a normal manual car under these circumstances then an accident is bound to occur. In this way a self-driving car helps in avoiding collision with other cars. Self-driving cars also help people who are intoxicated by driving

them back to their place. Intoxication causes a lot of accidents these days and this self-driving is pretty useful and safe.

- **Greater independence:** People who have disabilities are incapable of driving a manual car and hence self-driving cars are very much useful for these types of people. Senior citizens also benefit for self-driving car a lot as most of them have back pain, shoulder pain etc., due to old age.
- **More productivity:** As the car drives itself there will be no reason for the person to do any work. In this spare time, the person can respond to emails, attend to any business-related calls; students can their homework and anything that is productive.
- **Environmental gains:** Fewer traffic congestion results in less consumption of fuel by the car. Reduces greenhouse gases from needless idling.
- **Parking spaces:** Most of the times while using a manual car it becomes very difficult to parallel park a car. But self-driving cars park themselves. This helps people who have no idea how to park a car perfectly.
- **Stop sign:** Self-driving cars use camera to detect everything which includes traffic signs and boards. If there is a stop sign, it stops. If there is a no left turn sign, it does not take a left turn. All these data are fed into the car during manufacturing.
- **Front end collision:** Self-driving cars use sensors to sense the distance between it and the car that is in-front of it which helps to avoid front end collision.
- **Lower cost:** The safer the self-driving car, less the accidents that occur. Which helps in reducing the insurance costs of the car. The self-driving car will be able to accelerate and brake more efficiently unlike people which reduce usage of fuel and increases economy from wasted energy.
- **Other effects:** Self-driving cars reduce the cost of labor and mobility as a service, automated cars can help reduce the number of cars that are owned

individually by people. All of these can help make more space for motor bikes, cyclists and the pedestrians. The police can be aided by the cars increased awareness about illegal passenger behavior and crashing into another people's car deliberately or a pedestrian.

1.2 CHALLENGES

- Humans follow a sense of intuition and tactic when driving in order to avoid accidents. Driving is a challenging task for computers since it solely relies on the data which is fed to it to ensure safe driving.
- The challenge here is to come up with this huge data which consists of essential training for the car to self drive with accuracy considering all the obstacles and challenges to be faced when faced with traffic. These data should contain the human responses to such situations of danger so that it can be implemented in the self driven car and takes the necessary steps a human would take in its place. To ensure this, an algorithm should be designed in such a way that can take the same steps a human would take in the situation of danger.
- To overcome this, we implement machine learning algorithms and applying neural networks. Neural networks are computational representations of brains and are capable of learning in a similar fashion. Neural networks input data, performs a mathematic function and gives the output. Hence such a neural network can be implemented in the algorithms, when fed with data that consists of such situations, will perform its calculations and provides the best solution to overcome it in an efficient and accurate way. In order to train neural network, realistic data needs to be fed in order to take the necessary steps of movement.
- Convolutional Neural Network(CNN) is the neural network being implemented in this project. CNN consists of small networks where each network is assigned a particular task. A smaller CNN is used for image processing while a deeper CNN provides a steering angle for the autonomous vehicle. Another multi-layered network is used parallel with CNN to provide throttle and breaking facilities. Neural networks perform “imitation learning” which means that these networks observe the data and the various

reactions by humans and duplicate those values and train it. Hence in this way the solutions of the neural network are very near to those of human reactions in the situations of danger.

1.3 ENVIRONMENTAL FACTORS

➤ **Safety** is the biggest concern to mankind. Everyone wants to be safe. A lot of people like almost thousands of people die in motor vehicle crashes every year in most of the countries all around the world. Self-driving vehicles could most probably, reduce the number of accidents that occur-software may prove to be less error-prone than humans. But we have still not yet reached such a stage yet. In the near future, we would probably reach it.

➤ **Equity** is one more huge consideration. Self-driving cars can help the people who are unable to drive themselves, such as the elderly or disabled. But the widespread adoption of autonomous vehicles could also displace millions of employed as drivers which could result in many people being unemployed and will need to find some other way to meet their needs.

➤ **Environmental** impacts are also one major concern and it is very uncertain. If those vehicles are powered by gasoline, then the smoke emitted by the car could cause a lot of damage to the environment and climate emissions could skyrocket. If, however, the vehicles are electrified and paired with a clean electricity grid then there would be less air pollution and damage to the environment can reduce, perhaps significantly.

➤ **Taxi industries** might start using autonomous cars for driving people to their destination, but this will also result in increase in unemployment.

➤ **Other industries such as healthcare, restaurant, military, fire brigade etc.** Might make use self-driving technology for their benefits. Hospitals can implement this technology to their ambulances which will help people who are on the verge of death to the hospital as fast as possible. In the military, lots of unwanted deaths can be reduced. It also leads to reduction in deploying personnel. If a house is on fire in a faraway place, the autonomous car technology can be implemented in the fire trucks, so they reach the place on time and put out the fire.

So, basically self-driving cars help people in many ways which are yet not thought of. There might be few incidents where the car might go out of control and break traffic laws, injure pedestrians, collide with car that's there in the front etc.

1.4 BASICS OF CONVOLUTIONAL NEURAL NETWORK

CNN (Convolutional Neural Network) is a class of deep neural learning used commonly to analyze images. CNNs are multilayer perceptrons or fully connected networks where it can be seen that each neuron in one layer is connected to the subsequent neurons in the next layer.

CNN consisted of input , output and a number of layers which are hidden. The layers of CNN which are hidden consist of [1]convolutional layers, [2]pooling layers, [3] fully connected layers , [4]normalization layers and [5] activation function.

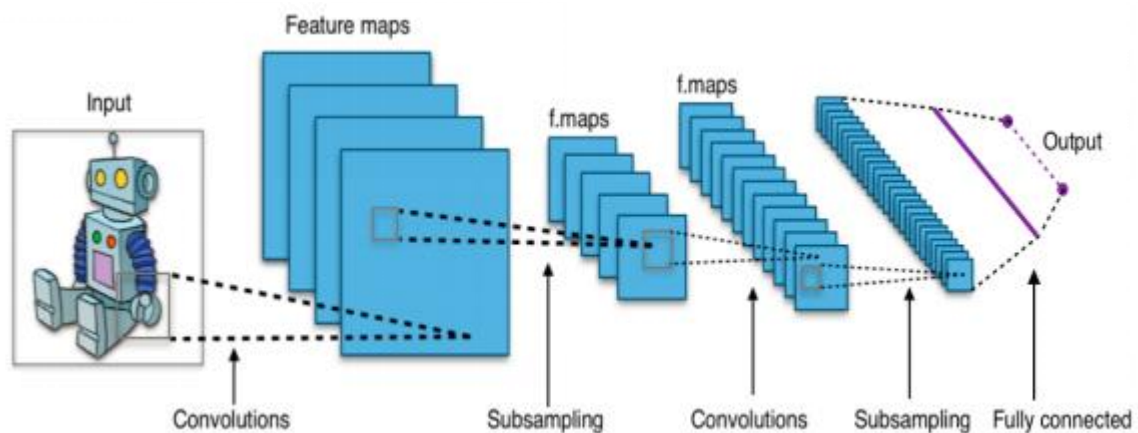


Fig 1.2 CNN architecture

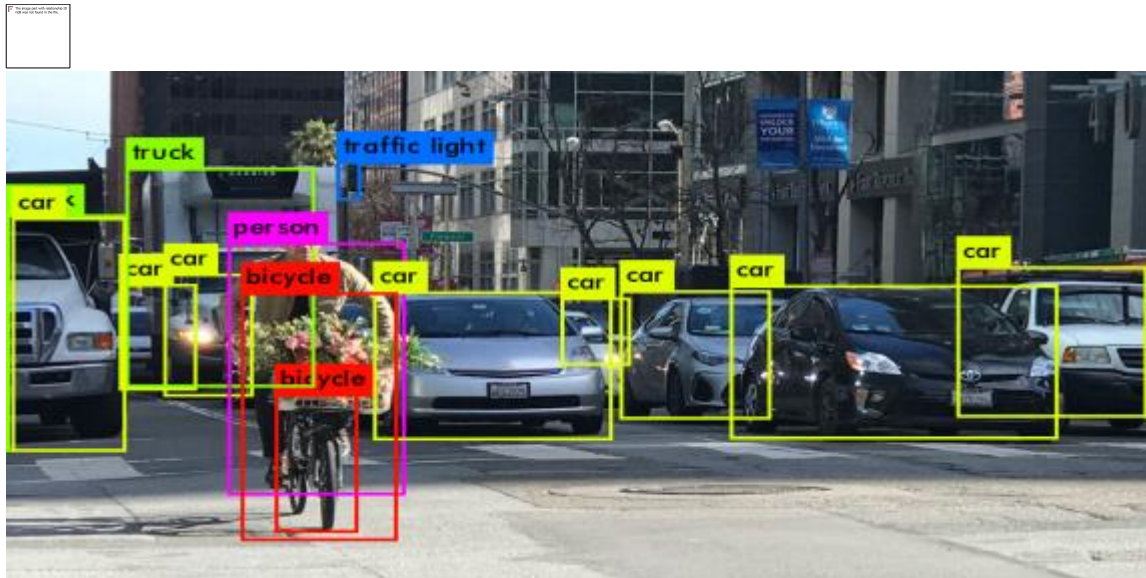


Fig 1.3 Object detection

1.5 APPLICATIONS OF CNN

- Image Classification and recognition
- Video analysis
- Object Detection
- Handwritten digit recognition
- Natural language processing
- Optical Character Recognition
- Emotion Detection
- Health risk assessment
- Checkers game and Go
- Feature extraction

CHAPTER 2

LITERATURE SURVEY

I. “Autonomous Driving with a Simulation Trained Convolutional Neural Network”

It was published in 2017 by Cameron in University of the Pacific Theses and Dissertations as a journal paper.

It used a reflexive neural network to run the autonomous car which converts the images into commands to drive it. The simulation data used to train the neural network was taken from Grand Theft Auto. The data set containing the necessary instructions was verified on connecting the neural network to the car system containing a camera input and the specific steering value, where the resulting average error rate was only 1.9%. This neural network was connected to a RC car system which provided a camera input and receives steering angle and throttling value.

The designed car could successfully navigate 98 of the 100 laps in response to various obstacles placed in front of it.

METHODOLOGY

Convolutional neural network was designed to implement this design. It consisted of a CNN network for image processing, a multi layered network for steering, a deep layered network for steering and additional networks for braking. It used python and Keras API to implement the neural network. The camera was used to provide images to the system and the existing steering angle as the input which was used to find the throttle. It consists of four modules:

- **Steering processing network:** CNN provides a specific steering angle based on the input provided by the camera. It consists of series of layers that help in pattern detection.

- Short time memory steering network: It is a small network which requires a small amount of processing power. The steering branch is responsible for handling at least 12 of the previous steering angles.
- Merging network: The merged network provides the combined output of the previously described networks. It has a deep layered network of six layers
- Anti-collision network- This network is independent of the other networks and its main purpose is to make sure that the brakes are applied when an object comes in front of it.

PROS

This system successfully drove 98% of the laps on track without navigation issues as well as achieving a 90% success rate in avoiding collisions using only a single camera as input.

CONS

This proposed system uses a single fixed camera so it cannot rotate and give more accurate information about the road view. This can be solved by using more cameras and variety of sensors to enhance the efficiency. Another disadvantage is that it needs to be trained with huge data sets in order to take a correct decision.

II. “Vision based Automated Parking System”

IEEE Conference paper published in June 2010.

This paper provided a solution to overcome monitoring of parking spaces using vision based automated parking. Since the number of cars is increasing rapidly, it is difficult to procure parking spaces in an efficient manner and to avoid traffic. The current parking system just provided the number of parking spaces available and not the actual feasible parking space. Hence this automated system was designed that could self-park based on the available spaces.

Coordinates systems was used to detect available car parking spaces using car classifier and regions of interest . The technology used in this paper had an accuracy ranging from 90% to 100% for a parking system of four spaces. This paper showed that implementation of a vision based car park management system would detect and indicate the car park spaces based on proximity and availability.

METHODOLOGY

Sensors along with a camera were installed to provide available parking spaces. Based on the input video, Haar-like features determined if a car was present within the parking premises. The classification of the object was done using a tool called OpenCV. Two types of input images needed to be fed to train this algorithm. One set called the positive images that contained the object to be detected and another set called the negative images had no object to be detected. Once this data was analyzed, coordinate system was used to detect the coordinate of the car parking spaces with the help of a marker tool of OpenCV. This method was used to determine the total no of parking spaces that were available and the best parking place for the car in the list of available spaces.

PROS

It accurately determines the presence of a car in a parking space and the total number of parking spaces available and that the utilization of a camera system to monitor parking of car is feasible. It helps to park the car in tight spaces with ease.

CONS

It used a simple camera to monitor spaces so decreases the accuracy of detection of spaces due to limited coverage. And the coordinate method used in this paper in selecting the specific parking locations has limited the usage by making parking bay location fixed and thus limits the camera to be at a fixed location.

III. “Multi-Sensor Based Collision Avoidance Algorithm for mobile robot”

It was an IEEE Conference Paper published in 2015.

Here collision avoidance systems are used to minimize the risk of collisions while controlling robots. In this paper, robot systems used collision and sensing detection techniques to avoid collision and to make sure of safe travel. The collision position was detected by inputs from various devices like GPS, cameras and sensors like Infrared sensors and ultrasonic sensors. The robot can detect and prevent collisions when it is traveling, and thus the movement of the system wasn't affected until the algorithm decided the collision position.

This collision avoidance algorithm was used effectively in various areas like air traffic control, underwater travel, and most importantly in autonomous cars. This obstacle detection technique ensured that two or more objects do not collide with each other, either two moving systems or one moving system with an obstacle. Obstacles to be detected by this algorithm were of two types: static obstacle where it remains in one position and dynamic obstacle whose position is not certain and may move.

METHODOLOGY

It consists of three algorithms which perform separate tasks and a fourth algorithm which combines all these algorithms to provide this system.

- Edge detection and distance measurement: To measure the distances and to detect the edges two infrared sensors were used. If the value detected by these two sensors was less than the threshold value, then the object was detected.
- Camera based object detection algorithm: Images were captured with the help of a camera from which the object was processed and identified. Then thresholding is done to eliminate noises present along with the image. After noise isolation, the object's dimension was calculated by using coordinate geometry and 2D system.

- GPS-distance measurement: It uses GPS to locate the robot. The found value is then converted from degree to radians to centimeters using a formula. Hence with this the exact location is found out and the exact distance between the robot system and the obstacle could be detected.

- Collision avoidance algorithm: This algorithm integrates all the three algorithms output as the input. It takes input from camera, GPS and the edge detection algorithm and uses it in the collision detection algorithm and a decision is taken based on the location of obstacle. If the robot detects an obstacle, it waits for some time, backs up, again waits for some time and then it turns if it finds that there is no obstacle remaining.

PROS

This algorithm was tested with different obstacles and was successful. The collision avoidance algorithm provided the distance showed the improved performance in detecting objects and avoiding collision.

CONS

It is used to efficiently detect objects and collisions but it includes only this one application. Hence it becomes costly to create such an application to implement only collision avoidance whereas in an automated system, collision avoidance is used along with smart parking, image processing, etc.

IV. “MIT Autonomous Vehicles Technology Study: Large-Scale Deep Learnings Based Analysis of Drivers Behaviour and Interaction with Automation”

It was published in September 2018 in MIT(Massachusetts Institute of Technology) .

The objectives of this proposed autonomous vehicle were to:

- Collect large scale driving data collection with the help of deep learning techniques: external and internal systems.
- Understanding human-vehicle technology interaction with driving data, driver characteristics, and experiences of humans with respect to technology.
- Find out how various factors and technologies could be implemented to save lives.

This system was designed using advanced technologies where the data streams included IMU, GPS, CAN messages and HD video streams of the driver’s face, driver’s cabin, the forward path and the instrument cluster. It implements “naturalistic driving study”, which is not restricted by experimental design and which collects audio, video and other elements of driving over a large period of time and to use this collected data to help the autonomous car.

The aim of this paper is to understand the data being collected throughout from vehicles in order to design, implement and run the autonomous system, inform the providers of insurance of safety measures, and enlighten the educational institutions on how to use these autonomous cars in order to reduce the burden of mankind.

It provides autonomy at all levels. It aims at understanding human behaviour in a semi- automated system for a period of 1 year and using those reactions to drive the autonomous car. It uses AI to inspect the driving experience through the collection of data and to use human expertise to understand how to react in difficult situations.

**V. “Review on Autonomous Car using Raspberry Pi”
Published in IJRASET in 2018.**

In this paper, it is stated that the project mainly focuses on the input which is the continuous set of images/videos that are taken by the webcam connected to the Raspberry Pi. The object is then detected and tracked by moving the camera in the path of the detected object. Raspberry pi is used since it has better size but it has less speed. Even though the FPS rate of both systems are different, accuracy of both systems are similar. Background subtraction method is used in this system with the help of a fixed camera by generating the foreground mask. The frame obtained is compared with the normal one with background images which has the static part of a video. The raspberry pi camera is used through which background subtraction can be done .

METHODOLOGY

In this system, the camera captures the image which is then sent to the raspberry pi, to which the camera was connected using an USB port. The input image was executed on the Raspbian OS software followed by the execution of the python code. The signals generated after the code is executed, is sent to the car. The car then detects objects and the path to the destination effectively and starts driving towards it. Its components include Raspberry pi, Pi camera, Motor driver, ultrasonic sensor module, raspbian OS, python, OpenCV.

PROS

Raspberry pi and camera is used to detect objects within the given range to ensure safe movement.

CONS

It uses Raspberry pi connected with a single webcam hence the input of the object is limited. It uses a fixed camera which cannot pivot to produce other available options. It does not use sensors to locate the objects more accurately.

VI. **“IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE)”**

In this paper, we study the design and different methodology of automated guided vehicle systems (AGVS). This paper provides an overview on AGVS technology discusses recent technological developments and describes the formulation to control the traffic inside industrial work space.

An automated guided vehicle is a programmable mobile vehicle. The automated guided vehicle is used in industrial application to move material around a manufacturing facility. The AGV are capable of transportation task fully automated at low expenses. AGV have to make the system automatic by doing the decision on the path selection. This is done through different method frequency selected mode, path selected mode and vision based mode etc.

The main parts of automated guided system are

Vehicle:

Vehicle is the central elements of AGV as they perform the actual transportation task. The vehicle individually according to the specific condition have to be designed and of the environment.

Guided path system:

The vehicle guidance system is the method by which AGV are defined and vehicles are controlled to follow the path ways. Most commonly used guidance technologies in AGV are 1. Landmarked based navigation 2. Behaviours based navigation 3. Vision based navigation.

Floor control and traffic management system:

To operate efficiently and increasing the productivity of AGV, the vehicle should be well managed. Delivery task must be allocated to minimizing the waiting time at load/unload station. There are several possible directions for further research. We can improve the guided tape type AGV utilizing better navigation technique. It can be adopted any environment and cheap among autonomous robot. There is significant

amount of difference between theoretical and practical work cycle value of time which can be optimized by adopting different methodology.

VII. “The International Journal of Engineering and Science (IJES) - Adaptive Cruise Control”.

The idea of assisting driver in longitudinal control of the vehicle to avoid collisions has been a focal point of research at many automobile companies and research organizations. When switched on, this device takes up the duty of accelerating or braking to maintain an even speed. While cars surely provide freedom and ease, cars can also be quite vulnerable. Even if you are a secured driver there are many other drivers on the road that may not be. Lot of things can go wrong with cars. You could be hit, or hit something or someone. Any of the thousands of parts on your vehicle could break or not function appropriately.

Adaptive cruise control is close to traditional cruise control in that it maintains the vehicle's speed. Although, unlike traditional cruise control, this new system can automatically adjust speed in order to retain a proper distance between vehicles in the same lane. This is accomplished through a radar headway sensor, digital signal processor and longitudinal controller. If the lead vehicle slows down, or if another object is detected, the system sends a signal to the engine or braking system to decelerate. Then, when the road is clear, the system will re-accelerate the vehicle back to the set speed.

In this paper “The International Journal of Engineering and Science (IJES) - Adaptive Cruise Control”, published in March 2013 states that there are many safety systems that exist, but provide safety after meeting with an accident. Some are: Air Bags, Anti-Lock Braking etc. The existing technologies include:

1. Enhanced Accident Response System (EARS)
2. Anti-lock Braking System (ABS)
3. Pressure Sensing system (Airbags)

Enhanced Accident Response System (EARS): There are several ways of vehicles now and in the future that handle an emergency situation. DaimlerChrysler's

Enhanced Accident Response System (EARS) unlocks doors, turns on interior lighting and shuts off fuel when airbags deploy.

Anti-lock Braking System (ABS): An anti-lock brake system (ABS) blocks a vehicle's wheels from locking during braking, which allows the driver to maintain better steering control. Learning to use the ABS correctly will deliver you with the considerable benefit from the system. ABS. Four-wheel ABS monitor and controls all the wheels of the vehicle, while two-wheel ABS only monitor and controls the rear wheels of a vehicle.

Pressure Sensing system (Airbags): Depending on the speed at collision and the stiffness of the object struck, front air bags get aerated to prevent occupants from slamming the dashboard, steering wheel, and windshield. Side air bags reduce the risk. Even though air bags provide life-saving benefits for the most of people, there are situations in which air bag deployment can have unfavorable effects, such as when occupants are unbelted.

Pros

It introduced a technology that could be used for safe cruising. It ensures the safety of the occupants using by automatic breaks and airbags.

Cons

It is a rather old technology since many new innovations have emerged with enhanced effects. The main drawback is that it focuses on providing safety after the accident has occurred.

VIII. “International Journal of Computer Applications (0975 – 8887) Volume 113 No. 9, March 2015”

As number of accidents are increasing each day, it is very important to avoid errors made by humans and help the needy. With help of self-driving cars, all of this may come to an end. The passengers need to enter the destination and then they can continue their work.

The system uses Raspberry Pi as a processing chip to build an autonomous car design . Required data from the real world to the car is provided with the help of a HD camera along with an ultrasonic sensor . The car reaches the entered destination safely and intelligently by avoiding human errors. Obstacle detection and lane detection algorithms are used together to acquire control of the car.

Lane detection algorithm is used with two approaches - model based technique and feature based technique. Feature based technique needs road having well-painted lines or strong lane edges, or else it fails while the model-based technique is robust against noise and missing data. In the proposed algorithm to detect the lanes, a combination of feature and model base is used.

The project has three phases:

- Remotely Controlled Car: Controlling the movement of the car can be done using some kind of remote control interface. It may be a web interface or a mobile interface. Since setting up python and SSH in raspberry pi is very easy without any cons, it is recommended to use a mobile based user interface for controlling the car motion.
- Car with Autonomous Obstacle Avoidance: Using ultrasonic sensor is better option in this case as it doesn't require high CPU computations and detects the obstacles as well as help us finding the distance. The distance of the nearby vehicle or any other obstacles is detected depending on the time taken to receive the reflected signal.
- Car with Autonomous Navigation: This phase is about making the car autonomous i.e. the car determines the road by itself and finds its line of motion. Following techniques are performed step-by-step to achieve autonomous behaviour in the car.

Pros

Collision with obstacles and any other car is avoided with the help of ultrasonic sensors.

Cons

Does not use Machine learning. It is accurate but efficiency needs to be improved by learning on its own.

IX. “International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE) Vol 5, Issue 4”

This paper published in April 2018 states that there are two sub-systems that are used to build this project. That is Image processing sub-system and obstacle detection subsystem. Camera is attached to image processing sub-system which captures the image. To detect the road lane mainly image processing is used here. To detect the obstacle in front of car and also calculate the distance between the obstacle and the car, obstacle detection sub-system is used. These are captured with help of a webcam interfaced to the Raspberry Pi. Raspberry Pi has less software glitches and provides overall performance and hence it is convenient to use. System takes out the data from the image and generates the command about turn. Generated commands are forward to obstacle detection subsystem. If adequate distance is available to move car forward the command from Raspberry pi is forwarded to motor driver else this command are rejected.

It is significant to detect the lane from the image of road and check their position in the form of pixel coordinates for the settlement of turn. To perform the lane detection action it is significant to convert image into gray scale. The white pixels from the region of interest are considered only. Position of lane on the road is given by the count of this image. The region of interest is slide vertically with respect to speed of car. Ultrasonic module provides the distance between two objects. Arduino's serial communication gives us stop command to the vehicle and also results about the distance.

In this project, the main aim is at the development of self-driving vehicles for convenient transportation without a driver. Cars that drive themselves will improve road safety, accessibility, fuel efficiency and increase productivity.

Pros: Raspberry pi and camera is used to detect objects within the given range to ensure safe movement.

Cons: It uses Raspberry pi connected with a single webcam hence the input of the object is limited. It uses a fixed camera which cannot pivot to produce other available options. It does not use sensors to locate the objects more accurately.

X. AUTONOMOUS VEHICLE - LITERATURE REVIEW 305AEE, 2015

To design a self-driving vehicle, the system has to be designed and planned in such a way that avoids obstacle during the motion with the help of information obtained from the sensors, installed on the vehicle. It moves to the predefined point of the destination.

METHODOLOGY

The software installed and implemented is the major component to run the vehicle from different aspect. The methodology is split into these 6 tasks :

1. Processing of sensor data provides the information on the position, obstacle, motion of the vehicle and environment to the system.
2. Localization which establishes the communication between the map of that position cars present and location of the vehicle.
3. Obstacle tracking allows the moving vehicle to track the moving and static objects.
4. Path planning method helps to determine one route that maximizes a plurality of criteria among multiple trails.
5. A behavioural module that gradually relax constraints in the driving process to succeed in situations of unpredictable environments.
6. Control the vehicle itself, its throttle, brake, gear shifter and steering wheel.

It can be said that the vehicle which is going to be built is very helpful for the people widely. Especially it will be a great vehicle for the people who has some disabilities, such as blind, deaf, weak to move, etc.

Pros: Uses sensors to detect.

Cons: Camera is not used only sensors are used.

CHAPTER 3

SYSTEM ANALYSIS

3.1 EXISTING METHOD

Automotive industry has been a great help to the mankind but along with being a boon, no other invention or technology has caused as much harm to the mankind causing over 1 million deaths every year. People die due to traffic accidents, due to intoxicance, due to error in judgement and tactics since the driver is completely in control of the car. With this traditional method of travelling, people spend an average of two hours a day commuting to their destinations. Due to this, time spent is high, efforts and energy are wasted and drained, inability to spend time with friends and family and increased stress.

The invention of autonomous cars provided an enhanced solution to all these problems where the customer can travel to their destination without putting any effort, time and stress.

3.2 PROPOSED METHOD

The proposed method of autonomous car uses these three techniques: convolutional neural network with lane detection, Haar cascade for stop/start signals and ultrasonic sensors for obstacle detection. The image of the road is taken from the RPI camera to the board via USB port. The camera sends high quality video to the board which acts as input to the machine learning algorithms which decides the movements of the RC car.

CNN are applied in many image recognition problems in order to classify the object within the image. Once the CNN has been trained with the training data set the CNN could interpret the traffic signals. To detect the road or lane we are going to use the lane detection algorithm from OpenCV library. Here we take the real time video frames from the RPI camera as the input for the algorithm.

Finally we use an ultrasonic sensor module to compute the actual distance between our self driving car and obstacle.

CHAPTER 4

SYSTEM REQUIREMENT SPECIFICATION

SOFTWARE AND HARDWARE REQUIREMENTS

- **RASPBERRY PIE:** It is a single board computer of five types-the model B+, model A+, model B, model A and the compute module. The operating system is stored in Secure Digital (SD) and programming memory in SDHC. It has 1 to 4 USB slots, HDMI and composite video output.

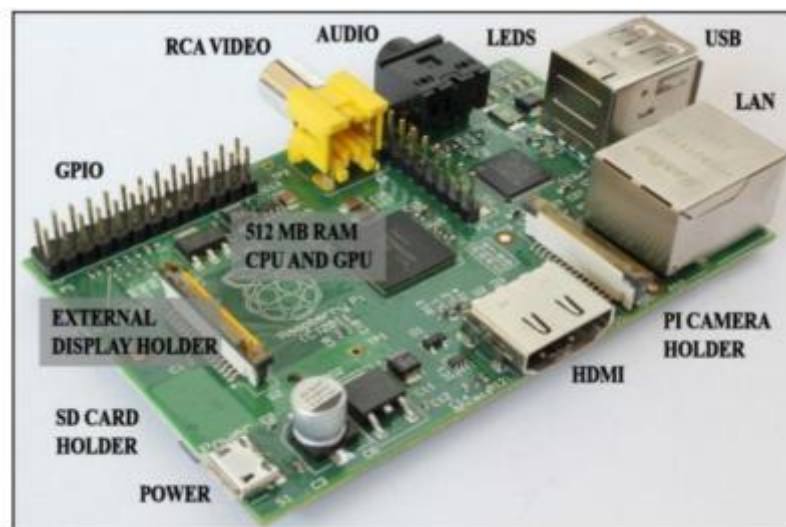


Fig 4.1 Raspberry Pi

- **CAMERA:** The camera captures images as well as HD videos. It connects to the USB port of raspberry pi using a cable.



Fig 4.2 Camera

- **MOTOR DRIVER:** It controls the dc motors for movement and can be interfaced with Arduino or Raspberry pi.
- **ULTRA SONIC SENSOR MODULE:** It is uses sonar to calculate non-distance measurement for accuracy. It consists of ultrasonic transmitter, receiver and control circuit. Then the time difference between transmission and detection is calculated.
- **RASPBIAN OS:** It is a free OS, user friendly and best optimized for the Raspberry Pi hardware. It is based on LINUX and freely available for download.
- **PYTHON:** It is a high level programming language used widely to allow the developers to write lesser lines of codes in comparison to other languages.
- **OPEN CV**
- **An L293 or SN755410 motor driver**
- **A Breadboard to connect everything on**
- **1 or 2 DC motors of 6volts**
- **4x AA battery:** to provide power
- **Jumper cables to connect everything up**
- **IC motor driver**

CHAPTER 5

SOFTWARE ENVIRONMENT

5.1 OPEN-CV

Open Source Computer Vision Library contains programming functions which is an open source vision . It contains a library of machine learning software. Open CV provided a basic infrastructure for computer applications which makes it easy for business to utilize these library functions and modify the codes already present.

Intel originally developed Open CV. These libraries are cross-platform, freely available for the users to use and it is a BSD licensed product.

Deep neural networks like Tensor flow, Caffe and Torch/Py Torch are supported by Open CV. It consists of libraries which has a collection of more than 2500 algorithms that are both classic and related to machine learning. Detection and recognition of faces, object identification, camera moments tracking and obtaining, track moving and static objects , producing high resolution images are some of the uses of these algorithms present in the Open CV libraries. These libraries are now being used extensively by private companies, government bodies and various research organizations.

Open CV was initially written and interfaced in C++. It has interfaces like Python, C++, Java and Matlab/Octave. Many wrappers in other languages like C#, Ruby, Perl and Haskell have been developed so that it appeals to a wider audience to be adopted.

Windows, Linux, Android, macOS, OpenBSD are few of the operating systems that support the running of OpenCV.

5.1.1 WHAT OPENCV CAN DO?

- It is mainly used in image classification to read and write images.
- For detection of features of faces
- For object identification and detection of shapes like circle, rectangle in a certain image.
- Recognition of texts in images like for example reading number plates during vehicle classification.
- It can modify colors and quality of the images
- Development of augmented reality apps
- Obstacle identification for obstacle avoidance algorithms.

5.1.2 APPLICATIONS OF OPENCV

- Segmentation and recognition
- Facial recognition systems
- Motion understanding
- Depth perception from camera input
- Augmented reality
- Gesture recognition
- Human and computer interaction
- Mobile robotics

For supporting these functions, Open CV includes few ML algorithms like:

- Decision tree learning
- Naive Bayes classifier

- K-means algorithm
- Deep neural network
- Artificial neural network

5.1.3 ADVANTAGES OF OPENCV

- OpenCV is available to any user free of cost to use.
- It is fast since it is programmed in C/C++
- Low RAM usage
- It is portable which means it can run on any device that supports C or C++

5.1.4 DISADVANTAGES OF OPENCV

- OpenCV is not as easy to use as Matlab can be easily used.
- OpenCV has a library of its own causing issues of conflict when using PCL library with open-cv libraries..

5.2 OPENCV-PYTHON

Open-CV Python is a library of Python programs for easier computing of functions..

Python is a general purpose programming language founded by Guido van Rossum . Because of its simplicity and code readability it became quite popular. Using python the programmer could write fewer lines of coding to express his/her ideas easily.

Python is slower compared to languages like C/C++. Python can be easily extended with C/C++, hence intensive code in C/C++ can be written and Python wrappers can be created used as Python modules. It has two advantages: the code is as fast as the

original C/C++ code and it is easier to code in Python than C/C++. OpenCV-Python is a Python wrapper for the original OpenCV C++ implementation.

OpenCV-Python uses Numpy. It is an optimized library for numerical operations. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

5.2.1 IMAGES

READ IMAGE

An image is read with the help of the function **cv2.imread()**. To read the image, the image must be stored in the working directory or the full path of the image must be mentioned.

- **cv2.IMREAD_COLOR** : Loads a color image. Any transparency of image will be neglected. It is the default flag.
- **cv2.IMREAD_GRAYSCALE** : Loads image in grayscale mode
- **cv2.IMREAD_UNCHANGED** : Loads image as such including alpha channel

For example:

```
import numpy as np
import cv2
img = cv2.imread('image1.jpg',0)
```

DISPLAY IMAGE

The image that is displayed resizes itself according to the size of the window. The function used to display the image on a window is **cv2.imshow()**.

For example, the code might look like:

```
cv2.imshow('image2',img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

WRITING AN IMAGE

To save an image this function is used- **cv2.imwrite()**.

```
cv2.imwrite('image3.png',img)
```

5.2.2 VIDEOS

Open CV provides an interface to interpret live streams obtained from the cameras. It captures the video from the camera, converts it into gray scale form and uses it in any of the function. These are captured frame-by-frame from the cameras.

A sample code to read a video is:

```
cap = cv2.VideoCapture(0)
```

```
while(True):
```

```
    # Capture frame-by-frame
```

```
    ret, frame = cap.read()
```

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
    # Display the resulting frame
```

```
    cv2.imshow('frame',gray)
```

```
    if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
        break
```

```
# When everything done, release the capture
```

```
cap.release()
```

```
cv2.destroyAllWindows()
```

CHAPTER 6

SYSTEM DESIGN/METHODOLOGY

The proposed project design mainly focuses on developing autonomous self-driven RC car. We do this by developing an RC car with a RPI 3B board, PI camera and other multiple sensors like ultrasonic sensors. A basic RC car is assembled with DC motors which receives the inputs and voltage from RPI board's GPIO pins and VCC pin.

The image of the road is taken from the RPI camera and sent to the board via Camera port. The camera sends medium quality video to the board which acts as input to the machine learning algorithms which decides the movements of the RC car.

A basic RC car consist of two DC motors, wheels, RPI 3B board motorcontroller. All the receive inputs from RPI's GPIO pins. There are three main divisions to bring about this: Harr cascade, convolutional neural network to detect the lane and ultrasonic sensors.

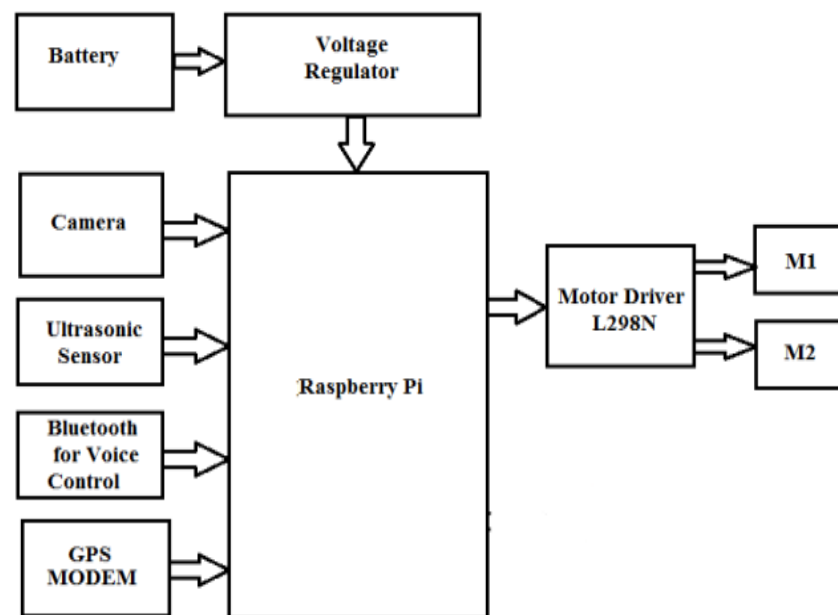


Fig 6.1 Block Diagram

We take inputs from the RPI camera which sends real-time video to the RPI board. These videos are splitted into frames and fed into the algorithm.

CNN

The CNN has been trained with the training dataset, it can interpret the Lanes. The CNN consists of three main layers.

- The first layer is the convolutional layer where the input is given. Here the image input is always in the matrix form that is, a filter is applied to an image and the pixels are recorded into matrix form.
- In the hidden layers, the image undergoes through many filters and becomes a linear pixel array. The output is according to the linear pixel array.

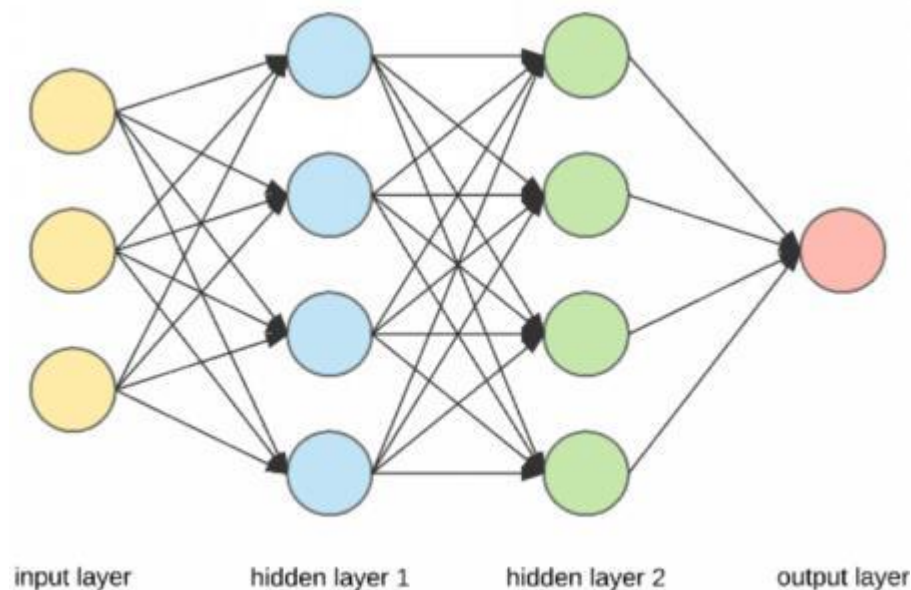


Fig 6.2 CNN Layers

We then detect the road or lane from Open CV library. Webcam also use Hough line to detect lanes .Here we take the real time video frames from the RPI camera as the input for the algorithm.



Fig11: Original road with region of interest (ROI)



Fig12: The detected road surface



Fig13: The contour around the road



Fig 14: Detected road edges in different parts of ROI

Fig 6.3 Detection of road surface

OBSTACLE DETECTION

Ultrasonic sensor module computes the distance between the obstacle and our RC autonomous car. The ultrasonic sensor which is placed in front of the RC car emits a low frequency ultrasonic waves which hit the obstacle in front of us when RPI sends command to the trigger to the ultrasonic sensor and rebounds back to the sensor. The sensor captures the rebounded ultrasonic waves and calculates the distance between the obstacles using the formula:

$$\text{Distance} = (\text{speed} \times \text{time}) / 2$$



Fig 6.4 Ultrasonic sensor

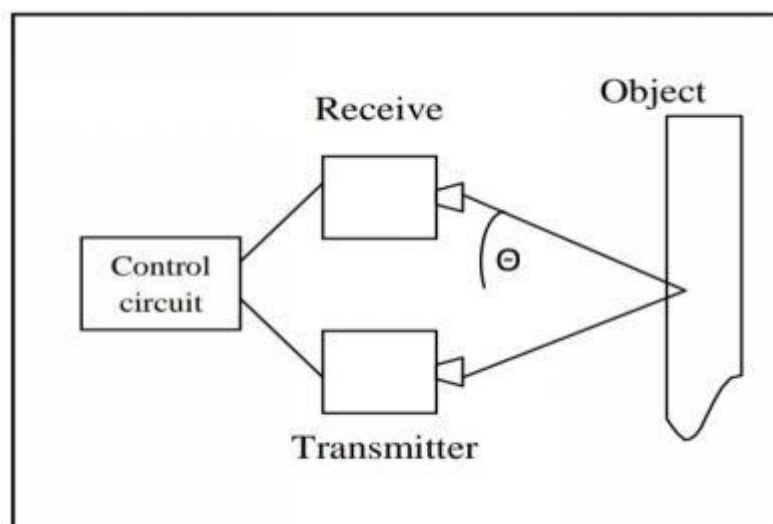


Fig 6.5 Working of sensor

SIGNAL DETECTION

We use haar cascade from openCv to detect signals. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. We get the classifiers from openCv which are already trained to detect the stop signs and signals. Each frame goes through this algorithm and gets converted into pixel array. These arrays are used with the classifiers to predict the signals. We also get the distance between the signals and the car. We calculate it using the focal length of the camera and the X,Y co-ordinate of the detected signal in the frame.

Using output of lane detection algorithm which gives us the steering angle, Haar cascade algorithm which gives us the stop signs and ultrasonic sensors which gives us the distance between the obstacles, we decide the next movement of the RC car. A threshold distance is set between the obstacles and the car as 20cm.

- If the distance is below 20cms, the car stops no matter what that is the RPI stops the Voltage to the DC motors and the car comes to halt until the obstacle is removed .
- If the distance is more than 20cms and the haar cascade recognizes a stop sign, the RC car comes to halt.
- If both the outputs pass the threshold value, then the lane detection algorithm gives the RC car the steering angle that is RPI will send commands to motor chip either to go forward or take a turn.
- To make a left turn if necessary, the output voltage for the left server motor is reduced.
- To make a right turn if necessary, the output voltage for the right server motor is reduced.
- To halt, we stop the output voltage for both server motor. The algorithm used to detect stop sign is HAAR cascade. It is a classifier which detects the object that it has trained for. It is done by superimposing positive image over a set of negative images.

COMMAND	GPIO pin 1	GPIO pin 2
STOP	LOW	LOW
FORWARD	HIGH	HIGH
LEFT	HIGH	LOW
RIGHT	LOW	HIGH

Fig 6.6 Commands for movement

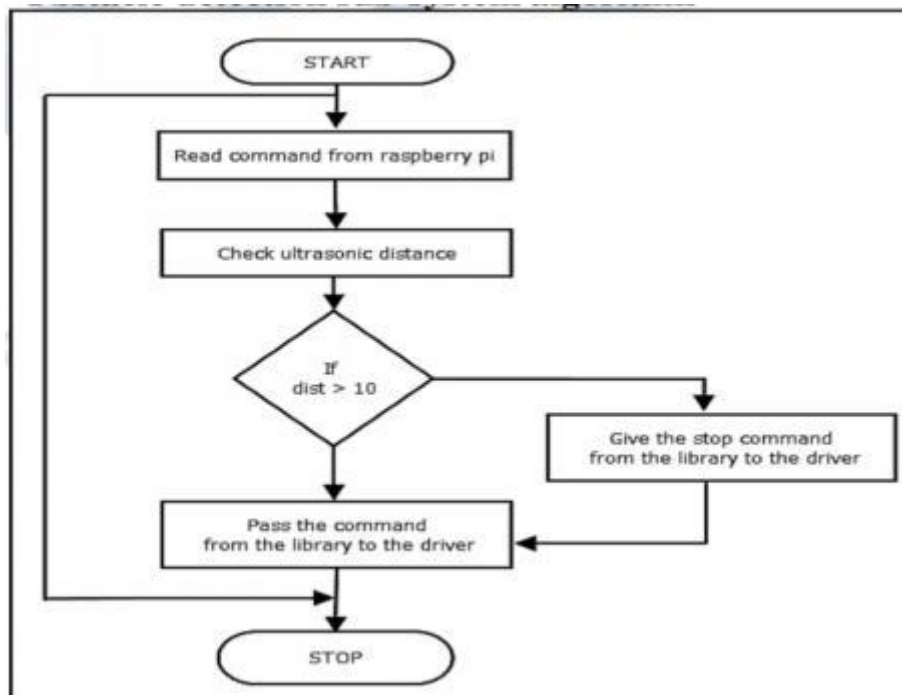


Fig 6.7 Flowchart of ultrasonic sensor and its working

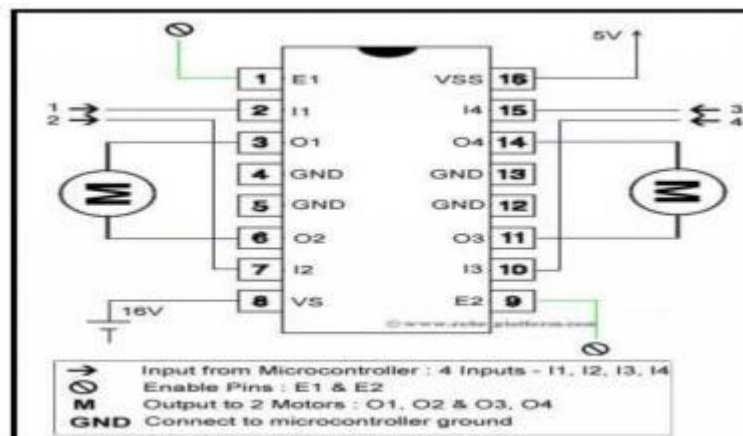


Fig 6.8 Pin diagram

CHAPTER 7

IMPLEMENTATION

The main aim behind the development of an autonomous car is to reduce the cost of separate functions by combining all of them into one system.

The following algorithm depicts the exact working of our autonomous model:

1. Live feed is taken from the RPI camera.
2. This input is fed to the RPI board connected to the breadboard.
3. The input video is then converted into a number of frames and each frame is fed into the algorithm.
4. Each picture then goes through the Haar cascade algorithm and checks for stop sign.
5. If yes, the car stops.

Else, the algorithm checks the distance between the obstacle and the car using the ultrasonic sensor.

6. If detected, then the car stops.

Else the pictures are fed into the CNN algorithm which tells the car to follow the lane rules.

➤ Haar Cascade

It is a machine learning algorithm which is used to detect and recognise objects in images and in videos. It's a machine learning approach where cascade function which trained over a period of time from positive and negative images. It is primarily used to detect objects in other blurred or normal images.

The algorithm has the following stages:

1. Haar Feature Selection
2. Creating Integral Images
3. Training
4. Cascading Classifiers

Object Detection using Haar feature-based cascade classifiers is a constructive object detection method put forward by Paul Viola and Michael Jones in their paper, “Rapid Object Detection using a Boosted Cascade of Simple Features” in 2001. It is an ML approach where a cascade function is trained with the help of positive and negative images. It is then utilised to detect objects in other images.

Originally, the algorithm needs many negative images (images without faces) and positive images (images of faces) to train the classifier. Then features are extracted from it. To do this, Haar features shown in the image below are utilised. They act like convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

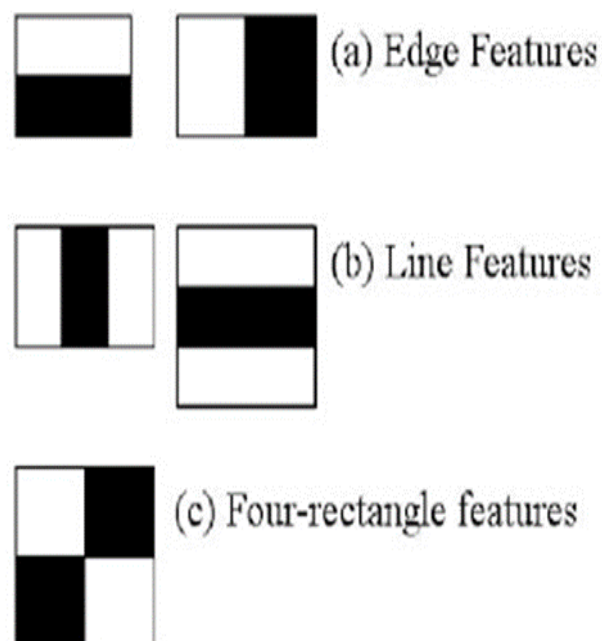


Fig 7.1 Haar cascade features

Most of features calculated, are not relevant. For example, take the following below image. The top most row displays just two good features. The first feature that is selected focuses on the region of the nose and cheeks is usually less dark than the region of the eyes. The second feature selected usually relies on the property that the bridge of the nose is less dark than the eyes. But the same windows applying on cheeks or any other place is immaterial.

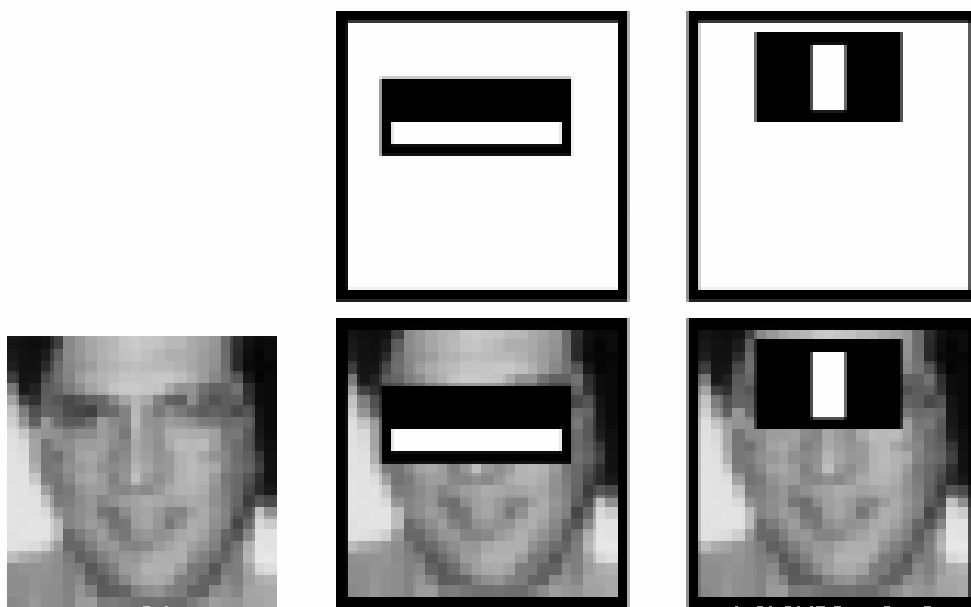


Fig 7.2 Face features Detection

In detection phase, over the input image a window of the target size is moved, Haar features are calculated and for each division of the image. This difference is then compared to a learned threshold that segregates non-objects from objects. Because each Haar feature is only a "weak classifier" (its detection quality is slightly better than random guessing) a great number of Haar features are required to describe an object with a lot of accuracy and are structured into cascade classifiers to form a strong classifier.

For training we need a set of negative and positive images. Positive images are the ones that is supposed to be detected and negative images are the ones that should not

be detected. The negative image samples must be prepared manually. The positive image samples are created using the `opencv_createsamples` application.

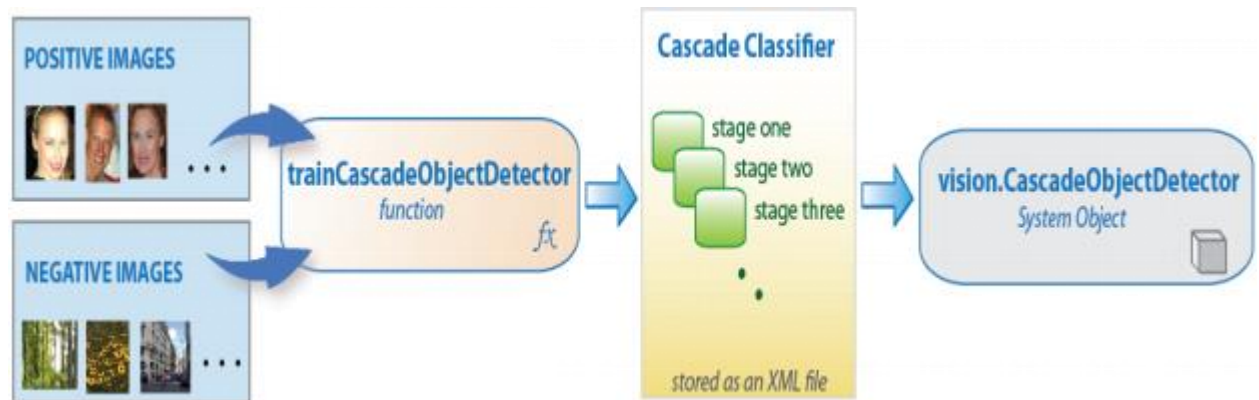


Fig 7.3 Haar cascade design

The cascade classifier has a lot of stages, where each stage has a lot of weak learners. The weak learners are referred as decision stumps which are simple classifiers. Boosting helps to provide the ability and the requirement to train a highly precise classifier by taking a weighted average of the decisions made by the weak learners. Each stage is trained using a technique called boosting.

Each stage of the classifier labels the on-going location of the sliding window as either positive or negative. Negative indicates that an object was not found. Positive indicates objects were found. The classification of this region is complete, if the label is negative. If the label is positive, the classifier passes the region to the next stage.

The above stages are designed to reject the negative samples always and as quick as possible. Assumption is made that the majority of the windows does not contain any object of interest. True positives are rare.

- When a positive sample is correctly classified A true positive occurs.
- when a negative sample is mistakenly classified as positive A false positive occurs
- when a positive sample is mistakenly classified as negative A false negative occurs

To make sure they work well, a low false negative rate must be there in each stage in the cascade. If a stage incorrectly labels an object as negative, the classification stops and mistake cannot be corrected. Although, a high false positive rate can be had. You can correct the mistake in subsequent stages even if the detector incorrectly labels a no object as positive. Adding more stages reduces true positive rate, but it further reduces the overall false positive rate.



Fig 7.4 Training stages

➤ **Open CV**

OpenCV is a cross-platform library where we can construct real-time computer vision applications. We take inputs from the RPI camera which sends real-time video to the RPI board. These videos are splitted into frames and fed into the algorithm.

CNN are applied in many image recognition problems in order to classify the object within the image, the CNN must know what to look for. Once the CNN has been trained with the training dataset the CNN could interpret the traffic signals.

The CNN consists of three major layers: output layer, input layer and hidden layer. The first layer is the convolutional layer where the input is given. Here the image input is always in the matrix form that is, a filter is applied to an image and the pixels are recorded into matrix form. In the hidden layers, the image undergoes through many filters and becomes a linear pixel array. The output is according to the linear pixel array. OpenCv is the algorithm that we are using for the detection of lanes.

OpenCV is used for the following:

- Image Processing – image manipulation is its main focus.
- Pattern Recognition – Has and explains various techniques to classify patterns.

➤ **Sensor circuitry**

We'll be using four pins on the Raspberry Pi for this project:

- GPIO 5V that is Pin 2 connected to Vcc (5V Power)
- GPIO GND that is Pin 6 connected to GND (0V Ground)
- GPIO 23 that is Pin 16 connected to TRIG (GPIO Output)
- GPIO 24 that is Pin 18 connected to ECHO (GPIO Input)
- Plug Vcc into positive rail of breadboard.
- Plug ground into negative rail of breadboard
- Plug GPIO 5V [Pin 2] into the positive rail
- GPIO GND [Pin 6] into the negative rail.
- TRIG directly into GPIO 23.
- link another blank rail using R1 (1k Ω resistor), plug ECHO into a blank rail

- Link your GND rail with the rail using R2 (2k Ω resistor).
- Add GPIO 24 to the rail with your R1

The ultrasonic sensor will always output 0V unless and until it has been triggered, in which case it will output 5V. So for this we need to set one GPIO pin out output that is used to trigger the sensor and another one GPIO which will be used as input to detect the voltage change.

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)

TRIG = 23
ECHO = 24

print "Distance Measurement In Progress"

GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

GPIO.output(TRIG, False)
print "Waiting For Sensor To Settle"
time.sleep(2)

GPIO.output(TRIG, True)
time.sleep(0.00001)
GPIO.output(TRIG, False)

while GPIO.input(ECHO)==0:
    pulse_start = time.time()

while GPIO.input(ECHO)==1:
    pulse_end = time.time()

pulse_duration = pulse_end - pulse_start

distance = pulse_duration * 17150

distance = round(distance, 2)

print "Distance:",distance,"cm"

GPIO.cleanup()
```

Fig 7.5 Sensor code

➤ Motor circuitry

Following are the steps for motor connection:

- Connect up the power and ground wires

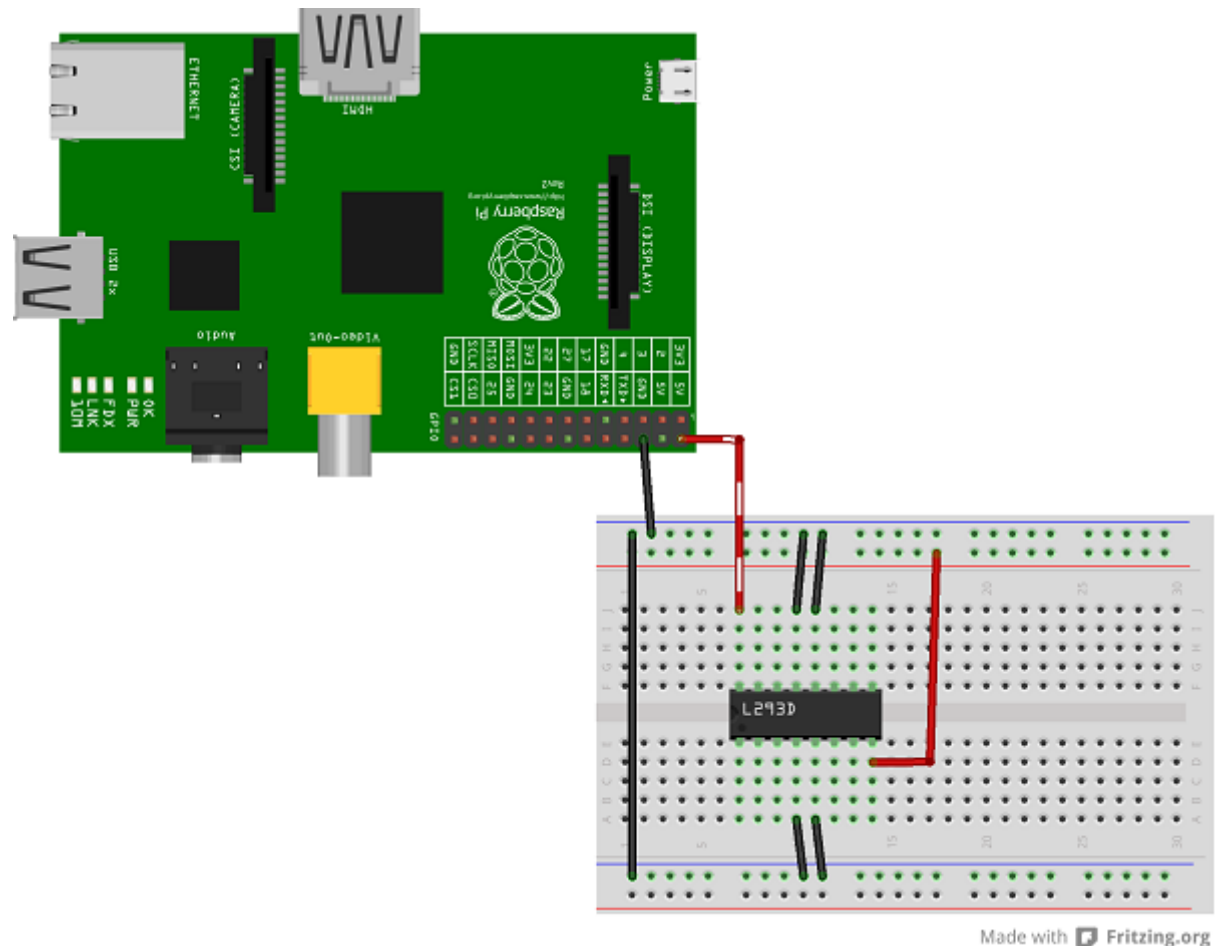


Fig 7.6 Motor Circuitry

- Adding the Data Wires: GPIO 25–Pin 22 > L293D–Pin 1, 24–Pin 18 > L293D–Pin , GPIO 23–Pin 16 > L293D–Pin 7
- Add the motor: Motor–wire 1 > L293D–pin 3, Motor–wire 2 > L293D–pin 6.
- Add the second motor: GPIO 11–Pin 23 > L293D–Pin 9, GPIO 9–Pin 21 > L293D–Pin 10, GPIO 10–Pin 19 > L293D–Pin 15

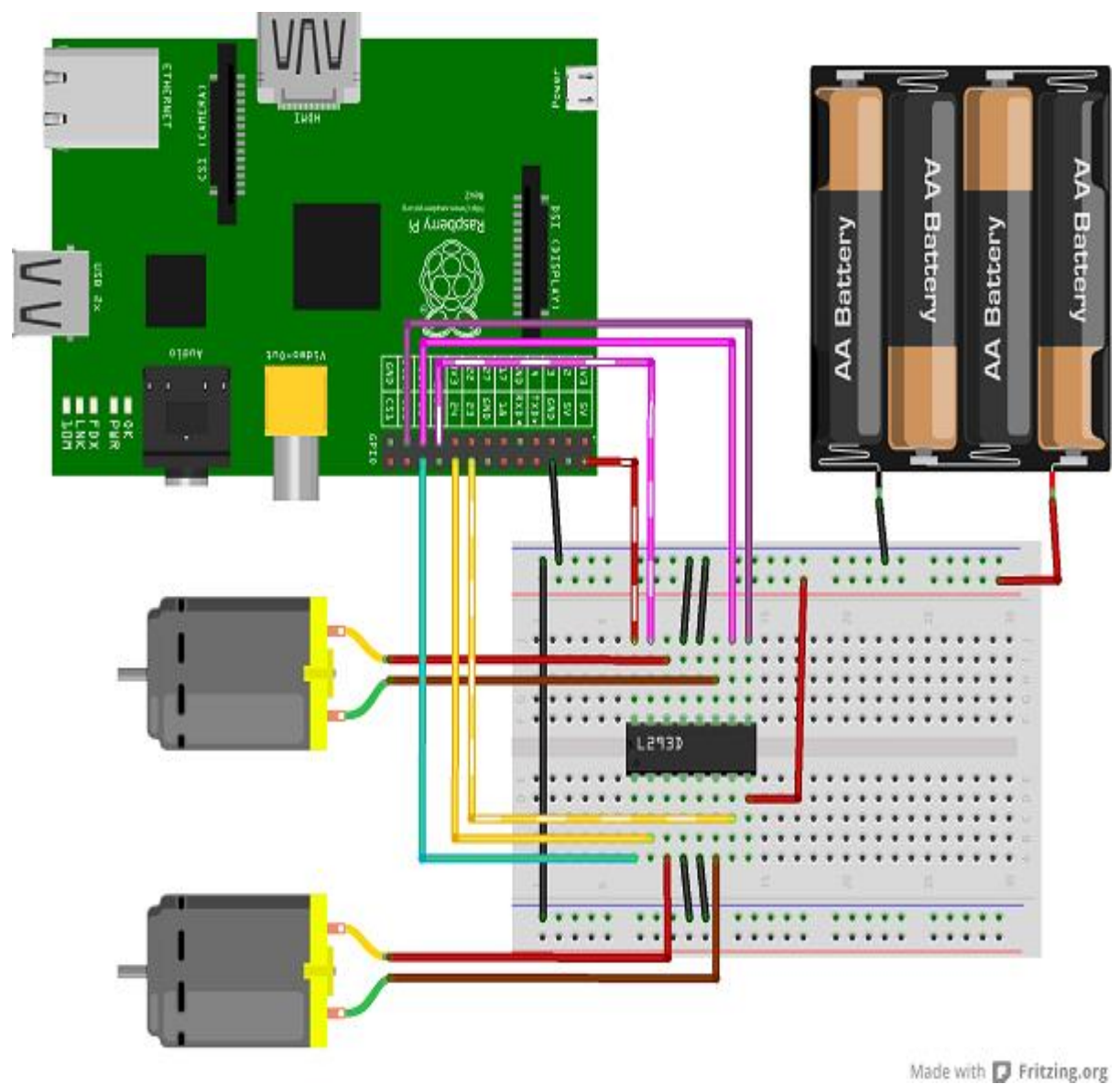


Fig 7.7 Motor circuit

////

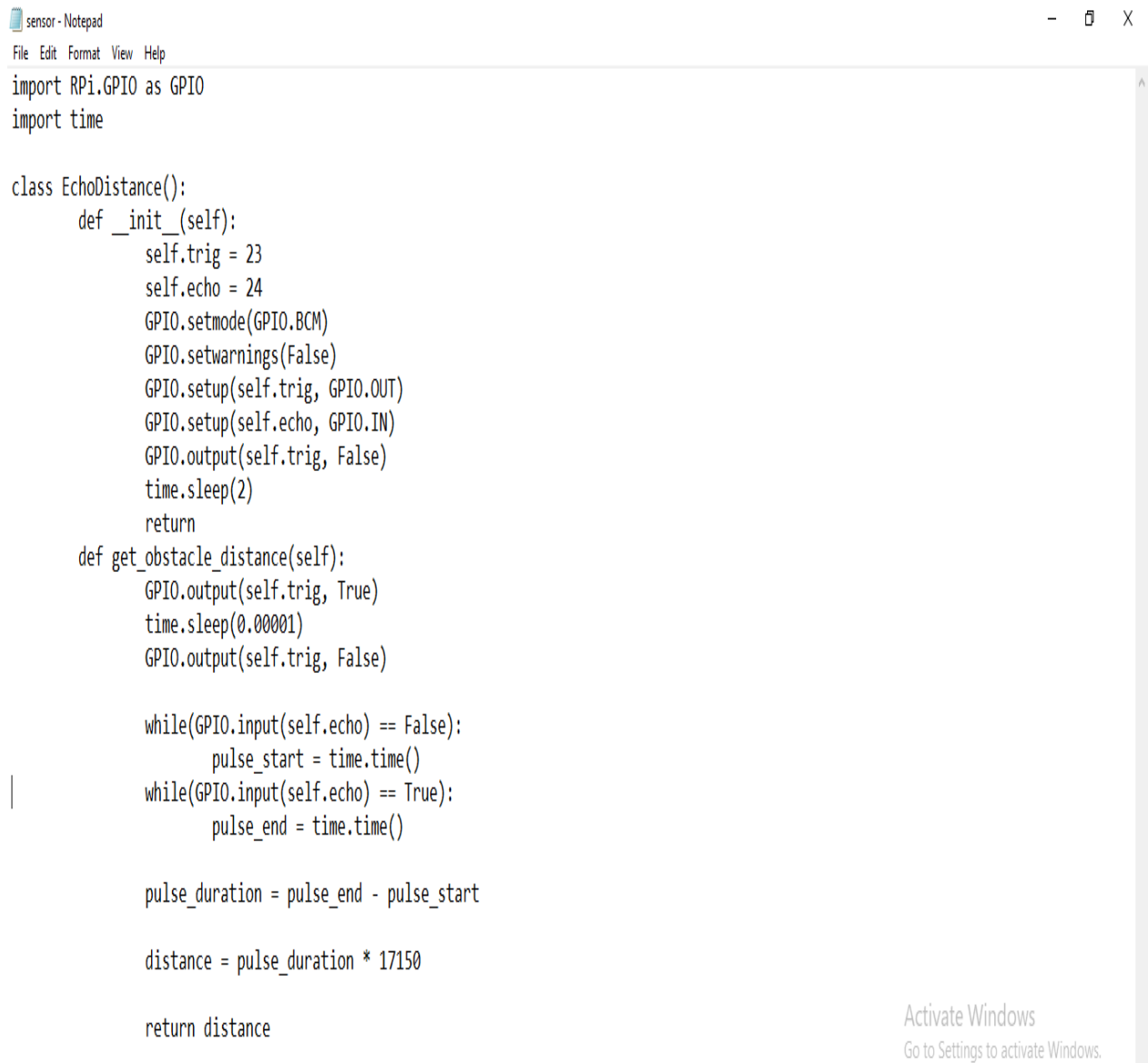
Motor sample code:

```
01  import RPi.GPIO as GPIO
02  from time import sleep
03
04  GPIO.setmode(GPIO.BOARD)
05
06  Motor1A = 16
07  Motor1B = 18
08  Motor1E = 22
09
10  GPIO.setup(Motor1A,GPIO.OUT)
11  GPIO.setup(Motor1B,GPIO.OUT)
12  GPIO.setup(Motor1E,GPIO.OUT)
13
14  print "Turning motor on"
15  GPIO.output(Motor1A,GPIO.HIGH)
16  GPIO.output(Motor1B,GPIO.LOW)
17  GPIO.output(Motor1E,GPIO.HIGH)
18
19  sleep(2)
20
21  print "Stopping motor"
22  GPIO.output(Motor1E,GPIO.LOW)
23
24  GPIO.cleanup()
```

➤ SOURCE CODES

➤ ULTRASONIC SENSOR

The code for controlling the ultrasonic sensor installed in our system is:



```
sensor - Notepad
File Edit Format View Help
import RPi.GPIO as GPIO
import time

class EchoDistance():
    def __init__(self):
        self.trig = 23
        self.echo = 24
        GPIO.setmode(GPIO.BCM)
        GPIO.setwarnings(False)
        GPIO.setup(self.trig, GPIO.OUT)
        GPIO.setup(self.echo, GPIO.IN)
        GPIO.output(self.trig, False)
        time.sleep(2)
        return
    def get_obstacle_distance(self):
        GPIO.output(self.trig, True)
        time.sleep(0.00001)
        GPIO.output(self.trig, False)

        while(GPIO.input(self.echo) == False):
            pulse_start = time.time()
        while(GPIO.input(self.echo) == True):
            pulse_end = time.time()

        pulse_duration = pulse_end - pulse_start

        distance = pulse_duration * 17150

        return distance
```

Activate Windows
Go to Settings to activate Windows.

➤ HAAR CASCADE

The code for controlling the start/stop sign using Haar cascade is:

```
import RPi.GPIO as GPIO
import time

class MoveCar():
    def __init__(self):
        self.forward=2
        self.backward=3
        self.left=4
        self.right=17
        self.control=''
        self.runtime=0.1

    def initialize(self):
        GPIO.setmode(GPIO.BCM)
        GPIO.setwarnings(False)
        GPIO.setup(self.forward, GPIO.OUT)
        GPIO.setup(self.backward, GPIO.OUT)
        GPIO.setup(self.left, GPIO.OUT)
        GPIO.setup(self.right, GPIO.OUT)

    def move_forward(self):
        self.control='f'
        GPIO.output(self.backward,0)
        GPIO.output(self.forward,1)

    def move_backward(self):
        self.control='b'
        GPIO.output(self.forward,0)
        GPIO.output(self.backward,1)
```

/

```
def move_left(self):
    GPIO.output(self.right,0)
    GPIO.output(self.left,1)

def move_right(self):
    GPIO.output(self.left,0)
    GPIO.output(self.right,1)

def move_stopall(self):
    GPIO.output(self.left,0)
    GPIO.output(self.right,0)
    GPIO.output(self.backward,0)
    GPIO.output(self.forward,0)

def move_stopfb(self):
    GPIO.output(self.backward,0)
    GPIO.output(self.forward,0)

def move_stoplr(self):
    GPIO.output(self.left,0)
    GPIO.output(self.right,0)

def brake(self):
    if(self.control==''):
        pass
    elif(self.control=='f'):
        print "Applying brake."
        self.move_backward()
        time.sleep(self.runtime)
```

```
        self.move_stopfb()
        self.control=''
    elif(self.control=='b'):
        print "Applying brake."
        self.move_forward()
        time.sleep(self.runtime)
        self.move_stopfb()
        self.control=''
    else:
        pass
```

```
def close(self):
    GPIO.cleanup()
```

- The code for the autonomous movement of the car is:

```
autorun - Notepad
File Edit Format View Help
import os
import cv2
import time
import numpy as np
import csv
from picamera import PiCamera
from picamera.array import PiRGBArray
import movecar
import pybrain
from pybrain.tools.xml.networkreader import NetworkReader
import stopdetect
import echodistance
import RPi.GPIO as GPIO
import thread
import sys

def write_to_file(msg):
    f = open("/var/www/html/decision.txt", 'wb')
    f.write(msg)
    f.flush()
    f.close()

def write_to_image(img):
    cv2.imwrite("/var/www/html/track.jpg", img)

motion = movecar.MoveCar()
motion.initialize()
motion.runtime = 0.5

stop_object = stopdetect.StopDetect()

echo_object = echodistance.EchoDistance()

red = 21
yellow = 20
green = 26
button = 16

GPIO.setup(red, GPIO.OUT)
GPIO.setup(yellow, GPIO.OUT)
GPIO.setup(green, GPIO.OUT)
GPIO.setup(button, GPIO.IN, pull_up_down=GPIO.PUD_UP)

GPIO.output(red, 0)
GPIO.output(yellow, 0)
GPIO.output(green, 0)

print "Loading network..."
write_to_file("Loading network...")

GPIO.output(yellow, 1)
if os.path.isfile('/home/pi/Documents/Scripts/AutonomousCar/network.xml'):
    fnn = NetworkReader.readFrom('/home/pi/Documents/Scripts/AutonomousCar/network.xml')
    print "Loaded..."
    write_to_file("Loaded...")
else:
    print "Network not present..."
    write_to_file("Network not present...")
```

```
exit(0)

GPIO.output(yellow, 0)

while(True):
    GPIO.output(red, 1)
    while(GPIO.input(button)==True):
        print "Waiting to start..."
        write_to_file("Waiting to start...")
        continue
    GPIO.output(red, 0)
    print "Button press: Starting autonomous run."
    write_to_file("Button press: Starting autonomous run.")
    try:
        GPIO.output(green, 1)

        time.sleep(1)

        resolution_x = 150
        resolution_y = 75

        camera = PiCamera()
        camera.resolution = (resolution_x, resolution_y)
        camera.framerate = 15

        rawCapture = PiRGBArray(camera, size=(resolution_x, resolution_y))

        time.sleep(0.1)
```



```
for frame in camera.capture_continuous(rawCapture, format='bgr', use_video_port=True):
    print "Image Captured..."

    color_image = frame.array
    rawCapture.truncate(0)

    if(GPIO.input(button)==False):
        print "Button press: Stopping autonomous run."
        write_to_file("Button press: Stopping autonomous run.")
        raise 1

    obstacle_distance = echo_object.get_obstacle_distance()
    if obstacle_distance <= 40.0:
        motion.brake()
        motion.move_stopall()
        cv2.imwrite("/var/www/html/track.jpg", color_image)
        print "Obstacle found at "+str(obstacle_distance)+" cm.\n"
        f = open("/var/www/html/decision.txt", 'wb')
        f.write("Obstacle found at "+str(obstacle_distance)+" cm.\n")
        f.flush()
        f.close()
        GPIO.output(red, 1)
        continue

    stops = stop_object.detect_stops(color_image)
    stops = sorted(stops, key=lambda l:l[1], reverse=True)
    stop_command = 0
    stop_distance = 0
```

```
        if len(stops)!=0:
            if stops[0][1] < 35.0:
                stop_command = 1
                stop_distance = stops[0][1]
                x, y, w ,h = stops[0][0]

        if stop_command == 1:
            motion.brake()
            # motion.move_stopall()
            cv2.rectangle(color_image, (x,y), (x+h,y+w), (0,255,0), 2)
            cv2.imwrite("/var/www/html/track.jpg", color_image)
            f = open("/var/www/html/decision.txt", 'wb')
            f.write("Stop found at "+str(stop_distance)+" cm.\n")
            f.flush()
            f.close()
            print "Stop found at "+str(stop_distance)+" cm.\n"
            GPIO.output(red, 1)
            time.sleep(3)
            continue

        if True:
            GPIO.output(red, 0)
            img = cv2.cvtColor(color_image, cv2.COLOR_BGR2GRAY)
            command = np.argmax(fnn.activate(np.ravel(img)))

            if(command == 0):
                #Move forward
                motion.move_stoplr()
                motion.move_forward()
```

```
        elif(command == 1):
            #Move backward
            motion.move_stoplr()
            motion.move_backward()

        elif(command == 2):
            #Move left
            motion.move_left()
            motion.move_forward()

        elif(command == 3):
            #Move right
            motion.move_right()
            motion.move_forward()

        else:
            #Do nothing
            motion.move_stopfb()

        cv2.imwrite("/var/www/html/track.jpg", color_image)
        thread.start_new_thread(write_to_image,(color_image,))
        direction = {0:'Moving forward', 1:'Moving backward', 2:'Moving left', 3:'Moving right'}
        f = open("/var/www/html/decision.txt", 'wb')
        f.write(direction[command])

        f.flush()
        f.close()
        print "Decision : "+str(direction[command])+"\n"

    except:
        camera.close()
        GPIO.output(green, 0)
        GPIO.output(red, 1)
        motion.move_stopall()
        time.sleep(1)
motion.close()
```

CONCLUSION

The autonomous car once implemented into the market to be utilized by users provide a lot of advantages. It provides safety to the needy like number of accidents can be reduced. Self-driving cars can help the people who are unable to drive themselves, such as the elderly or disabled. Accessible, affordable, and convenient self-driving cars could increase the total number of kilometers driven each year. Other industries such as health care, restaurant, military, fire brigade etc, can make use of self-driving technology for their benefits. Hospitals can implement this technology to their ambulances which will help people who are on the verge of death to the hospital as fast as possible.

REFERENCES

- [1] Cameron Franke, “Autonomous Driving with a Simulation Trained Convolutional Neural Network”,in University of the Pacific Theses and Dissertations,2018
- [2] Hamada R. H. Al-Absi ,Justin Dinesh Daniel Devaraj ,Patrick Sebastian , Yap Vooi Voon, “Vision based Automated Parking System”, IEEE Conference paper, 2010.
- [3] Lex Fridman , Daniel E. Brown, Michael Glazer, William Angell, Spencer Dodd, Benedikt Jenik, Jack Terwilliger, Julia Kindelsberger, Li Ding, Sean Seaman, Hillary Abraham, Alea Mehler, Andrew Sipperley, Anthony Pettinato, Bobbie Seppelt, Linda Angell, Bruce Mehler, Bryan Reimer, “MIT Autonomous Vehicle Technology Study: Large-Scale Deep Learning Based Analysis of Driver Behaviour and Interaction with Automation”, September 2018.
- [4] Supriya K Kokate, Dr M H Nerkar, “Review on Autonomous Car using Raspberry Pi”,published in International Journal for Research in Applied Science & Engineering Technology (IJRASET) ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887 Volume 6 Issue I in 2018.
- [5] E. H. Miller, “International Journal of Engineering Research in Electronics and Communication Engineering(IJERECE) Vol 5, Issue 4”, published in April 2018
- [6] Juan Rosenzweig,Michael Bartl,“AUTONOMOUS VEHICLE - LITERATURE REVIEW”, 305AEE, 2015
- [7] Minal Zunjarrao, Sayali Shirude,“The International Journal of Engineering and Science (IJES)-Adaptive Cruise Control”.

CHAPTER 8

PUBLICATION

Paper Title: Automated driving car using RPI board and CNN algorithm

Authors: 1.Prashant Holla 2.Suveda N 3.Sharan C 4.Nithyanand SN

Date: 10th April 2019

Conference: 5th National Conference on Advancements in Information Technology

Journal: Journal of Computer Science Engineering and Software Testing



Fig. 8.1 Certificate of winning paper presentation for Automated driving car using RPI board and CNN algorithm



. 8.2 Certificate of participation of paper presentation for Automated driving car using RPI board and CNN algorithm

Report AUTOMATED DRIVING CAR USING RPI BOARD AND CNN ALGORITHM

ORIGINALITY REPORT

26%

SIMILARITY INDEX

23%

INTERNET SOURCES

9%

PUBLICATIONS

19%

STUDENT PAPERS

PRIMARY SOURCES

1

theijes.com

Internet Source

4%

2

www.iosrjournals.org

Internet Source

3%

3

www.willberger.org

Internet Source

2%

4

research.ijcaonline.org

Internet Source

2%

5

www.technoarete.org

Internet Source

2%

6

projects-raspberry.com

Internet Source

2%

7

Submitted to Coventry University

Student Paper

2%

8

pt.scribd.com

Internet Source

1%

9

pdfs.semanticscholar.org

	Internet Source	1 %
10	www.ucsusa.org Internet Source	1 %
11	www.mathsjournal.org Internet Source	1 %
12	Submitted to American University of the Middle East Student Paper	1 %
13	www.i-scholar.in Internet Source	1 %
14	Submitted to M S Ramaiah University of Applied Sciences Student Paper	<1 %
15	answers.opencv.org Internet Source	<1 %
16	Submitted to Laureate Education Inc. Student Paper	<1 %
17	Submitted to Siddaganga Institute of Technology Student Paper	<1 %
18	Alajlan, Abrar M., Marwah M. Almasri, and Khaled M. Elleithy. "Multi-sensor based collision avoidance algorithm for mobile robot", 2015 Long Island Systems Applications and	<1 %

Technology, 2015.

Publication

19	allenlu2007.wordpress.com Internet Source	<1 %
20	Submitted to Sunway Education Group Student Paper	<1 %
21	docs.opencv.org Internet Source	<1 %
22	www.theseus.fi Internet Source	<1 %
23	irjet.net Internet Source	<1 %
24	Submitted to University of Bedfordshire Student Paper	<1 %
25	export.arxiv.org Internet Source	<1 %
26	www.mathworks.com Internet Source	<1 %
27	Submitted to University of Newcastle upon Tyne Student Paper	<1 %
28	Submitted to Universiti Teknikal Malaysia Melaka Student Paper	<1 %

29	Zhiguo Yan, Zheng Xu, Jie Dai. "The Big Data Analysis On The Camera-Based Face Image In Surveillance Cameras", Intelligent Automation and Soft Computing, 2018 Publication	<1 %
30	Ahmed Aly Ibrahim, Zainhom Sayed Ghareeb, Omar M. Shehata, El-Sayed Imam Morgan. "A Robotic Surveillance Platform Based on an On-board Computer Vision Approach", Proceedings of the 4th International Conference on Control, Mechatronics and Automation - ICCMA '16, 2016 Publication	<1 %
31	Submitted to Higher Education Commission Pakistan Student Paper	<1 %
32	Submitted to University of the Arts, London Student Paper	<1 %
33	Submitted to Universiti Teknologi MARA Student Paper	<1 %

Exclude quotes	Off	Exclude matches	< 15 words
Exclude bibliography	On		