Implementation of Computer Vision-Based Obstacle Avoidance and Self-Steering System for Autonomous Vehicles

A Project Work submitted to

Visvesvaraya Technological University
in partial fulfilment of the requirements
for the award of degree of

Bachelor of Engineering
in
Electronics and Communication Engineering

Submitted by

Tejas M	**1KG17EC085**
Tejesh K	**1KG17EC086**
Sudha K	**1KG17EC083**
Varshini G Bali	**1KG17EC089**

Under the Guidance of

Mr. Ravikiran B.A.
Assistant. Professor
Department of ECE
KSSEM, Bengaluru

Department of Electronics and Communication Engineering
K.S. School of Engineering and Management
No. 15, Mallasandra, off Kanakapura Road, Bengaluru-560109
2020-21

K.S. School of Engineering and Management

No. 15, Mallasandra, off Kanakapura Road, Bengaluru-560109

Department of Electronics and Communication Engineering

Certificate

This is to certify that the project work entitled ***Implementation of a Computer Vision Based Obstacle Avoidance and Self-Steering System for Autonomous Vehicle*** *is a bonafide work carried out by*

Tejas M	1KG17EC085
Tejesh K	1KG17EC086
Sudha K	1KG17EC083
Varshini G Bali	1KG17EC089

in partial fulfilment for the award of ***Bachelor of Engineering in Electronics and Communication Engineering*** *of Visvesvaraya Technological University, Belgaum, during the year 2020-21. It is certified that all the suggestions indicated during internal assessment have been incorporated in the report and this thesis satisfies the academic requirement in respect of project work prescribed for the degree.*

Name and Signature of Internal Guide   Head of the Department   Principal/ Director

University Examiners:

Name and Signature of Examiner-1   Name and Signature of Examiner-2

K.S. School of Engineering and Management

No. 15, Mallasandra, off Kanakapura Road, Bengaluru-560109

Department of Electronics and Communication Engineering

Declaration

We,

Tejas M	1KG17EC085
Tejesh K	1KG17EC086
Sudha K	1KG17EC083
Varshini G Bali	1KG17EC089

the students of eight semester ***BE (Electronics and Communication Engineering)*** *declare that the project entitled* ***Implementation of a Computer Vision Based Obstacle Avoidance and Self-Steering System for Autonomous Vehicle*** *is carried out by us at* ***K.S. School of Engineering and Management*** *as a partial fulfilment of academic requirement of* ***BE in Electronics and Communication Engineering*** *under* ***Visvesvaraya Technological University****. The content in the thesis is not submitted to any other University either partially or wholly for the award of any other degree.*

Sl. No.	Reg No.	Name of Student	Signature with Date
1	1KG17EC085	Tejas M	
2	1KG17EC086	Tejesh K	
3	1KG17EC083	Sudha K	
4	1KG17EC089	Varshini G Bali	

Date: Bangalore

Acknowledgement

We are grateful to K. S. School of Engineering and Management, Bangalore for providing an encouraging environment during our studies.

After the months of sincere hard work, it is now the time for us to express our words of gratitude for people who are equally responsible and helped us in completing our project work on time.

We would like to thank our internal guide **Mr. Ravikiran B. A.,** Asst. Prof, Department of Electronics and Communication Engineering, K. S. School of Engineering and Management, for guiding us and supporting us throughout our project.

We would like to express our heartfelt gratitude to **Dr. Girish V. Attimarad**, Professor and HOD, Department of Electronics and Communication Engineering, K. S. School of Engineering and Management for the enormous support and motivation.

We show our immense gratitude to **Dr.K. Rama Narasimha** Principal, K. S. School of Engineering and Management, for giving us the facilities and timely support for carrying out the project work.

We extend our thanks to the entire faculty of the Department of Electronics and Communication, K. S. School of Engineering and Management, Bangalore who have encouraged us throughout this project. Last but not the least, we thank our family and friends for their invaluable help and support during the course of the project work.

-Project team

Tejas M	1KG17EC085
Tejesh K	1KG17EC086
Sudha K	1KG17EC083
Varshini G Bali	1KG17EC089

Abstract

A self-driving car, also known as an Autonomous Vehicle (AV), driverless car, or robo-car is a vehicle that is capable of sensing its environment and moving safely with little or no human input. The human error is one of the main reasons for road fatalities and accidents to overcome this problem.

The self-driving car upon filling the obstacles in the path has to find a way to overcome them. The above action is performed by initially detecting the obstacle through pi-cam and based on the data received from pi-cam the necessary action has to be performed. The necessary action to be performed is commanded by Raspberry pi to Arduino. The Arduino signals the motor driver to perform its operation. The distance of 40cm is maintained between the model and obstacle. The distance parameter is accurately calculated by model through the aid of pi-cam.

The self-driving car on road has to identify the markings on the lane or identify the road surface to navigate or steer seamlessly through its path. The above actions are performed by the image processing technique where various processes and parameters are utilized to identify the lane markings. Also, certain pointers are used to calibrate the exact position to be maintained on the road surface, based on the above methods the model can steer along the path seamlessly.

Contents

List of Figures

Chapter 1

INTRODUCTION

A **self-driving car**, also known as an **Autonomous Vehicle** (**AV**), **driverless car**, or **robo-car** is a vehicle that is capable of sensing its environment and moving safely with little or no human input. The human error is one of the main reasons for road fatalities and accidents to overcome this problem

Self-driving cars combine a variety of sensors to perceive their surroundings, such as cameras, radar, LIDAR, sonar, GPS, odometry and inertial measurement units. Advanced control systems interpret sensory information to identify appropriate navigation paths, as well as obstacles and relevant signage.

Fig 1.1: Self-Driving Cars

Automated vehicles are technological development in the field of automobiles. Now a days, due to inconvenience of public transportation peoples are using their private vehicles. Due to such a large number of vehicles, the traffic problem has been occurred. To resolve this traffic problem, traffic rules are designed. But disobey of such traffic rules causes accidents. And maximum accidents will be occurred due to human error. To reduce these accidents and to improve safety transportation we require Autonomous

Vehicle. Autonomous drive technology is one of the most important innovation in the automotive industry. If we will able to implement this technology and have total control over it, then it can result in large benefits for both individuals & society.

Autonomous cars will consist of up to 75% of the cars on the roads. Tens of millions of people have lost their lives or have become disabled worldwide in the last 10 years as a consequence of traffic accidents, the purpose of this project is to create a safe self-driving car that could help millions of people each year. Almost all the traffic accidents are caused by human mistakes. Unfortunately, according to statistic, in the next 10 years the number of lives lost each year will likely be doubled. To avoid such problems we are moving towards Autonomous Car.

An autonomous car is a vehicle capable of sensing its environment and operating without human involvement. A human passenger is not required to take control of the vehicle at any time, nor is a human passenger required to be present in the vehicle at all. An autonomous car can go anywhere a traditional car goes and do everything that an experienced human driver does.

Autonomous cars rely on sensors, actuators, complex algorithms, machine learning systems, and powerful processors to execute software.

Autonomous cars create and maintain a map of their surroundings based on a variety of sensors situated in different parts of the vehicle. Radar sensors monitor the position of nearby vehicles. Video cameras detect traffic lights, read road signs, track other vehicles, and look for pedestrians. Lidar (light detection and ranging) sensors bounce pulses of light off the car’s surroundings to measure distances, detect road edges, and identify lane markings. Ultrasonic sensors in the wheels detect curbs and other vehicles when parking.

Sophisticated software then processes all this sensory input, plots a path, and sends instructions to the car’s actuators, which control acceleration, braking, and steering. Hard-coded rules, obstacle avoidance algorithms, predictive modeling, and object recognition help the software follow traffic rules and navigate obstacles.

1.1 History of Autonomous Vehicles

In GM’s 1939 exhibit, Norman Bel Geddes created the first self-driving car, which was an electric vehicle guided by radio-controlled electromagnetic fields generated with magnetized metal spikes embedded in the roadway. By 1958, General Motors had made this concept a reality. The car’s front end was embedded with sensors called pick-up coils that could detect the current flowing through a wire embedded in the road. The current could be manipulated to tell the vehicle to move the steering wheel left or right.

In 1977, the Japanese improved upon this idea, using a camera system that relayed data to a computer to process images of the road. However, this vehicle could only travel at speeds below 20 mph. Improvement came from the Germans a decade later in the form of the VaMoRs, a vehicle outfitted with cameras that could drive itself safely at 56 mph. As technology improved, so did self-driving vehicles’ ability to detect and react to their environment.

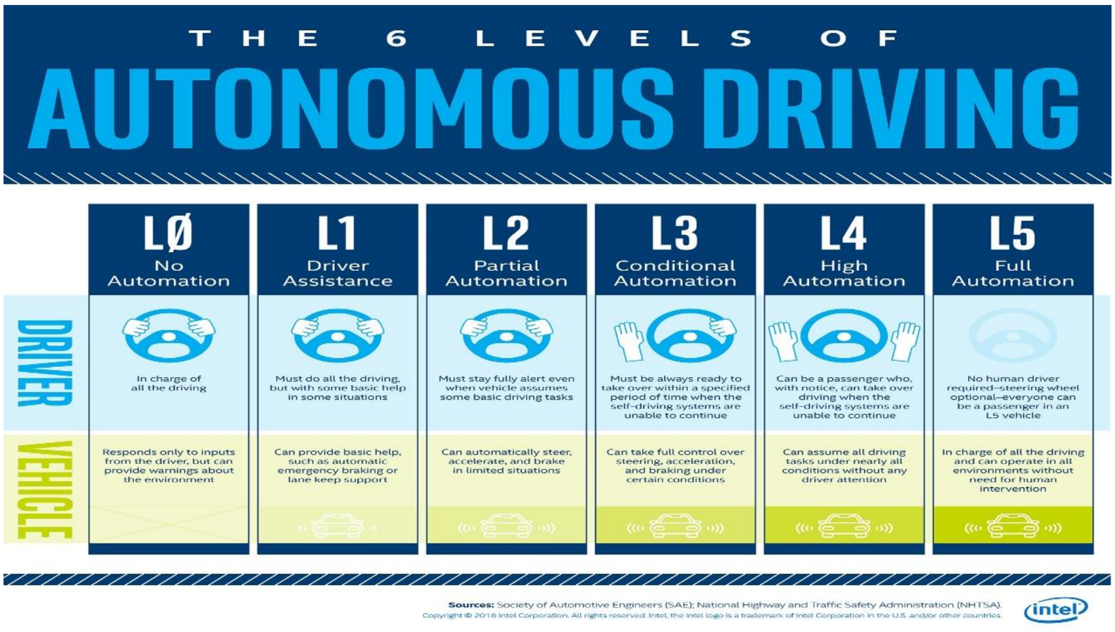
1.2 Autonomous Cars Today

At present, many vehicles on the road are considered to be semi-autonomous due to safety features like assisted parking and braking systems, and a few have the capability to drive, steer, brake, and park themselves. Autonomous vehicle technology relies on GPS capabilities as well as advanced sensing systems that can detect lane boundaries, signs and signals, and unexpected obstacles. While the technology isn’t yet perfect, it’s expected to become more widespread as it improves, with some predicting that up to half of the automobiles rolling off of assembly lines worldwide will be autonomous by 2025. Dozens of states already have legislation on the books concerning the use of autonomous vehicles in preparation for when this technology is commonplace.

Autonomous vehicles are expected to bring with them a few different benefits, but the most important one is likely to be improved safety on the roads. The number of accidents caused by impaired driving is likely to drop significantly, as cars can’t get drunk or high like human drivers can. Self-driving cars also don’t get drowsy, and they don’t have to worry about being distracted by text messages or by passengers in the vehicle. And a computer isn’t likely to get into an accident due to road rage. A 2015

National Highway Traffic Safety Administration report found that 94 percent of traffic accidents happen because of human error: By taking humans out of the equation, self-driving vehicles are expected to make the roads much safer for all.

1.3 Levels of Autonomous Driving



THE 6 LEVELS OF AUTONOMOUS DRIVING

	LØ No Automation	L1 Driver Assistance	L2 Partial Automation	L3 Conditional Automation	L4 High Automation	L5 Full Automation
DRIVER	In charge of all the driving	Must do all the driving, but with some basic help in some situations	Must stay fully alert even when vehicle assumes some basic driving tasks	Must be always ready to take over within a specified period of time when the self-driving systems are unable to continue	Can be a passenger who, with notice, can take over driving when the self-driving systems are unable to continue	No human driver required–steering wheel optional–everyone can be a passenger in an L5 vehicle
VEHICLE	Responds only to inputs from the driver, but can provide warnings about the environment	Can provide basic help, such as automatic emergency braking or lane keep support	Can automatically steer, accelerate, and brake in limited situations	Can take full control over steering, acceleration, and braking under certain conditions	Can assume all driving tasks under nearly all conditions without any driver attention	In charge of all the driving and can operate in all environments without need for human intervention

Sources: Society of Automotive Engineers (SAE); National Highway and Traffic Safety Administration (NHTSA).
Copyright © 2018 Intel Corporation. All rights reserved. Intel, the Intel logo is a trademark of Intel Corporation in the U.S. and/or other countries.

Fig 1.2: Levels of Autonomous Driving

1.3.1 - Level 0 : No Automation

Level 0 indicates that a vehicle does not have any assisted or automated driving technologies. All actions of the vehicle are dictated by a human. For example, a human must manually control speed, braking, and make stopping judgements. This also includes cars with features such as conventional cruise control systems because they do not operate unless they are set/adjusted by a human.

1.3.2 - Level 1 : Driver Automation

This is the lowest level of automation. The vehicle features a single automated system for driver assistance, such as steering or accelerating (cruise control). Adaptive cruise control, where the vehicle can be kept at a safe distance behind the next car, qualifies as Level 1 because the human driver monitors the other aspects of driving such as steering and braking.

1.3.3 - Level 2 : Partial Automation

Level 2 vehicles have two or more assisted driving technologies that work together simultaneously. For example, a level two vehicle can determine the speed of traffic ahead to coordinate its acceleration and steering on the highway, but the driver is required to be alert at all times to intercept when necessary. These vehicles are only capable of self-driving under certain conditions.

1.3.4 - Level 3 : Conditional Automation

Level 3 vehicles have limited automation driving functionality under certain conditions. The vehicle has the ability to make certain decisions without human judgement using sensors such as LiDAR. Humans are required to be on standby in case something doesn't go as intended, but the vehicle takes full control of operating when certain conditions are met during a route. An example of conditional automation is Audi's AI traffic jam pilot. The system can intervene at the point on a highway when a traffic jam is met and crawls inch by inch moving through it. Once traffic is cleared, the car alerts the driver to take back over as the condition is no longer met for it to operate on its own. The vehicle only monitors the environment under certain conditions, while all other circumstances fall on the driver.

1.3.5 - Level 4 : High Automation

Level 4 vehicles have full automation driving technology functionality in most conditions. However, under select circumstances human oversight is required, such as driving on back roads or if a blizzard were to occur and cause poor visibility. For instance, a driver may control all the movements of a vehicle on back roads and then transition full control over to the vehicle for the highway portion of the journey. The driver becomes a passenger when the vehicle is in self-driving mode under the established conditions and human input is not required.

1.3.6 - Level 5 : Full Automation

Level 5 vehicles have full automation technology and require no involvement of humans. In comparison to level 4 cars, these cars have much more advanced environmental detection systems with the ability to operate under all conditions. An example of this would be the vehicle’s ability to operate autonomously through elevated terrains. Since the function of the driver is completely eliminated, the vehicle does not have traditional parts such as a steering wheel, brake and acceleration pedals, or gear shift that cater to a human operation.

1.4 Machine Learning Algorithm :

Over the past two decades Machine Learning has become one of the mainstays of information technology and with that, a rather central, albeit usually hidden, part of our life. The ever-increasing amount of data which is becoming available gives a good reason to believe that smart data analysis will become even more pervasive as a necessary ingredient for technological progress. Machine learning is a subfield of computer science that evolved from the study of pattern recognition and computational learning theory in artificial intelligence. Machine learning dwells into the study and construction of algorithms that can learn from and make predictions on data. Such algorithms operate by building a model from example inputs in order to make data-driven predictions or decisions, rather than following strictly static program instructions. Machine learning is closely related to computational statistics; a discipline that aims at the design of algorithms for implementing statistical methods on computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms is infeasible. Machine learning is, to keep it simple, an algorithm developed to note changes in data and evolve in its design to accommodate the new findings. As applied to predictive analytics, this feature has wide ranging impact on the activities normally undertaken to develop, test, and refine an algorithm for a given purpose.

1.4.1 Machine Learning Algorithms Types:

Machine learning algorithms are organized into taxonomy, based on the desired outcome of the algorithm. Common algorithm types include:

- Supervised learning – In this, the algorithm generates a function that maps inputs to desired outputs. One standard formulation of the supervised learning task is the classification problem: the learner is required to learn (to approximate the behaviour of) a function which maps a vector into one of several classes by looking at several input-output examples of the function.

- Unsupervised learning – This models a set of inputs: labelled examples are not available.

- Semi-supervised learning - This combines both labelled and unlabelled examples to generate an appropriate function or classifier.

- Reinforcement learning – In this, the algorithm learns a policy of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm.

- Transduction – It is similar to supervised learning, but does not explicitly construct a function: instead, tries to predict new outputs based on training inputs, training outputs, and new inputs.

- Learning to learn – In this the algorithm learns its own inductive bias based on previous experience.

1.4.2 The Advantages of Machine Learning

There are several reasons why machine learning algorithms are increasingly being used. But here are two main advantages:

- Feature learning: One of the interesting advantages of machine learning is that a system randomly initialized and trained on some datasets will eventually learn good feature representations for a given task. Classical approach involved handcrafting features by an expert human. This took several years of painstakingly fine tuning several parameters to get it right. Nowadays machine learning is used to discover relevant features in otherwise disordered datasets. Such features can be useful for things such as face detection, face recognition, speech recognition or image classification. Deep learning particularly aims to build higher-level abstract feature representation of data layer by layer. These features can be very powerful in speech and image recognition.

• Parameter optimization: This is similar to feature learning as a group of tuneable parameters can be visualized as a feature. Machine learning mostly employs a gradient based method of optimizing a large array of parameters. Again, such parameters may be large in number for example; a deep neural architecture can have billions of tuneable parameters. These parameters when well set can result in a system working properly. It is not feasible for a human or humans to find such an optimal setting for large number of parameters by hand, thus large-scale machine learning algorithms such as stochastic gradient descent are used to find an optimal setting.

1.4.3 Applications

The most common applications of machine learning include are adaptive websites, affective computing, bio-informatics, brain-machine interfaces, classifying DNA sequences, detecting Credit Card and Internet fraud, Marketing, Natural Language Processing, Optimization and Metaheuristic, Robot locomotion, Sentiment analysis, Speech and Handwriting recognition, Lane and Sign recognition, Structural health monitoring and Syntactic pattern recognition.

1.5 PROBLEM STATEMENT:

Non-autonomous vehicles have been around several years and based on online survey we have found that ratio of accident happening due to human error is quite high somewhere between 94% and 96% of all car accidents are caused by human error. The results of the studies seem to argue in favour of the use of self-driving technology as it solves one of the main problems that is safety. The listed are several factors that often contribute to these accidents:

I. Distractions:

Distracted driving is quickly becoming one of the leading causes of vehicle accidents. And when they are driving, they are often tempted to quickly look down at their phone to read or send a text message. But even looking away from the road for a few seconds can result in disastrous consequences.

II. Drowsiness and Fatigue:

A surprising number of people drive when they are tired or sleepy.

III. Aggressive/Reckless Driving:

A large number of vehicle accidents are caused by aggressive driving behaviour's that sometimes cross over into recklessness. Examples include speeding, tailgating, weaving in and out of lanes, running stop lights and stop signs, making dangerous or illegal turns, dangerous or illegal passing manoeuvres, and many others. These types of behaviours greatly increase the chances of a car crash.

IV. Traffic congestion is found to increase.

1.6 Organisation of Project Report

The report on implementation of computer vision based obstacle avoidance and self – steering system for autonomous vehicles is organized as follows:

Chapter 1: Introduction: This chapter gives the overall information about the project. It includes the background knowledge and problem statement.

Chapter 2: Literature Review: This chapter gives details about the survey conducted during the project.

Chapter 3: Requirements Specifications: It deals with aim, objectives and methodologies.

Chapter 4: Design and Implementation: Overview gives the details of the block diagram and proposed system of computer vision based obstacle avoidance and self-steering system for autonomous vehicle.

Chapter 5: Hardware and Software Requirements: It mentions about hardware and software requirements for the project and its implementation.

Chapter 6: Testing: This chapter includes the results and its corresponding snapshots and its readings have been recorded.

Chapter 7: Conclusion and Future Work: In this chapter the main results obtained are chalked out and discussion is carried out to find if the objectives defined earlier is met. It also discusses the enhancement that can be made so as to make the model more efficient and friendlier.

Chapter 8: References: This section deals with various papers referred for the development of the project.

Chapter 2

LITERATURE REVIEW

Paper 1:

Implementation And Demonstration Of Obstacle Detection In Self Driving Cars With GPS Tracking In Real-Time System.

Author: Dr. Ashoka S B, Lakshmikanth D

The model in this paper performs two main features detecting obstacles and tracking the vehicle's location. Detecting the obstacle with the use of ultrasonic sensor which works with the help of reflecting the sound. The sound waves transmitted continuously by the ultrasonic sensor scans the surroundings of the robotic car if the sound waves gets reflected by an obstacle the signal is send to microcontroller. The microcontroller decides which path is obstacle free and moves towards the obstacle free zone. The model in this paper takes its own decision based on the signals generated and move towards free zone and tracks the vehicles location using GPS neo-6 module. This paper proposed a design of embedded controller for self-driving car in a unique way. We are able to track the self-driving car using GPS NEO-6 module and also we can find obstacles come across the path by using ultrasonic sensor. The design is suitable in all weather conditions with high performance rate. We have designed this module in such a way that it follows traffic rules without any interventions. The obstacles are detected within the short range of 200 meters to 500 meters. The future implementation of our paper can add more applications of self-driving car that is cars can be operated using public voice commands, Traffic signal can be detect using image recognition and parking assistance can be provided.

Paper 2:

Real Time Lane Detection For Autonomous Vehicles

Author: Abdulhakam.AM.Assidiq, Othman O. Khalifa, Md. Rafiqul Islam, Sheroz Khan

The system used in this paper uses series of images. Out of these series various frames are used. The algorithm conducts image segmentation and remove the shadow of the road. The lanes are long and have curves hence they are considered as straight lines for reasonable range for safety of vehicles. The lanes are detected using Hough Transformation with restricted area. The proposed lane detection in this paper can be applied in both painted and unpainted roads as well as slightly curved and straight roads. In this paper, a vision-based lane detection approach capable of reaching real time operation with robustness to lighting change and shadows is presented. The system acquires the front view using a camera mounted on the vehicle then applying few processes in order to detect the lanes. Using a pair of hyperbolas which are fitting to the edges of the lane, those lanes are extracted using Hough transform. The proposed lane detection system Hough transform. The proposed lane detection system can be applied on both painted and unpainted road as well as curved and straight road in different weather conditions. This approach was tested and the experimental results show that the proposed scheme was robust and fast enough for real time requirements. Eventually, a critical overview of the methods were assist.

Paper 3:

Enhancement of canny edge detection Algorithm

Author: Sumeyya ilkin, Selin hangisi, Merve tafrali, Suhap sahin

The One of the most important aspects of image processing is the algorithms used for edge detection. Different methods are used to find the edges on the images. Canny edge detection algorithm is one of the most successful methods. Although it yields successful results, the processing load is very complicated. In this study, to enhance the canny edge detection algorithm, enhancement was performed on the canny edge detection algorithm using kernels which used in Sobel, Robert and Prewitt methods. The enhanced Canny edge detection algorithm was compared using Canny edge detection algorithm which using 3x3 gauss kernel and Canny edge detection algorithm which using a 5x5 gauss kernel. Study was tested using a special dataset. The results obtained after the test phase were compared using Mean Square Error (MSE), Peak Signal-to-Noise Ratio (PSNR), Correlation Index (CI) and Time elapsed (TE) metrics. As a result of the comparison, it was determined that enhanced Canny edge detection algorithm which using the Sobel kernel gives the most effective result.

This paper explains us how to transform an image taken by simple pin whole camera into a top view perspective of an image. This method helps increase the scope of extracting information from images. Different perspective view is one of the key requirements in wireless sensor network image based monitoring systems. This is a practical feasible technique and can be applied to different applications. This technique applies a simple rotation operation and follows a simple pin whole camera model technique, hence requires a very low processing time hence can be applied for low cost application products.

Paper 4:

Lane changing algorithm for autonomous car via virtual curvature mode

Author: M.C.Ho, P.T.Chan, A.B.Rad

This paper addresses the lane changing problem of autonomous vehicles when there is no road infrastructure support. The autonomous vehicle should drive from the current lane to the adjacent lane in the absence of a reference path to guide the vehicle to the new lane. We suggest an algorithm that incorporates a virtual road curvature with bicycle model for lane change guidance. As the name suggests, the virtual road curvature does not physically exist. It is a user assigned radius of a curved path which connects the current lane to the adjacent lane. Since the lateral sensor readings during lane changing maneuver are erroneous, the steering angle along with the virtual curvature is fed into a bicycle model to estimate the lateral position during the transition to the next lane. Details of the algorithm and the virtual road curvature determination are presented in the paper. In contrast to other lane changing methods, controller switching is not required and the same controller is for both lane keeping and lane changing. The algorithm is verified experimentally and the results are comparable with lane changing with physical transition lane.

Paper 5:

A Simple Bird's Eye View Transformation Technique

Author: M.Venkatesh, P.Vijayakumar

This paper explains us how to transform an image taken by simple pin hole camera into a top view perspective of an image. This method helps increase the scope of extracting information from images. Different perspective view is one of the key requirements in wireless sensor network image based monitoring systems. This is a practical feasible technique and can be applied to different applications. This technique applies a simple rotation operation and follows a simple pin hole camera model technique, hence requires a very low processing time hence can be applied for low cost application products. This technique can be classified under digital image processing as geometrical image modification.

Paper 6:

Vehicle Detection and Tracking from Video Captured by Moving Host

Author: Thathupara Subramanyan Kavya, Young-Min Jang, Tao Peng, Sang-Bock Cho

An efficient vision-based vehicle detection and tracking system is presented in this paper. One of the most vital stages of the driver assistance system is the detection and tracking. In this paper we propose an efficient system to detect and track vehicles from the video frames of a moving host. Vehicle detection and tracking system is an integral part of the driver assistance system. This paper has two major contributions (i) The proposed method has given excellent performance using a moving camera without additional sensors. (ii) The proposed method can detect fast moving and slow-moving objects and hence is suitable to use in real-time applications. Cascade classifiers are used to detect vehicles based on specific features and the classifier is trained using a set of positive and negative images. This technique proved robust against changes in orientation, size and colour. Alongside with Unscented Kalman filter, the limitations of the linear filters can be reduced which makes it suitable for real time vehicle tracking.

Paper 7:

Advancement of driverless cars and heavy vehicles using artificial intelligence (object detection)

Author: Balika J. Chelliah, Vishal Chauhan, Shivendra Mishra, Vivek Sharma

In this paper autonomous vehicle has many external sensors connected to it. By these external sensors it perceived or sense the environment and make decision accordingly. Real-time video object detection plays a major role in autonomous car for detection, obstacles detection and many more applications that helps an Autonomous Vehicle to work in efficient manner. Autonomous vehicles make use of different types of sensos together to decide on control of vehicle based on condition changes such as steering or applying thrust in case of emergency. However, detecting objects in a video stream frame-by-frame is difficult when delay of milliseconds can lead to collision. The main objective of this system is to train the model and testing many images of different objects in object detection algorithms. The specific objectives are:-

I. To basically feed different images of different objects that mostly a self-driving car will see like traffic lights, people, footpaths, fellow vehicles and many more.

Testing images using YOLO algorithm

Paper 8:

Deep learning based image recognition for autonomous driving

Author: Hironobu Fujiyoshi, Tsubasa Hirakawa, Takayoshi Yamashita

Haar Cascade Classifier method is object detection method. Haar-like feature is a feature that is widely used in object detection, which offers fast extraction process and able to represent low-resolution images. It's a machine learning based technique which uses a set of positive and negative images for training purpose. Object detection is the problem of finding the location of an object of a certain category in the image. Practical face detection and pedestrian detection are included in this task. Face detection uses a combination of Haar-like features and AdaBoost, and pedestrian detection uses HOG features and support vector machine (SVM). This chapter introduces end-to-end learning that can infer the control value of the vehicle directly from the input image as the use of deep learning for autonomous driving, and describes visual explanation of judgment grounds that is the problem of deep learning models and future challenges. In conventional machine learning (here, it is defined as a method prior to the time when deep learning gained attention), it is difficult to directly solve general object recognition tasks from the input image. This problem can be solved by distinguishing the tasks of image identification, image classification, object detection, scene understanding, and specific object recognition.

Paper 9:

Human face detection algorithm via Haar cascade classifer combined with three additional classifiers

Author: Li Cuimei, Qi Zhiliang, Jia Nan, Wu Jianhua

Haar Cascade Classifier method is object detection method developed by Viola Jones. Haar-like feature is a feature that is widely used in object detection, which offers fast extraction process and able to represent low-resolution images. This method has been successfully applied in many object detections including face detection and pedestrian detection. It's a machine learning based technique which uses a set of positive and negative images for training purpose. This method is quite fast and effective in detecting cars, signs and objects in real time. Viola and Jones' Haar-like features and cascade Classifiers the typical cascade classifier is the very successful method of Viola and Jones for face detection. Generally, many object detection tasks with rigid structure can be addressed by means of this method, not limited to face detection. Here the researchers propose a framework to combine several features into a cascade, i.e. a sequence of tests on the image or on particular regions of interest, organized into several stages, each based on the results of one or more different Haar-like features. When the object passes through all stages of classifiers, it gets recognized. To implement the framework, OpenCV along with a python wrapper called CV2 is used.

Paper 10:

Algorithm of autonomous vehicle steering system law estimation while the desired trajectory driving

Author: Sergey Sergeevich Shadrin, Andrey Mikhailovich Ivanov.

The article discusses an estimation algorithm of control actions on steering system in order to provide vehicle driving along desired trajectory with accounting of non-steady (transient) driving modes. During driving of an autonomous vehicle along a desired trajectory one of the most important tasks is to determine the exact control actions, including those on steering system. Thus, the vehicle drove along the same approximate trajectory, but with different control actions on steering wheel. Therefore, it can be stated, that with increase of vehicle velocity the driving along the same fixed trajectory is accompanied with decrease of steering wheel angle amplitude and phase advance shift.

Paper 11:

THE YAN: [Self Driving Car Using Deep Learning].

Author: Ms. Sujeetha, Chitrak Bari, Gareja Prdip, SiddhantPurohit.

In this paper, an autonomous robotically handled driving vehicle, using many features such as mapping, tracking and local planning. That can successfully create a car that can demonstrate proper lane changes, parking, and U-turns on its own. The different innovations we are using are obstacles and curb detection method, road vehicle tracker, and checking different traffic situations. This will make a robust autonomous self-driven car. Additionally a driverless car can reduce the time taken to reach the destination because it will take the shortest path, avoiding the traffic congestion. The human errors will be avoided thereby allowing disabled people (even blind people) to own their car. The idea of a car without a driver gives us a feeling of skepticism and we all will try to avoid it. An autonomous vehicle that will drive selflessly will need sensory input devices like cameras, radar and lasers for the car to perceive the world around it, creating a digital image to map.

Paper 12:

Application of ultrasonic sensors in road surface condition

Author: Shota Nakashima, Shingo Aramaki, Shenglin Mu, Kanya Tanaka

Ultrasonic sensor which is applied for detection of acoustic energy with wide frequency range above 20 kHz. There are two types of ultrasonic sensors namely, active and passive. Active sensors send and receive signals. Passive signals just receive the signal. The major criteria that has to be taken care when Self-Driving Car is taken into consideration is collision avoidance. The ultrasonic Sensor on the vehicle to avoid collision avoidance. The ultrasonic Sensor on the vehicle to avoid collision is the feasible solution to avoid accidents. The major task of ultrasonic was to transmit and receive the acoustic energy wave. Ultrasonic has the characteristics of centralized direction, good penetration and small amplitude which provide a higher amount of accuracy in real time. Steps to be followed for obstacle avoidance:

1. Watch the surroundings to calculate the distance of the obstacles from the car.
2. The minimum threshold distance that is safe for the car is 25cm. If the distance that is calculated comes out to be less than the threshold, stop the car and check other sides.
3. Rotate the car and move ahead.

Paper 13:

IoT based self-driving cars

Author: Rahul P Kharapakr, Abhishek S Khandare, Zeeshaan W Siddiqui.

This paper aims to represent a mini version of self-driving car using IOT with raspberry pi and Arduino UNO working as a main processor chip, the 8mp high resolution pi camera will provide the necessary information and the raspberry pi will analyze the data (samples) and it will get trained in pi with neural network and machine learning algorithm which would result in detecting road lanes, traffic lights and the car will take turns accordingly. In addition to these features the car will overtake with proper LED indications if it comes across an obstacle. Self-driving Cars technologically a reality and in the next decade they are expected to reach the highest level of automation. The brain of system is the Raspberry Pi which capable of exchanging data with the sensors and fast enough for calculate millions of data per second. Since our car will use deep learning it will need heavy parallel computing power for that reason the Raspberry Pi will useful here. The cars will be controlled by servo motor controller which is controlled by Arduino UNO, capable of reading its rotational speed. Camera will be used to find and detect objects that are then processed by the main controller (raspberry pi 3b+) within the car. Self-driving cars are technological development in the field of automobiles. Many companies throughout the world are making a serious and continuous effort to make driving a safe and risk-free process

Paper 14:

Robust lane marking detection and road geometry computation

Author: Tao Peng, Sang Bock Cho.

Detection of lane markings based on a camera sensor can be a low-cost solution to lane departure and curve-over-speed warnings. A number of methods and implementations have been reported in the literature. However, reliable detection is still an issue because of cast shadows, worn and occluded markings, variable ambient lighting conditions, for example. We focus on increasing detection reliability in two ways. First, we employed an image feature other than the commonly used edges: ridges, which we claim addresses this problem better. Second, we adapted RANSAC, a generic robust estimation method, to fit a parametric model of a pair of lane lines to the image features, based on both ridgeness and ridge orientation. In addition, the model was fitted for the left and right lane lines simultaneously to enforce a consistent result. Four measures of interest for driver assistance applications were directly computed from the fitted parametric model at each frame: lane width, lane curvature, and vehicle yaw angle and lateral offset with regard the lane medial axis. We qualitatively assessed our method in video sequences captured on several road types and under very different lighting conditions. We also quantitatively assessed it on synthetic but realistic video sequences for which road geometry and vehicle trajectory ground truth are known.

Paper 15:

Obstacle Detection and Safely Navigate the Autonomous Vehicle from Unexpected Obstacles on Driving Lane

Author: Malik Haris, Jin Hou

The model uses stochastic techniques such as curvature prepotential, depth variance potential and gradient potential is used to segment obstacles in the image from Markova random field [MRF] frames. Semantic segmentation is performed to segment paths and filter obstacles. The prediction of steering angle is done by analysing the data of unexpected obstacles on the roadway and determine the threat factor. The threat factor helps us to ignore or consider the obstacles. This research addresses how the robustness of obstacle detection method in a complex environment, by integrating a Markov random field (MRF) for obstacle detection, which is left intentionally or unintentionally on roadway, road segmentation, and CNN-model to navigate the autonomous vehicle safely. The designed model predicts the steering wheel angle after the high threat value found of obstacle on the roadway of autonomous vehicle for both manual and self-driving vehicle on the basis of analysis that our approach improves the safety of autonomous vehicle or human driving vehicle in terms of risk or collision with obstacles on the road by 35% as compared to vision-based navigation system rather than those who do not have the capability of detecting the high threat value obstacle on the road from long distances

Chapter 3

REQUIREMENT SPECIFICATIONS

3.1 AIM

Implementation of computer vision based obstacle avoidance and self-steering system for autonomous vehicles

3.2 OBJECTIVES

The main objective of this project is to design and implement computer vision based self-steering system and obstacle avoidance for autonomous vehicles using image processing and deep learning. To satisfy this, the objectives planned are:

- To carry out literature review on lane detection system for self-steering control on self-driving autonomous vehicles.
- To gather necessary information on various modules and procedures required to implement self-steering system for autonomous vehicles.
- To implement necessary machine learning algorithm to detect obstacle and measure distance of the obstacle using computer vision.
- To assemble the necessary hardware to design the actual self-driving car prototype.
- To compute and document the responses of the trained software.
- To test and validate self-driving car prototype response.

3.3 METHODOLOGY

METHODOLOGY 1:

The self-driving car upon filling the obstacles in the path has to find a way to overcome them. The above action is performed by initially detecting the obstacle through pi-cam and based on the data received from pi-cam the necessary action has to be performed. The necessary action to be performed is commanded by Raspberry pi to Arduino. The Arduino signals the motor driver to perform its operation. The distance of 40cm is maintained between the model and obstacle. The distance parameter is accurately calculated by model through the aid of pi-cam.

METHODOLOGY 2:

The self-driving car on road has to identify the markings on the lane or identify the road surface to navigate or steer seamlessly through its path. The above actions are performed by the image processing technique where various processes and parameters are utilized to identify the lane markings. Also, certain pointers are used to calibrate the exact position to be maintained on the road surface, based on the above methods the model can steer along the path seamlessly.

Chapter 4

DESIGN AND IMPLEMENTATION

The Project has the objective of designing and implementing a Vision Based self-steering system and obstacle avoidance for autonomous vehicles using image processing and deep learning.

4.1 BLOCK DIAGRAM

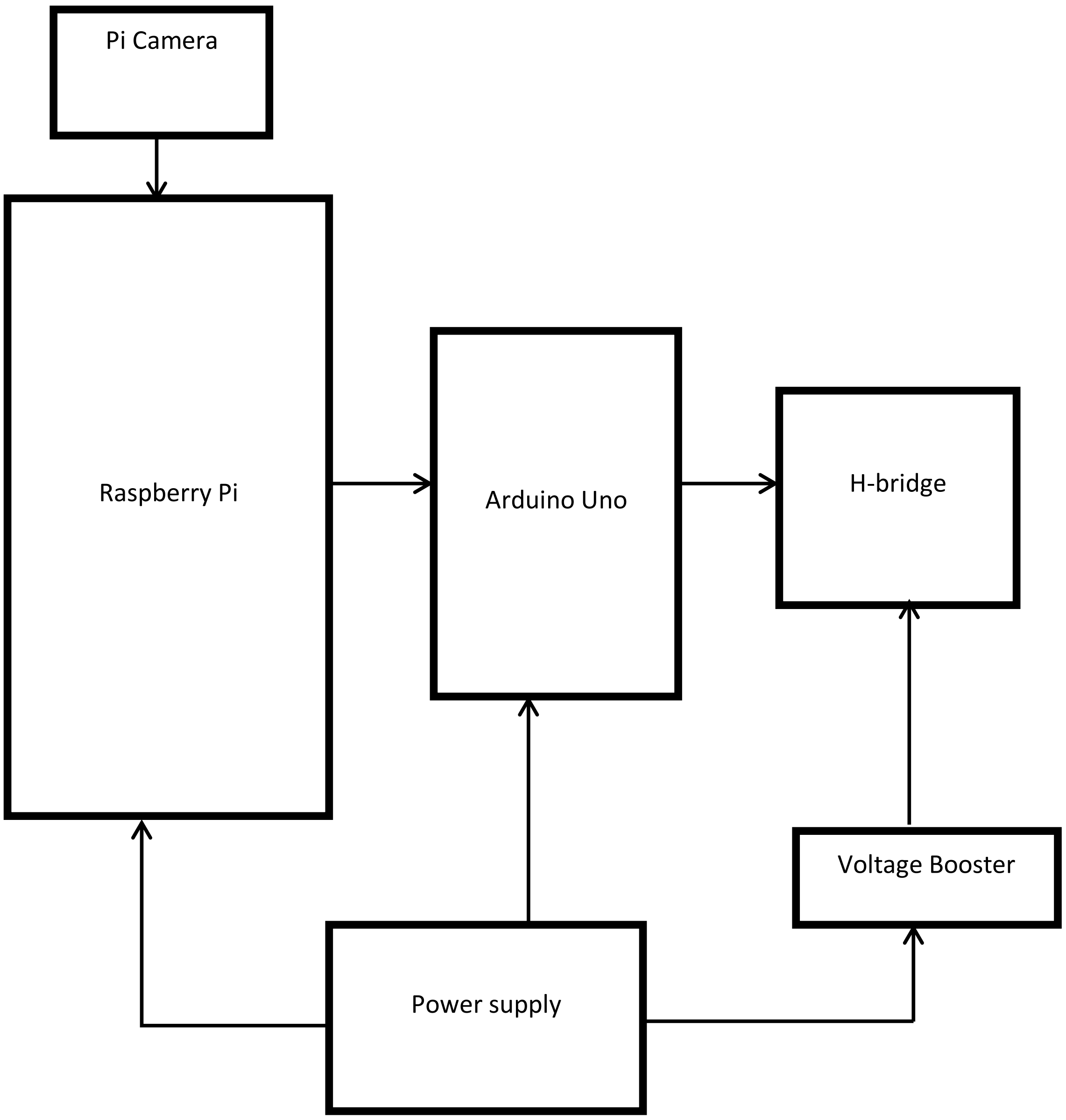


Fig 4.1 Block Diagram

4.2 MAIN BLOCK

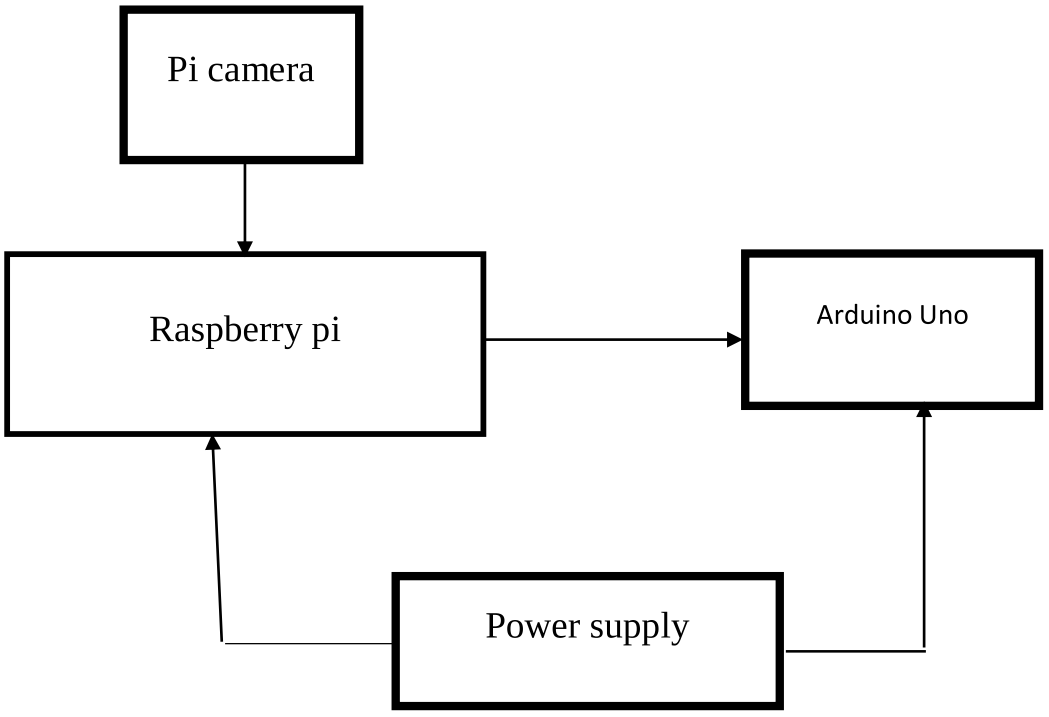


Fig 4.2 Main block diagram

4.3 IMPLEMENTATION OF HARDWARE

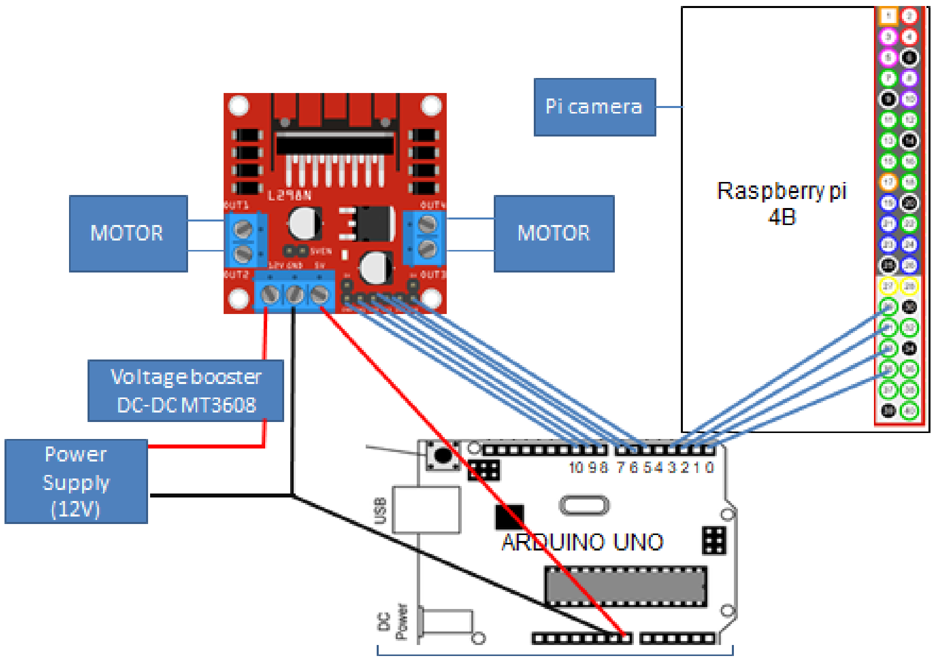


Fig 4.3 Circuit diagram of the project

WORKING

- **Motor Driver L293D**: L293D is a typical Motor driver or Motor Driver IC which allows DC motor to drive on either direction. L293D is a 16-pin IC which can control a set of two DC motors simultaneously in any direction. It means that you can control two DC motor with a single L293D IC.

- **Raspberry Pi 4**: The Raspberry pi is a small sized single-board computer. The model we are using is known as "Model b+". It consists of a Wi-Fi module and a slot to connect the external camera module. The Power is provided through the custom build PCB that we have made. It acts as a master device. It is used as the main central processor to coordinate the functions, take appropriate inputs and then the decisions accordingly.

- **Pi Camera**: It is the camera shipped along with Raspberry Pi which can be used to take high-definition videos as well as photographs.

- **Arduino UNO**: This microcontroller is used in our self-driving car which acts as a slave device. It also acts according to the input provided by the raspberry pi. It regulates the motor and adjusts its speed as per the requirements.

4.4 PROPOSED METHOD FOR STEERING CONTROL

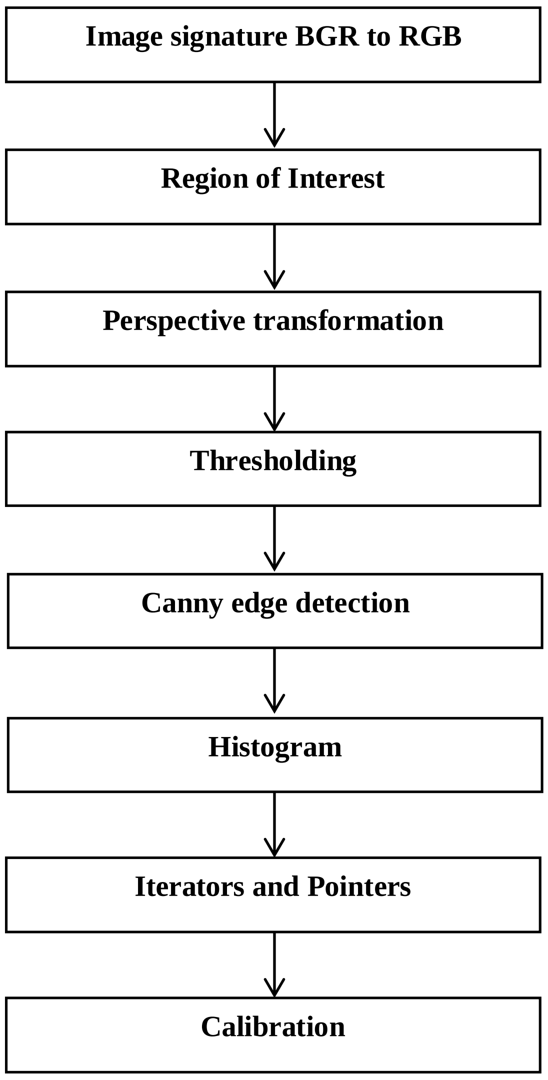


Fig 4.4 Proposed Method for Steering Control

Convert image signature: The images that were initially taken by us as input are in the format of BGR as default in the OpenCV software. We convert it into RGB format since majority of libraries support it.

Region of interest: The region of interest is created in such a way that only the bottom half part of the frame is utilized for performing image processing operations. It is performed by creating four coordinates which are measured in pixels and join the coordinates to make region we are interested.

Perspective Transformation (Birds eye view): The perspective transformation operation is performed on the image frame of region of interest. This increases the scope of extracting information from images and the image frame is visible in the form of birds eye view.

Threshold: The image from previous step is converted from RGB to gray scale. The thresholding operation is applied to the image gray scale image where it becomes a binary image that is black and white. Thresholding is performed to differentiate the road as well as lane markings.

Canny edge detection: Canny edge detection is an operation that is performed to find edges of the give image. Here this operation is applied after thresholding in order to get the edges of the lane lines of the road track.

Histogram: The histogram refers to the pixel intensity values of the image frame. Histogram is utilized to plot the pixel intensity values of the threshold of the image.

Iterators and pointers: Iterators and pointers are utilized to detect the left and right lane markings of the road track. This performed by the use data in the dynamic array that consists of intensity values.

Calibration: The calibration is performed to create lane centre and overlap it with the frame centre with the use of dynamic array of the image frame.

After the above operations are performed we use variable and a pointer which indicates the difference between lane centre and frame centre if they don’t overlap. Here if there is any offset from the lane centre and frame centre we get some magnitude value depending on the direction of offset whether right or left. Here information of the offset and magnitude values is sent to Arduino microcontroller. The Arduino then commands the motor driver to correct the movement by overlapping the lane centre and frame centre.

4.5 PROPOSED METHOD FOR OBSTACLE AVOIDANCE

This project adapted the shape-based approach and used Haar feature-based cascade classifiers for object avoidance. Since each obstacle requires its own classifier, we use only one obstacle to demonstrate the operation obstacle avoidance. We use the detector that is provided by OpenCV library. Positive samples (contains target obstacle) are collected using raspicam and were cropped so that only required obstacle is visible. Negative samples were also collected using raspicam, which contains some random images.

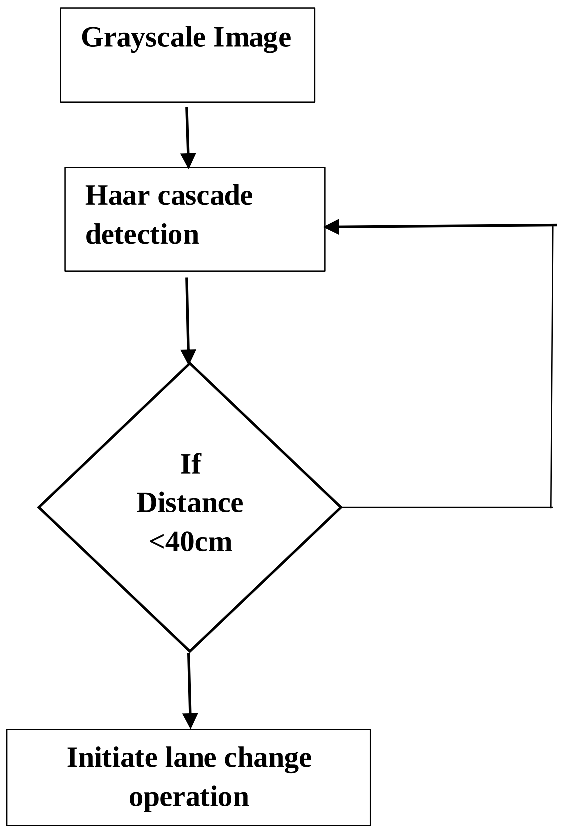


Fig 4.5 Proposed Method for Obstacle Avoidance

To recognize different states of the obstacle, some image processing is needed beyond detection. Flowchart in Figure 4.5 summarizes the Obstacle recognition process. Firstly, trained cascade classifier is used to detect Obstacle. The bounding box is considered as a region of interest (ROI). Secondly, The distance is measured with help of depth of image in camera, then the distance (in cms) is also displayed on the obstacle window. Thirdly, If the distance between the camera and obstacle is lesser than 40 cm, the lane changing function is initiated.

4.5.1 Cascade Training

The next step is the training of classifier. There are several ways to train a cascade classifier some of the methods include opencv train cascade, opencv_haar training, Cascade Trainer GUI, etc. We will be using Cascade Trainer GUI, which is one of the easiest ways to train a cascade. Cascade Trainer GUI is a program that can be used to train, test and improve cascade classifier models. It uses a graphical interface to set the parameters and make it easy to use OpenCV tools for training and testing classifiers. After the Cascade trainer GUI application has finished its work, the trained cascade will be saved in <file_name>.xml file in the folder, which was passed as -data parameter. Other files in this folder are created for the case of interrupted training, so you may delete them after completion of training. The flowchart of Haar Transform is as shown in figure 4.5.1

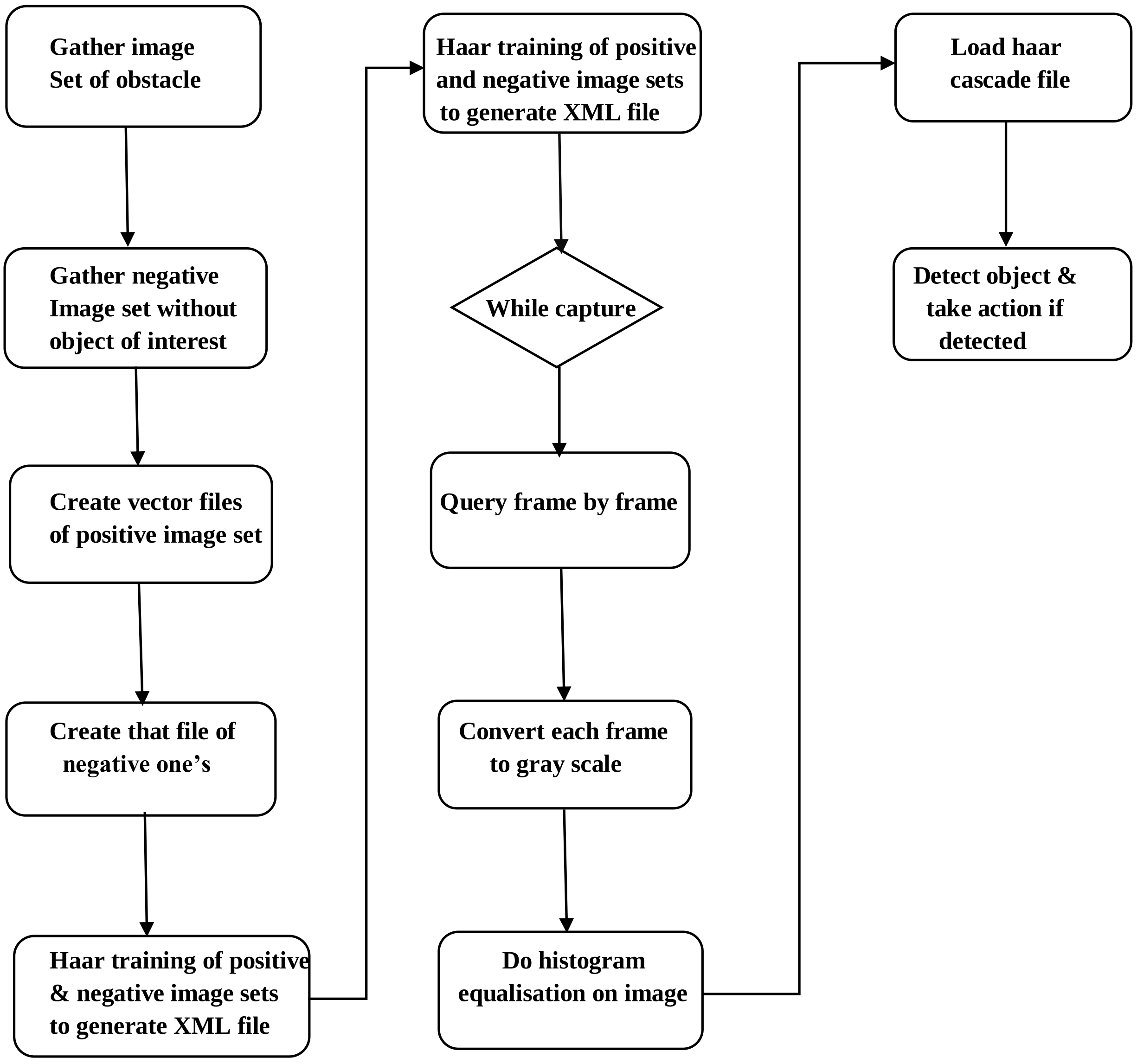

Gather image Set of obstacle
Gather negative Image set without object of interest
Create vector files of positive image set
Create that file of negative one's
Haar training of positive & negative image sets to generate XML file
Haar training of positive and negative image sets to generate XML file
While capture
Query frame by frame
Convert each frame to gray scale
Do histogram equalisation on image
Load haar cascade file
Detect object & take action if detected

Fig 4.5.1 Flowchart of Haar Transform

Chapter 5

HARDWARE AND SOFTWARE REQUIREMENTS

5.1 Hardware Requirements

The hardware components required for this project are

1. Raspberry Pi
2. L298 H-Bridge
3. Pi Camera
4. Arduino Uno
5. DC Motor
6. Voltage Booster
7. Power Bank
8. Rechargeable Batteries

5.2 Software Requirements

The software requirements for this project are

1. C++ and OpenCV Libraries
2. Arduino uno (IDE)
3. Raspbian OS

5.1 Hardware Description

5.1.1 Raspberry Pi:

A Raspberry Pi is a credit card sized computer originally designed for education, inspired by the 1981 BBC Micro. Creator Eben Upton's goal was to create a low-cost devise that would improve program and skills and hardware understanding at the pre-university level. Raspbian is the official operating system of the Raspberry Pi foundation. The Pi also supports wireless internet with built in Wi-Fi and Bluetooth.

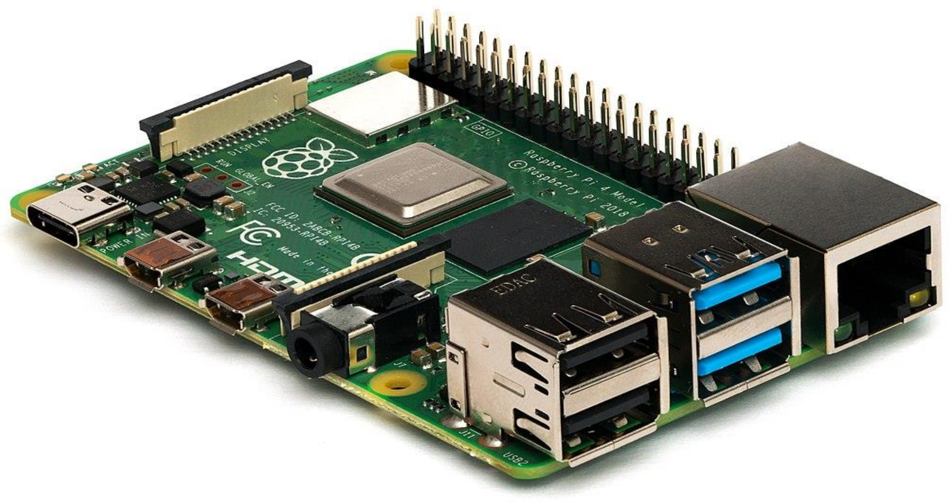


Fig 5.1.1 Raspberry Pi 4 Model B

Raspberry Pi 4 Model B with a 1.5 GHz 64-bit quad core ARM Cortex-72 processor, on-board 802.11ac Wi-Fi, Bluetooth 5, full gigabit Ethernet (throughput not limited), two USB 2.0 ports, two USB 3.0 ports, and dual-monitor support via a pair of micro HDMI(HDMI Type D) ports for up to 4K resolution. The Pi 4 is also powered via a USB-C port, enabling additional power to be provided to downstream peripherals, when used with an appropriate PSU

5.1.2 L298 H-Bridge:

An H bridge is an electronic circuit that switches the polarity of a voltage applied to load. These circuits are often used in robotics and other applications to allow DC motors to run forwards or backwards. The H-bridge arrangement is generally used to reverse the polarity/direction of the motor, but can also be used to 'brake' the motor, where the motor comes to a sudden stop, as the motor's terminals are shorted, or to let the motor 'free run' to a stop, as the motor is effectively disconnected from the circuit.

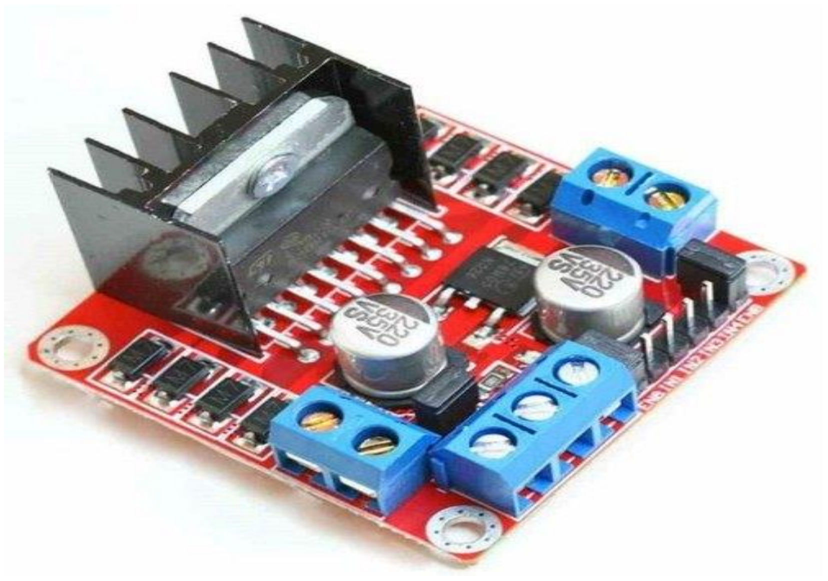


Fig 5.1.2 L298 H-Bridge

5.1.3 Pi Camera:

The Pi camera module is a portable light weight camera that supports Raspberry Pi. It communicates with Pi using the MIPI camera serial interface protocol. It is normally used in image processing, machine learning or in surveillance projects. It is commonly used in surveillance drones since the payload of camera is very less. Apart from these modules Pi can also use normal USB webcams that are used along with computer.

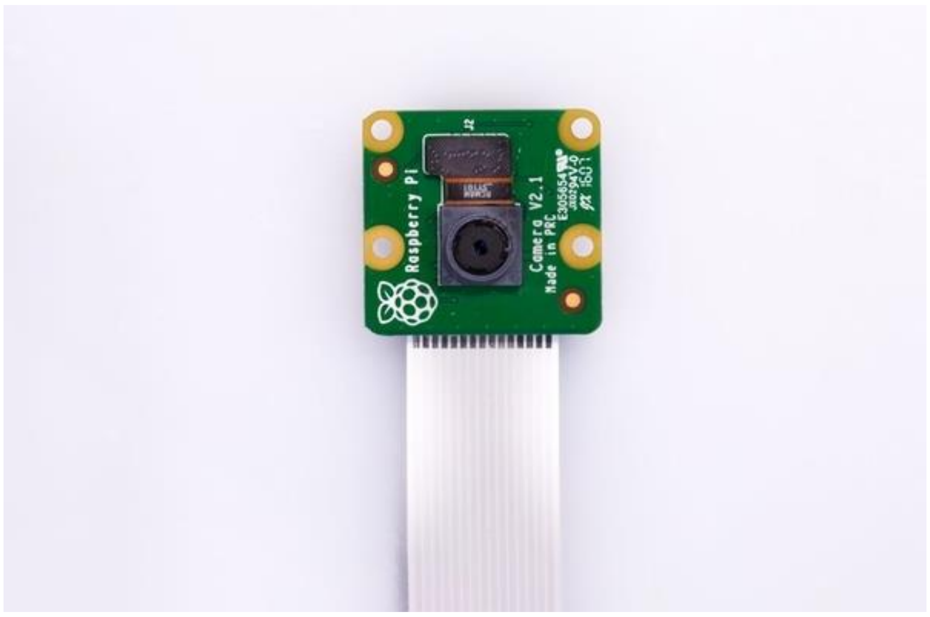


Fig 5.1.3 Pi Camera

5.1.4 Arduino Uno:

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started.

Fig 5.1.4 Arduino UNO

5.1.5 DC Motor:

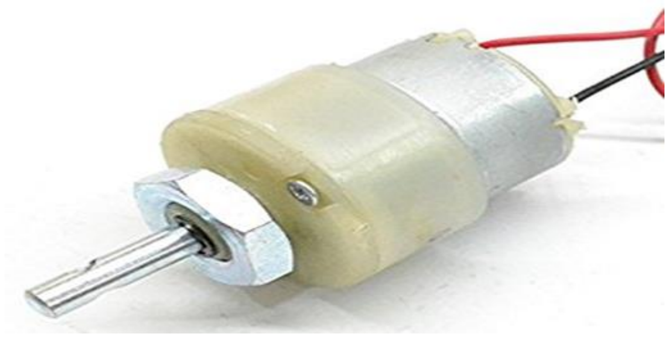


Fig 5.1.5 DC Motor

A DC motor relies on the fact that like magnet poles repel and unlike magnetic poles attract each other. It relies on the fact that like magnet poles repel and unlike magnetic poles attract each other. A coil of wire with a current running through it generates an electromagnetic field aligned with the centre of the coil. By switching the current on or off in a coil its magnetic field can be switched on or off or by switching the direction of the current in the coil the direction of the generated magnetic field can be switched 180°. A simple DC motor typically has a stationary set of magnets in the stator and an armature with a series of two or more windings of wire wrapped in insulated stack slots around iron pole pieces (called stack teeth) with the ends of the wires terminating on a commutator. The speed of a DC motor can be controlled by changing the voltage applied to the armature. The introduction of variable resistance in the armature circuit or field circuit allowed speed control. Modern DC motors are often controlled by power electronics systems which adjust the voltage by "chopping" the DC current into on and off cycles which have an effective lower voltage. DC motors can operate directly from rechargeable batteries, providing the motive power for the first electric vehicles and today's hybrid cars and electric cars as well as driving a host of cordless tools. DC motors are still found in applications as small as toys and disk drives, or in large sizes to operate steel rolling mills and paper machines.

5.1.6 VOLTAGE BOOSTER XL6009:

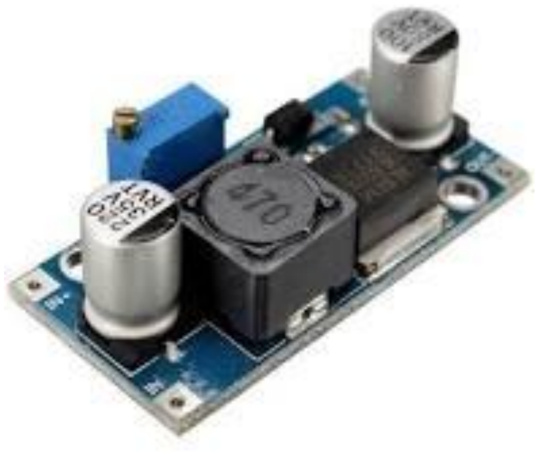


Fig 5.1.6 Voltage Booster XL6009

This module is a non-isolated step-up (boost) voltage converter featuring adjustable output voltage and high efficiency. Description: XTW6009 is a 4 a switch current high-performance step-up (BOOST) module. The XL6009E1 module uses the second generation of high frequency switch technology as the core chip, LM2577 performance far beyond the first generation of technology. XL6009 booster module cost is lower, performance is more outstanding, LM2577 module will be eliminated. If the input or output current is more than 2A (need to plug heat sink), if the output power is higher than 15 W, advise to add the heat sink. Please do not use without heat sink for continuous use. The output voltage must be higher than the input voltage. IN-IN + input positive input negative, positive output OUT + OUT-output negative.

5.2 Software Description

5.2.1 Installation of Operating System on Raspberry Pi

Raspberry Pi is a small computer; hence operating system (OS) should be installed. As the Raspberry doesn’t have hard drive, OS is installed in the external memory. For that, memory card (SD card) is used for the installation of operating system and all the required software and supporting files are stored in the same SD card. The proposed work implement Simulink support package on Raspberry Pi 4 model B. This package support enables development software for algorithms which can run in Raspberry Pi. Moreover, it allows controlling peripheral devices that connected via its GPIO interfaces such as serial, I2C and SPI by using command functions. Simulink support package allows users to create Simulink models in various fields such as audio, image or embedded system using general and support package toolboxes. Furthermore, Raspberry Pi can be used to create applications in standalone mood by deploying Simulink model onto it.

The proposed lane detection algorithm:

The aim of the image processing is to obtain a processed image for each original captured image; this processed image can give information about the position of the vehicle such as its orientation and the lateral distance with respect to the center line of the road lane at a particular look-ahead distance.

There are two main stages are implemented the first stage is the pre-processing stage and then the second stage is the lane detection stage. The main aim of pre-processing stage is to get a gray image and remove its noise. The major aim of lane detection is to detect the lane lines in front of the vehicle to obtain the lateral offset distance and the lane lines angles with respect to the vehicle.

5.2.2 Open CV

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garagethen. The library is cross platform and free for use under the open-source BSD license. OpenCV supports deep learning frameworks like TensorFlow, Torch/PyTorch.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers.

In 1999, the OpenCV project was initially an Intel Research initiative to advance CPU intensive applications, part of a series of projects including real-time ray tracing and 3D display walls. The main contributors to the project included a number of optimization experts in Intel Russia, as well as Intel‛s Performance Library Team.

OpenCV has a modular structure, which means that the package includes several shared or static libraries. The following modules are available:

- Core functionality (core) - a compact module defining basic data structures, including the dense multi-dimensional array Mat and basic functions used by all other modules.
- Image Processing (imgproc) - an image processing module that includes linear and non-linear image filtering, geometrical image transformations (resize, affine and perspective warping, generic table-based remapping), color space conversion, histograms, and so on.
- Video Analysis (video) - a video analysis module that includes motion estimation, background subtraction, and object tracking algorithms.
- Camera Calibration and 3D Reconstruction (calib3d) - basic multiple-view geometry algorithms, single and stereo camera calibration, and object pose estimation, stereo correspondence algorithms, and elements of 3D reconstruction.
- 2D Features Framework (features2d) - salient feature detectors, descriptors, and descriptor matchers.
- Object Detection (objdetect) - detection of objects and instances of the predefined classes (for example, faces, eyes, mugs, people, cars, and so on).
- High-level GUI (highgui) - an easy-to-use interface to simple UI capabilities.
- Video I/O (videoio) - an easy-to-use interface to video capturing and video codecs.

Structure of Open CV

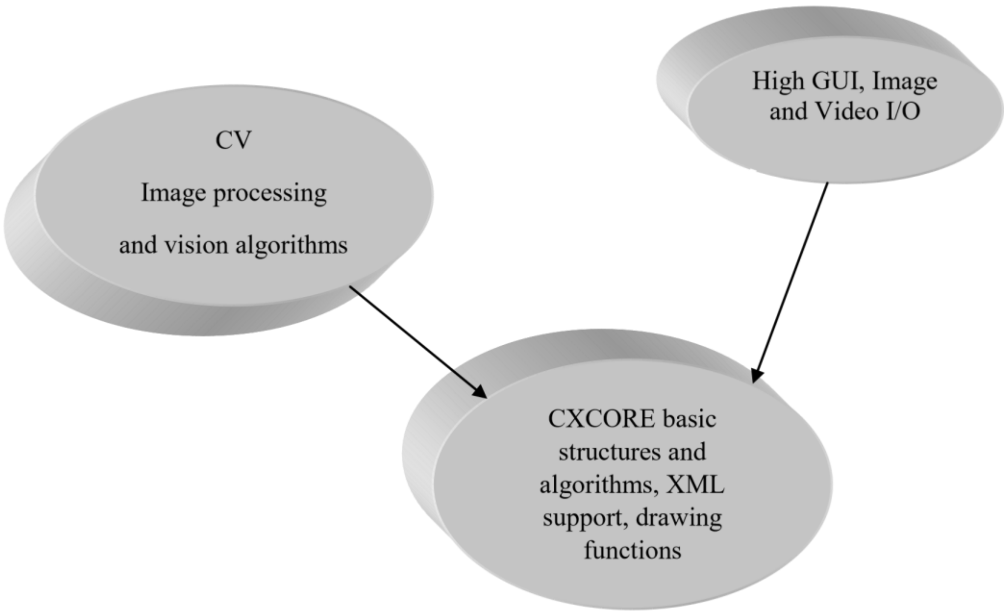

CV
Image processing
and vision algorithms
High GUI, Image and Video I/O
CXCORE basic structures and algorithms, XML support, drawing functions

Fig 5.2.1 Structure of Open CV

Once OpenCV is installed, the OPENCV_BUILD\install directory will be populated with three types of files:

- Header files: These are located in the OPENCV_BUILD\install\include subdirectory and are used to develop new projects with OpenCV.
- Library binaries: These are static or dynamic libraries (depending on the option selected with CMake) with the functionality of each of the OpenCV modules. They are located in the bin subdirectory (for example, x64\mingw\bin when the GNU compiler is used).
- Sample binaries: These are executables with examples that use the libraries. The sources for these samples can be found in the source package.

General description

- Open source computer vision library in C/C++.
- Optimized and intended for real-time applications.
- OS/hardware/window-manager independent.

- Generic image/video loading, saving, and acquisition.
- Both low and high level API.

Provides interface to Intel's Integrated Performance Primitives (IPP) with processor specific optimization (Intel processors).

Features

- Image data manipulation (allocation, release, copying, setting, conversion).
- Image and video I/O (file and camera based input, image/video file output).
- Matrix and vector manipulation and linear algebra routines (products, solvers, SVD).
- Various dynamic data structures (lists, queues, sets, trees, graphs).
- Basic image processing (filtering, edge detection, corner detection, sampling and interpolation, color conversion, morphological operations, histograms, image pyramids).
- Structural analysis (connected components, contour processing, distance transform, various moments, template matching, Hough transform, polygonal approximation, line fitting, ellipse fitting, and Delaunay triangulation).
- Camera calibration (finding and tracking calibration patterns, calibration, fundamental matrix estimation, homography estimation, stereo correspondence).

- Motion analysis (optical flow, motion segmentation, tracking).
- Object recognition (eigen-methods, HMM).
- Basic GUI (display image/video, keyboard and mouse handling, scroll-bars).
- Image labeling (line, conic, polygon, text drawing)

OpenCV working with video capturing

OpenCV supports capturing images from a camera or a video file (AVI).

- **Initializing capture from a camera:**

CvCapture* capture = cvCaptureFromCAM(0); // capture from video device #0

- **Initializing capture from a file:**

  CvCapture* capture = cvCaptureFromAVI("infile.avi");

  Capturing a frame: IplImage* img = 0;

  if(!cvGrabFrame(capture)){   // capture a frame

  printf("Could not grab a frame\n\7"); exit(0); }

  img=cvRetrieveFrame(capture); // retrieve the captured

  frame

  To obtain images from several cameras simultaneously, first grab an image from each camera. Retrieve the captured images after the grabbing is complete.

- **Releasing the capture source:** cvReleaseCapture(&capture).

5.2.3 Arduino IDE

Arduino Coding Environment and basic tool

Arduino code is written in C++ with an addition of special methods and functions, which we'll mention later on. C++ is a human-readable programming language. When you create a 'sketch' (the name given to Arduino code files), it is processed and compiled to machine language.

The Arduino Integrated Development Environment (IDE) is the main text editing program used for Arduino programming. It is where you'll be typing up your code before uploading it to the board you want to program. Arduino code is referred to as sketches

.

Debugging Arduino Code and Hardware

Unlike other software programming platforms, Arduino doesn't have an onboard debugger. Users can either use third-party software, or they can utilize the serial monitor to print Arduino's active processes for monitoring and debugging.

By using the Serial class, you can print to the serial monitor, debugging comments and values of variables. On most Arduino models, this will be using serial pins 0 and 1 which are connected to the USB port.

Libraries

In Arduino, much like other leading programming platforms, there are built-in libraries that provide basic functionality. In addition, it's possible to import other libraries and expand the Arduino board capabilities and features. These libraries are roughly divided into libraries that interact with a specific component or those that implement new functions.

Pin Definitions

To use the Arduino pins, you need to define which pin is being used and its functionality. A convenient way to define the used pins is by using:

'#define pinName pinNumber'.

The functionality is either input or output and is defined by using the pinMode () method in the setup section.

Declarations

1. Variables

Whenever you're using Arduino, you need to declare global variables and instances to be used later on. In a nutshell, a variable allows you to name and store a value to be used in the future. For example, you would store data acquired from a sensor in order to use it later. To declare a variable you simply define its type, name and initial value.

It's worth mentioning that declaring global variables isn't an absolute necessity. However, it's advisable that you declare your variables to make it easy to utilize your values further down the line.

2. Instances

In software programming, a class is a collection of functions and variables that are kept together in one place. Each class has a special function known as a constructor, which is used to create an instance of the class. In order to use the functions of the class, we need to declare an instance for it.

3. Setup()

Every Arduino sketch must have a setup function. This function defines the initial state of the Arduino upon boot and runs only once.

Here we'll define the following:

Pin functionality using the pinMode function
Initial state of pins
Initialize classes
Initialize variables
Code logic

4. Loop()

The loop function is also a must for every Arduino sketch and executes once setup() is complete. It is the main function and as its name hints, it runs in a loop over and over again. The loop describes the main logic of your circuit.

How to program Arduino

The basic Arduino code logic is an "if-then" structure and can be divided into 4 blocks:

1. Setup - will usually be written in the setup section of the Arduino code, and performs things that need to be done only once, such as sensor calibration.

2. Input - at the beginning of the loop, read the inputs. These values will be used as conditions ("if") such as the ambient light reading from an LDR using analogRead().

3. Manipulate Data - this section is used to transform the data into a more convenient form or perform calculations. For instance, the AnalogRead() gives a reading of 0-1023 which can be mapped to a range of 0-255 to be used for PWM.(see analogWrite())

4.Output - this section defines the final outcome of the logic ("then") according to the data calculated in the previous step. Looking at our example of the LDR and PWM, turn on an LED only when the ambient light level goes below a certain threshold.

Arduino Code libraries

Library Structure

A library is a folder comprised of files with C++ (.cpp) code files and C++ (.h) header files.

The .h file describes the structure of the library and declares all its variables and functions.

The .cpp file holds the function implementation.

Importing Libraries

The first thing you need to do is find the library you want to use out of the many libraries available online. After downloading it to your computer, you just need to open Arduino IDE and click on Sketch > Include Library > Manage Libraries. You can then select the library that you want to import into the IDE. Once the process is complete the library will be available in the sketch menu.

In the code provided by circuito.io instead of adding external libraries like mentioned before, we provide them with the firmware folder. In this case, the IDE knows how to find them when using #include.

From Software to Hardware

There is a lot to be said of Arduino’s software capabilities, but it’s important to remember that the platform is comprised of both software and hardware. The two work in tandem to run a complex operating system.

Code → Compile → Upload → Run

At the core of Arduino, is the ability to compile and run the code.

After writing the code in the IDE you need to upload it to the Arduino. Clicking the Upload button (the right-facing arrow icon), will compile the code and upload it if it passed compilation. Once your upload is complete, the program will start running automatically

Chapter 6

TESTING

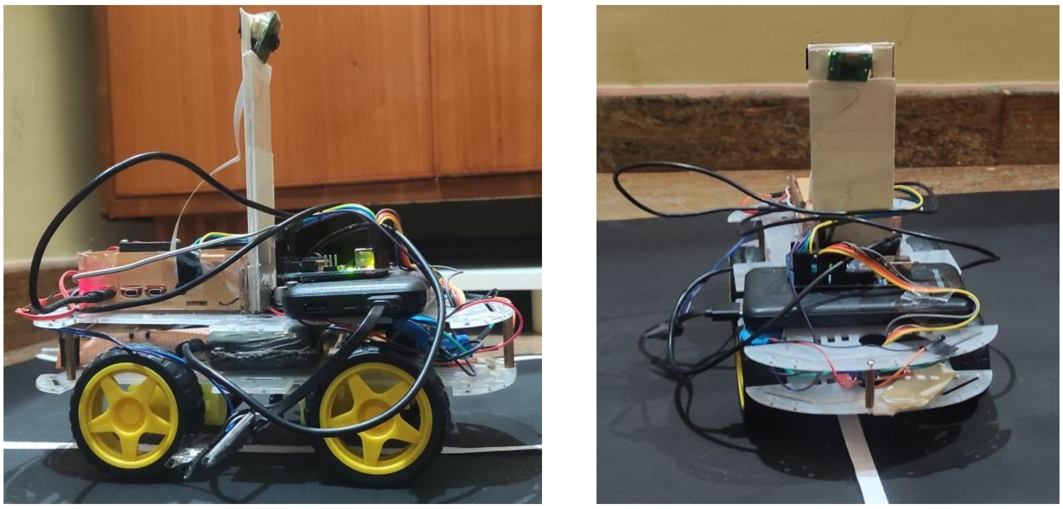
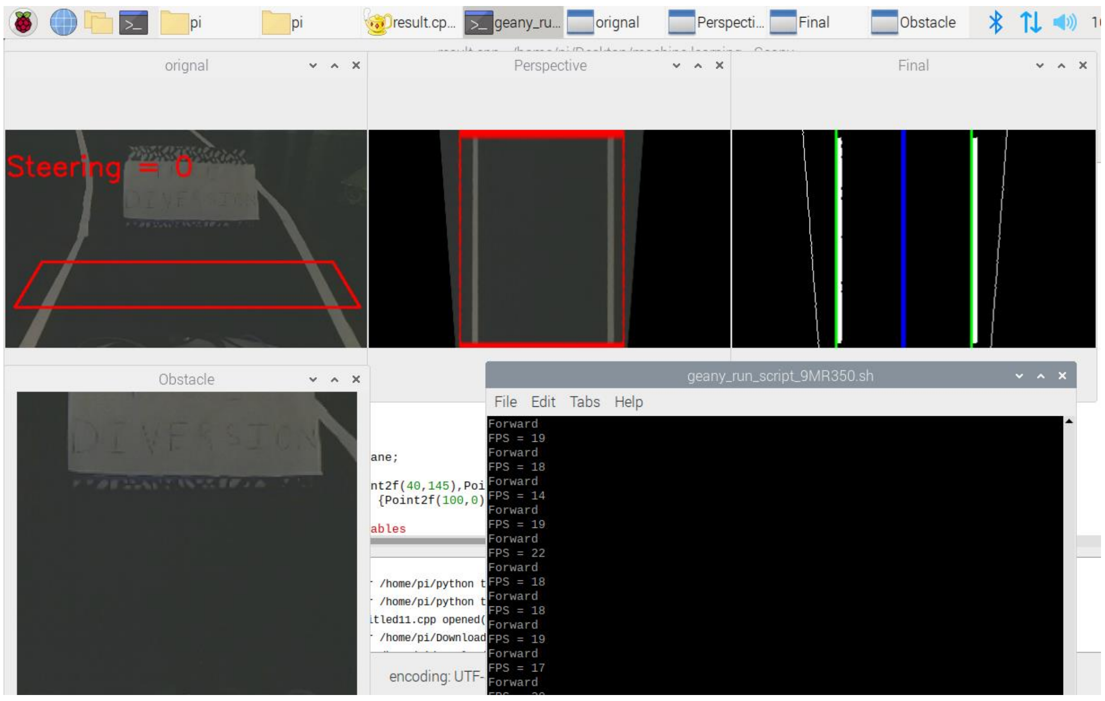


Fig 6.1: Complete model of Autonomous Car

6.1 Testing for steering control

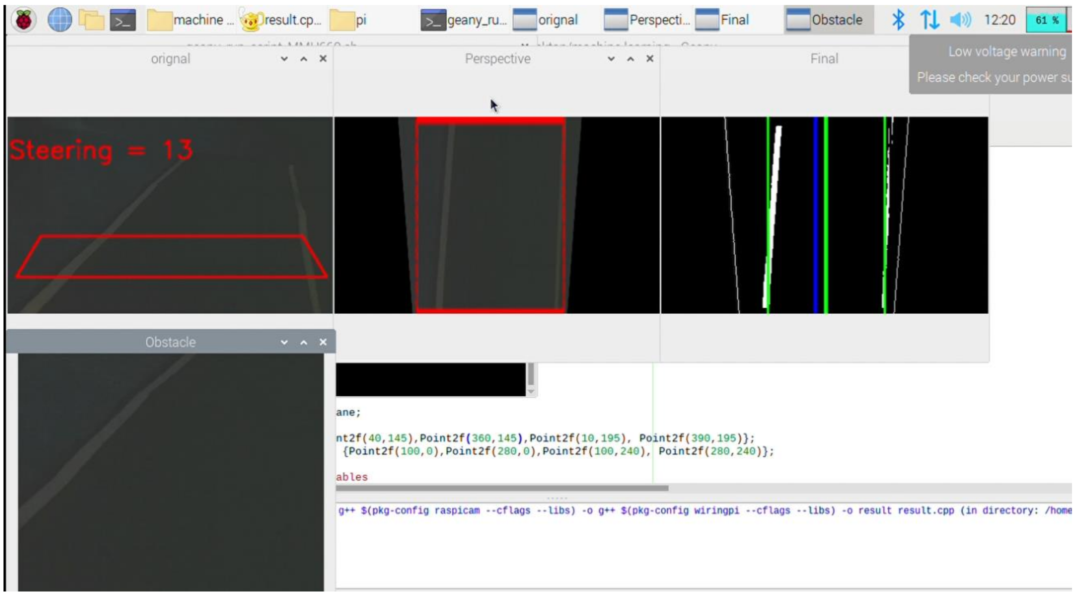
The steer control system is designed to automatically steer the autonomous vehicle and detect the change in the trajectory with respect to lane markings. The deviation of vehicle with respect to lane markings i.e., if there is no overlap between lane centre and frame centre it gives values with magnitude. Whenever the model is placed in the centre of the road track the steering value will be zero. Whenever there is deviation of the prototype towards the right lane line the negative varying values will be displayed according to the offset. If the offset is towards the left lane lines the varying values will be positive. Based on the values and their magnitude the data is sent to the Arduino Uno for correction of the prototype autonomous vehicles trajectory.



```
ane;

nt2f(40,145),Poi
 {Point2f(100,0)

ables

 /home/pi/python t
 /home/pi/python t
itled11.cpp opened(
 /home/pi/Download
```

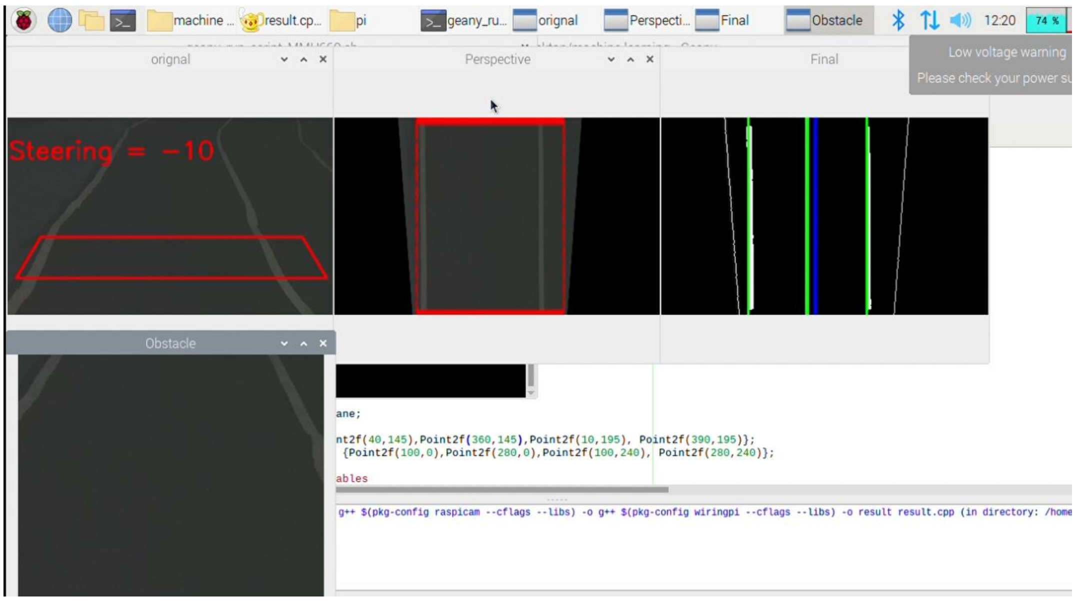
```
Forward
FPS = 19
Forward
FPS = 18
Forward
FPS = 14
Forward
FPS = 19
Forward
FPS = 22
Forward
FPS = 18
Forward
FPS = 18
Forward
FPS = 19
Forward
FPS = 17
Forward
```

Fig 6.2: Steering value when model placed at centre of track



```
ane;

nt2f(40,145),Point2f(360,145),Point2f(10,195), Point2f(390,195)};
 {Point2f(100,0),Point2f(280,0),Point2f(100,240), Point2f(280,240)};

ables

g++ $(pkg-config raspicam --cflags --libs) -o g++ $(pkg-config wiringpi --cflags --libs) -o result result.cpp (in directory: /home
```

Fig 6.3: steering value due to deviation of model towards left lane line.



```
ane;

nt2f(40,145),Point2f(360,145),Point2f(10,195), Point2f(390,195)};
 {Point2f(100,0),Point2f(280,0),Point2f(100,240), Point2f(280,240)};

ables
```

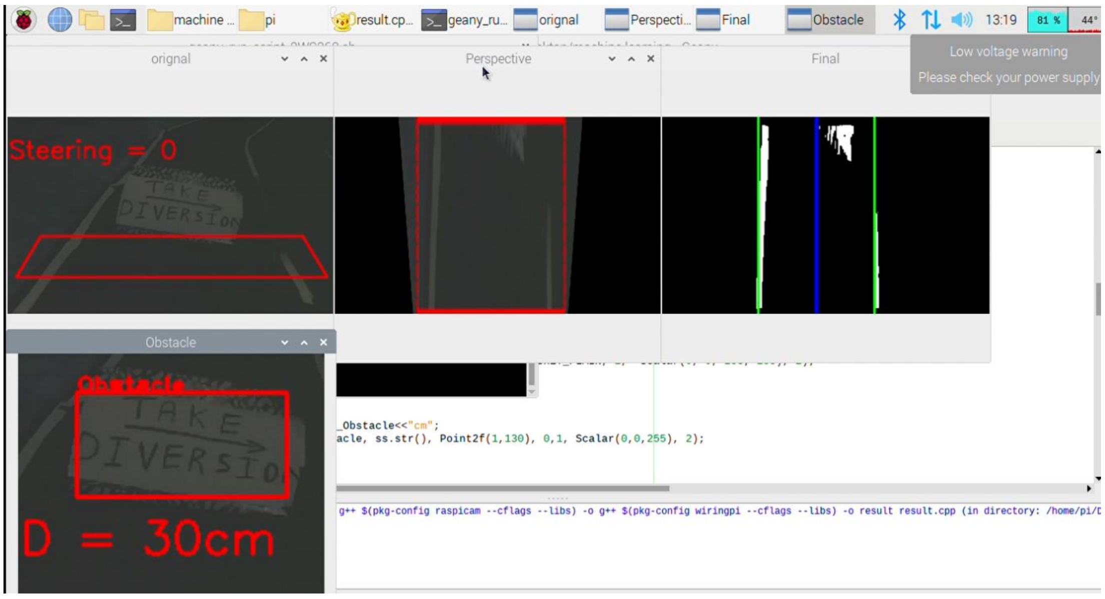
```
g++ $(pkg-config raspicam --cflags --libs) -o g++ $(pkg-config wiringpi --cflags --libs) -o result result.cpp (in directory: /home
```

Fig 6.4: Steering value due to deviation of model towards right lane line.

The figures above shows the results of the autonomous steering system at various deviations. The variable values with their magnitude indicate the amount of deviation of model towards right or left lane lines.

6.2 Testing for obstacle detection

The obstacle present on the track of the road is detected with the computer vision i.e., with the pi-camera mounted. The figure below shows the obstacle detected and the amount of distance between the obstacle and the prototype vehicle.



```
_Obstacle<<"cm";
acle, ss.str(), Point2f(1,130), 0,1, Scalar(0,0,255), 2);
```

```
g++ $(pkg-config raspicam --cflags --libs) -o g++ $(pkg-config wiringpi --cflags --libs) -o result result.cpp (in directory: /home/pi/D
```

Fig 6.5: Obstacle detection with distance from camera

Chapter 7

CONCLUSION AND FUTURE WORK

Driverless Car revolution which aims at the development of autonomous vehicles for easy transportation without a driver. For the Economy, Society and individual business this autonomous technology has brought many broad implications.

Our aim was to design and an autonomous vehicle for the detection of obstacles, lane detection and steering angle, and hence we developed an autonomous vehicle with minimum hardware which is reliable and cost effective. The implemented project simplifies driving on road. The implemented system can continuously acquire the information about obstacles. The data acquired from the Pi Camera is given to Raspberry Pi for further processing. Pi camera were used to detect the obstacles. The Raspberry Pi was used for processing and an Arduino UNO microcontroller for running the motors.

The Raspberry Pi acquires the data from Pi camera which monitors the road for obstacles. The actions are priority based. The priority which is considered is explained in the testing section. Once obstacle is detected the car stops the motion immediately and overtakes the obstacle. The algorithm mentioned in the paper has been successfully implemented on a small autonomous car.

7.1 Advantages of Autonomous Cars

1. **Decreased the number of accidents:**

   Autonomous cars prevent human errors from happening as the system controls the vehicle. It leaves no opportunity for distraction, not just like humans who are prone to interruptions. It also uses complicated algorithms that determine the correct stopping distance from one vehicle to another. Thereby, lessening the chances of accidents dramatically.

2. Lessens traffic jams:

Driverless cars in a group participate in platooning. This allows the vehicles to brake or accelerates simultaneously. Platoon system allows automated highway system which may significantly reduce congestion and improve traffic by increasing up the lane capacity. Autonomous cars communicate well with one another. They help in identifying traffic problems early on. It detects road fixing and detours instantly. It also picks up hand signals from the motorists and reacts to it accordingly.

3. Time-saving vehicle:

As the system takes over the control, the driver has a spare time to continue work or spend this time catching up with their loved-ones without the having the fear about road safety.

4. Accessibility to transportation:

Senior citizens and disabled personnel are having difficulty driving. Autonomous vehicles assist them towards safe and accessible transportation.

7.2 Disadvantages

1. Expensive:

High-technology vehicles and equipment are expensive. They prepare a large amount of money for research and development as well as in choosing the finest and most functional materials needed such as the software, modified vehicle parts, and sensors. Thus, the cost of having Autonomous cars is initially higher. However, this may lower down after 10 years giving way for the average earner people to have one.

2. Safety and security concerns:

Though it has been successfully programmed, there will still be the possible unexpected glitch that may happen. Technologies are continuously updating and almost all of this equipment may have a faulty code when the update was not properly and successfully done.

3. Prone to Hacking:

Autonomous vehicles could be the next major target of the hackers as this vehicle continuously tracks and monitors details of the owner. This may lead to the possible collection of personal data.

4. Fewer job opportunities for others:

As the artificial intelligence continues to overcome the roles and responsibilities of humans, taxi, trucks, or even co-pilots may be laid off as their services will no longer be needed. This may significantly impact the employment rate and economic growth of a certain country

7.3 Future Work

To enhance the performance and ensure practically of the car, the efficiency and processor speed need to be raised. A camera of better resolution would also be required as the scenes keep changing in the real world. We trained the car to detect the obstacle that is stagnant future work may be to train the obstacle that is moving vehicle.

Chapter 8

REFERENCES

[1] Dr. Ashoka S B, Lakshmikanth D, "Implementation and Demonstration of Obstacle Detection in Self-Driving Cars with GPS Tracking in Real-Time System", International Journal of Science and Research (IJSR).

[2] Abdulhakam. A M .Assidiq, Othman O. Khalifa, Md. Rafiqul Islam, Sheroz Khan, "Real Time Lane Detection for Autonomous Vehicles", " IATSS September 2014.

[3] Sumeyya ilkin, Selin hangisi, Mervetafrali, Suhapsahin, "The Enhancement of Canny Edge Detection algorithm using Prewitt Robert and Sobel Kernels", International Journal of Applied Engineering Research ISSN 2019.

[4] M.L. Ho, P.T. Chan, A.B. Rad, "Lane Change Algorithm for Autonomous Vehicles via Virtual Curvature Method", Journal of Advanced Transportation, Vol. 43, No. 1.

[5] M. Venkatesh, P. Vijayakumar, "Transformation Technique" International Journal of Scientific & Engineering Research, Volume 3, Issue 5, May-2012 (ISSN).

[6] Thathupara Subramanyan Kavya, Young-Min Jang, Tao Peng, Sang-Bock Cho, "Vehicle Detection Tracking from a Video captured by Moving Host", International Research Journal of Engineering and Technology (IRJET) Volume: 04 Jan -2017.

[7] Balika J. Chelliah, Vishal Chauhan, Shivendra Mishra, Vivek Sharma, "Advancement of Driverless Cars and Heavy Vehicles using Artificial Intelligence (Object Detection)", International Journal of Engineering and Advanced Technology (IJEAT) Volume-9 Issue-1, October 2019.

[8] Hironobu Fujiyoshi, Tsubasa Hirakawa, Takayoshi Yamashita, "Deep learning based image recognition for autonomous driving" IATSS September 2016.

[9] Li Cuimei, Qi Zhiliang, Jia Nan, Wu Jianhua,"Human Face Detection Algorithm via Haar Cascade Combined with Three Additional Classifiers",International Journal of Scientific & Engineering Research, Volume 3, Issue 5, May-2012 (ISSN).

[10] Sergey Sergeevich Shadrin, Andrey Mikhailovich Ivanov, "Algorithm for Autonomous Vehicles Steering System Control Law Estimation while the Desired Trajectory Driving", International Journal of Engineering Research & Technology (IJERT) 2019.

[11] Ms. Sujeetha, Chitrak Bari, Gareja Prdip, Siddhant Purohit, "THE YAN: [Self Driving Car Using Deep Learning]", International Journal of Applied Engineering Research ISSN 2019.

[12] Shota Nakashima, Shingo Aramaki , Yuhki Kitazono , Shenglin Mu , Kanya Tanaka and Seiichi Serikawa, "Application of Ultrasonic Sensors in Road Surface Condition Distinction Methods", International Journal of Engineering and Advanced Technology (IJEAT) Volume-8 Issue-1, October 2018.

[13] Rahul P Kharapakr, Abhishek S Khandare, Zeeshaan W Siddiqui "IoT based self-driving cars", International Journal of Engineering Research & Technology (IJERT) 2019.

[14] Tao Peng, Sang Bock Cho, "Robust lane marking detection and road geometry computation" International Journal of Scientific & Engineering Research, Volume 3, Issue 5, May-2012 (ISSN).

[15] Malik Haris and Jin Hou, "Obstacle Detection and Safely Navigate the Autonomous Vehicle from Unexpected Obstacles on the Driving Lane", IATSS 21 August 2020.