2023 International Conference on Advances in Electronics, Communication, Computing and Intelligent Information Systems (ICAECIS) | 979-8-3503-4805-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICAECIS58353.2023.10170382

Implementation of Driverless Car

Alisha Fatima
School of Electronics and Communication
REVA University
Bengaluru, India
alishafatima2307@gmail.com

Ankitha Kavya Gowda
School of Electronics and Communication
REVA University
Bengaluru, India
ankitha6661@gmail.com

Bengaluru, India
27tauseef@gmail.com

Chandana C U
School of Electronics and Communication
REVA University
Bengaluru, India
chandanacu12716@gmail.com

Prof. MD. Tauseef
School of Electronics and Communication
REVA University

Bhavitha Y
School of Electronics and Communication
REVA University
Bengaluru, India
bhavithay.p@gmail.com

***Abstract*— In recent times, there has been a drastic increase in the number of road accidents and their severity. These accidents mostly happen due to human error. Some common human errors like distracted driving, drowsy driving, driving at high speeds and delayed reaction times of the drivers result in fatal accidents that cause loss of life and damage to property. These factors can be eliminated by means of self-driving cars. A driverless car (or self-driving car or autonomous car) is a vehicle that can travel from one point to another by self-navigating without any human intervention by using sensors, cameras, radar, GPS, and AI (Artificial Intelligence). We will be implementing a Driverless Car using Deep Learning, OpenCV, DonkeyCar, TensorFlow and C++/Python.**

Keywords—DonkeyCar, TensorFlow, Keras, Python3, CNN (Convolutional Neural Network), OpenCV

I. INTRODUCTION

There has been a large spike in the number of automobile accidents these days. According to the U.S. Department of Transportation (USDOT), it has been estimated that about 20,160 people have died in road accidents in the first six months of 2021 in the U.S. alone. In India, data from the National Crime Records Bureau (NCRB) shows that in 2020, an average of 37 people died per 100 car crashes which results in a total of about 1.3 lakh deaths. Also, we need to note that these numbers include the period of lockdown due to the Covid-19 pandemic, during which there was minimal traffic movement.The major causes of these catastrophic accidents include distracted driving, drowsy and drunk driving. These factors lead to increased human error, which in turn increases the risk of having an accident. We can overcome the dangerous consequences of human error using autonomous vehicles or driverless cars. Most leading automobile manufacturers across the globe have started research and development of semi-autonomous and fully autonomous cars.

The Society of Automotive Engineers (SAE) has outlined six levels of driving automation which range from level 0 (completely manual) to level 5 (completely autonomous).

Fig 1: Levels of Automation as specified by the SAE

A. Problem statement

In recent times, there has been a drastic increase in the number of road accidents and their severity. These accidents mostly happen due to human error. Some common human errors like distracted driving, drowsy driving, driving at high speeds and delayed reaction times of the drivers result in fatal accidents that cause loss of life and damage to property. These factors can be eliminated by means of self-driving cars. A [13] driverless car (or self-driving car or autonomous car) is a vehicle which can travel from one location to another by self-navigating without any human dependency by using sensors, cameras, radar, GPS, and AI (Artificial Intelligence). We will be implementing a Driverless Car using Deep Learning, OpenCV, DonkeyCar, TensorFlow and C++/Python.

B. Objectives

Our main objective is to implement a Driverless Car to perform the following actions:
• To detect and identify traffic lights and road signs.
• To detect and avoid obstacles, pedestrians, and other vehicles.
• To record user inputs like steering angles, throttle levels and video inputs.
• To train the DonkeyCar deep learning model using the user inputs.
• To travel to a set location without any accidents and without any human intervention.

C. Scope of Work

The major scope of our project can be described by the points listed below:
• Build a model for the Driverless Car and a physical track to train it.
• To record user inputs like steering angles, throttle levels and video inputs.
• To train the model to maintain lane and complete the entire track without any external intervention.
• To detect and avoid obstacles.
• To detect and identify road signs and traffic lights.

D. Motivation

In recent times there has been an increase in the availability of handheld devices which in turn increases distractions. Many people get distracted while driving by using their phones to receive calls, to play music, to see Google Maps, etc. Also, many people feel drowsy while driving due to which they start shifting lanes, increasing speed, etc. these

factors cause major road accidents. Such factors are leading to an increase in demand for autonomous vehicles or self-driving cars. A Driverless Car is a vehicle which has the ability of sensing its surroundings and moving safely from one location to another with minimal human intervention. And in doing this it reduces the chances of major road accidents.

II. Literature Review

In the artificial potential field method, different potential functions (PF's) are assigned to various obstacles dividing them into categories. On the basis of these PF's, the path is planned without dependence on dynamics of the vehicle. Whereas in optimal path-planning controller, the vehicle dynamics are included. In this paper [1] a model predictive path-planning controller is used, which is the combination of above two methods of path planning. It combines the advantages of both these methods by using both PF assignment for various obstacles and it increases the feasibility as it also includes the prediction of the vehicle's dynamics.

Lane Perception and vehicle control acquires information from image processing algorithms, after which the control system of the car will change the direction and speed of the car to make the vehicle autonomous. It uses technologies like Computer Vision, RADAR, GNSS, etc. Perception of lane and autonomous vehicles is included in this paper[2].

In this paper [3] a prototype of a self-driving car which uses a cheaper alternative as cameras is discussed. It uses computer vision to provide safe, quick and efficient navigation. YOLO v3 algorithm is used to localize and detect objects and lanes. Stereo vision for depth calculation, steering control, trajectory planning is also monitored. The viability of self-driving cars that use camera as the main input source is shown in this paper, providing a foundation to future self-driving cars which depend on vision based sensors in real-world implementation.

The paper [4] discuss the acceptance approach in Indian market scenarios for autonomous vehicles. There are many OEM's which has already implemented this into their exiting production vehicles. National Highway Traffic Safety Administration (NHTSA) has classified this technology into 4 different levels: Function specific Automation, Combined Function Automation, Limited Self-Driving Automation, and Full Self-Driving Automation.
Objectives of autonomous vehicles and barriers to Implementation are also described here. The transition of conventional vehicles into an autonomous vehicle by adopting and implementing different upcoming technologies is discussed in this paper.

This paper[5] explains an algorithm which uses the technique of overapproximation. Here the area in which the vehicle can drive in the presence of moving obstacles, whether a trajectory can be calculated in discrete steps is discussed. The method is explained with 3 examples:i) a situation with traffic and spacial constraints in x and y directions; ii) a situation on the highway with longitudinal and lateral constraints; iii) a situation with highway and moving traffic. Using this approach we can firstly detect and calculate all the possible trajectories in which the vehicle can move, secondly, we can detect the possibility and chances of a collision taking place.The approach in this paper can be used to detect narrow passages which are difficult to detect using other techniques.

Profound Neural Networks can extract almost all of the data from a raw RGB picture and CNNs and perform complex calculations. An Autonomous vehicle which uses identification of diverse articles in condition and groups them using a CNN model is proposed in this[6] paper. It provides proof that YOLO is a fast algorithm which can be used fro object detection in real time. It explains how Canny Edge Detection technique can be extremely accurate in detecting lanes. Thus, it combines the usage of Canny Edge detection algorithm for lane detection and Yolo algorithm for object detection.

A capable hardware system using simple embedded systems and software system developed using deep neural networks is described. This system uses parallel processes, sensor interface and motor control. The ROS uses TCP/IP as the communication network for these processes. An arduino UNO is used to control the model by interfacing it with an IR sensor for avoiding obstacles. Computer vision and machine learning technology is run on an NVIDIA Jetson TX2 board. Deep Learning algorithm is used which combines SSD and Mobile Nets. This model combines Tensorflow, Robot Operating System(ROS) and OpenCV to transform the remote controlled car to an autonomous car.

Self-driving cars platform created using an on-board computer a RGBD camera and an RC car. OpenCV is used for computer vision applications, assistance in lane detection and lane following. YOLOV3 is an object detection system and indoor positioning system (IPS). There are three main aspects in this paper[8]: The car, the IPS and the track.

In this paper[9], the system of an autonomous car is designed using Fuzzy Logic Control technology combined with AI algorithms of Image Processing techniques and Computer Vision. A smart and efficient emulation of a human-driver system is introduced. The system demonstrates all the tasks which are performed by a human driver. It also provides services like adaptive cruise control(ACC), automatic parking, blind spot warning(BSW), etc. We use a software environment called MATLAB to model the Fuzzy Logic Toolbox. The results after simulation show that this technology is very identical to the behavior of a human driver.

This paper[10] develops an LPV (Linear Parameter Varying) MISO controller by using an H-infinity approach based on the lateral error feedback at the centre of gravity of the vehicle and look ahead distance. The use of Path tracking for very fast vehicles is described in this paper.

In this article[11], a priority queue-based min-heap sorting algorithm is used to categorize lives of people according to some given parameters.The probability of a person surviving is calculated by the algorithm. This presents a fragile and flexible algorithm so that it can be designed with

consideration of ethical standards. A tree sort algorithm uses fast sorting to identify the one with lowest priority. We can ideally predict the priority of incidents using this technique. Thus, this technique will help resolve ethical dilemmas in the world of autonomous vehicles.

This paper[12] compares, Faster R-CNN, RetinaNet, YOLOv3 and YOLOv4 based on the precision of how well they detect objects in daytime and night-time scenarios. The threshold average precision difference is measured for 50% and 75% accuracy for the two datasets(Berkeley deepDrive and UA-DETRAC). It is observed that YOLOv4 has better tracking and counting accuracy in day and night whereas Faster R-CNN works better for low light images.Out of these RetinaNet is shown to have best mean average precision. The results show that they exhibit 15-20% less accuracy in low light conditions on an average with maximum discrepancy of 33%.

III. Methodology

A. Proposed Work

Nowadays, loss of life due to road accidents is becoming a serious concern. Road accidents commonly occur when a person is drowsy, drunk, distracted or driving at high speeds. These are all human errors. These minor errors can cause major accidents. To reduce these errors, we need to turn towards autonomous vehicles.

Our model will use a Raspberry Pi 4 Model B as the main processor. It will be attached to a camera which will record images of the track while driving the vehicle in User Mode. The throttle and the steering angles will be recorded as well. These images from the camera and, the steering and throttle values from the controller will be used to create a dataset.

After we collect sufficient data. The DonkeyCar library uses a data pipeline which pre-processes the data before grouping it into sample batches which will be used for training the model. Images which are not clear will be automatically discarded by the DonkeyCar. This dataset will be used to train the TensorFlow model which uses Behavioral Cloning by using CNN. In Behavioral Cloning the model learns from the training data and calculates the steering angle it needs to use based on the image. Basically, the CNN network learns by mapping the images to the steering and throttle values through Supervised Learning. DonkeyCar is an open-source self-driving library for Python. The DonkeyCar uses Keras as it's default deep learning library.

Once the model is fully trained it should be able to complete the entire track without any human intervention. The model can also be trained to detect and avoid obstacles, detect and recognize road signs and traffic lights. We can also interface a GPS System in the model. When the user enters their destination location. It's value will be stored along with the current geographic location of the user. Using this, the car can automatically transport the user from one location to another.

B. Block Diagram

The working of a driverless car can be explained using the below block diagram.

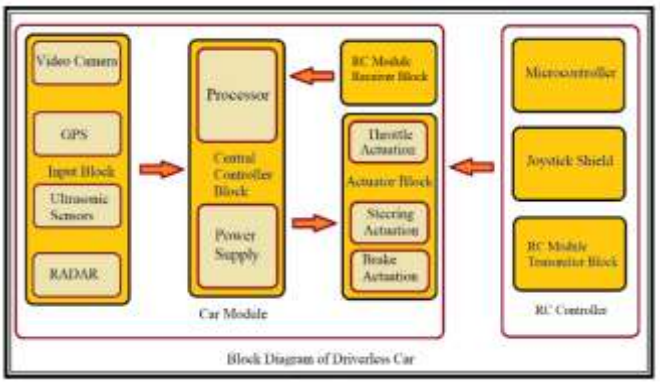


Fig 2: Block Diagram of a Driverless Car

The block diagram can be divided into two parts: The Car Module and the RC Controller

Car Module

The car module can be further divided into four blocks:

1. Input Block

It consists of devices that take input from the surrounding environment. It consists of a camera system to take video input. The camera we have used is a 5 Megapixel Raspberry Pi Camera Module. A GPS (Global Positioning System) will be used for routing and tracking. Ultrasonic sensors will be attached to the sides of the car along with RADAR (Radio Detection and Ranging) sensors.

2. Central Controller Block

This block consists of the power supply and the processor. The processor we are using is Raspberry Pi 4 Model B. The power supply we are using for the Raspberry Pi is a 10000mAH power bank. The raspberry pi takes the information from the input block. The video from the camera is used to detect traffic lights, traffic signs, check for pedestrians, detect nearby vehicles and check if the car is in lane. These actions are done by using OpenCV and TensorFlow Models. When the user enters the destination location, the processor stores the GPS coordinates of the current position of the car and the coordinates of the destination location in its memory. Then the most efficient route is calculated and followed. Input from ultrasonic sensors is used to calculate distance from curbs and other vehicles while parking. RADAR is used to calculate the angle, distance and velocity of the surrounding vehicles.

3. RC Module Receiver Block

It is implemented using a NRF24L01 transceiver module. It has the python code to receive the user commands from the RC Controller. These commands are sent to the Raspberry Pi, where they are stored and are used for training the DonkeyCar model.

4. Actuator Block

It takes input from the Central Controller Block and is responsible for the speed (Throttle Actuation), Direction (Steering Actuation) and Stopping (Brake Actuation). It has been implemented using a TB6612FNG dual motor driver and 4 brushed DC motors

RC Controller

It has three blocks:

1. Microcontroller - The microcontroller used is an Arduino UNO. It converts the inputs from the user into

digital data which can be transmitted from the RC Module Transmitter Block.

2. Joystick Shield - It acts as the interface to take the inputs from the user. It has buttons which control the direction and speed of the car module.
3. RC Module Transmitter Block - It will transmit the commands from the Arduino to the Raspberry Pi for controlling the speed and direction of the Car Module during training phase.

C. Flow Chart

The working of the Driverless Car can be explained using two flowcharts that explain the two
phases in making it. The two phases are:
1. Training Phase.
2. Trained Phase.

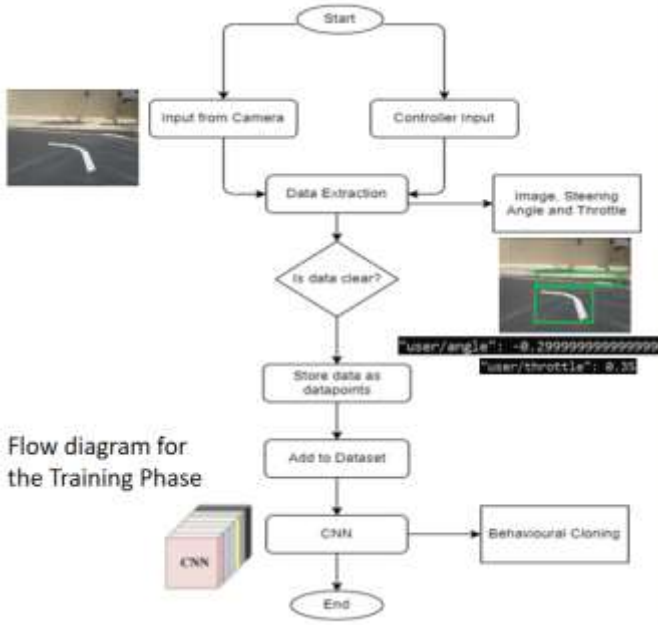


Fig. 3: Flow diagram for the Training Phase

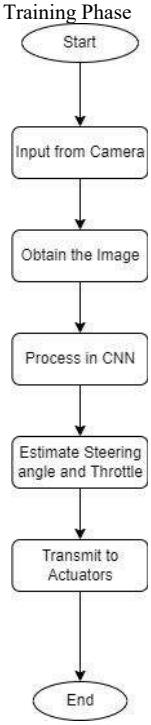


Fig 4: Flow diagram for the Trained Phase

The training phase involves collection of the dataset. The inputs taken from the camera and the remote controller are sent for data extraction. After extraction we check if the data is valid by checking the image clarity and if the other data is within thresholds limits. Here the three inputs which are taken from the data are: the image, steering angle and the throttle level. This data is processed and stored as datapoints in the Raspberry Pi. Then, this data is added to the dataset. Once the dataset is large enough, it is sent to the CNN for training which is done using a process called Behavioral Cloning.

The phase after the model is trained is called as Trained Phase. In this phase the input from the data is taken after which the image is extracted. This image is sent to the CNN for processing. After processing, the values for throttle and steering angle are estimated. These estimated values are sent to the actuators of the car model. Thus, the vehicle is autonomous.

IV. Requirements and Result Analysis

A. Hardware Requirements

The hardware components we have used to implement the project so far have been listed below:

1. Power Supply: We have used 3 separate power supplies.
   i. For Raspberry Pi – Li-Polymer Power Bank(10000mAH)
   ii. For the Arduino – General AA batteries.
   iii. For powering the motors - Orange ISR 18650 2000mAh (10c) Lithium-ion Battery
2. Single Board Computer: Raspberry Pi 4 Model B. It is responsible for processing the input data. It runs the deep learning network, and it performs the automation.
3. Microcontroller: Arduino UNO ATmega 328P. It acts as the controller for the remote.
4. Camera Module: 5MP Rasberry Pi Camera Module. It takes video inputs.
5. RC Controller Module: We are using a Funduino Joystick Shield module.
6. The car body:
   i. The metal chassis: It is the base of the car.
   ii. The motors: Brushed DC motor with a gearbox with an operating voltage of 3-12V.
   iii. The motor driver: We are using a TB6612FNG which is a dual-motor driver.
   iv. The wheels: Plastic wheels with 70mm diameter.
7. Communication Module: NRF24L01 transceiver module. We have used 2 NRF24s. One is responsible for the transmission of data from the Arduino in the remote controller. The other one is responsible for receiving the data to the Raspberry Pi in the car.

B. Software Requirements

1. Python3 - Python is a general-purpose, interactive, object-oriented, and high-level programming language. Python has a wide range of libraries and a very clear syntax which makes it useful in machine learning applications.
2. TensorFlow - TensorFlow is an open-source software library which is used for Large scale machine learning

and numerical analysis. TensorFlow combines the study of machine learning, deep neural networks and algorithms for processing information.

3. OpenCV - OpenCV stands for open-source computer vision. It is an image processing library of the program functions aimed at Real-Time Computer Vision. Computer vision allows computers to see and process visual data.
4. DonkeyCar - It is a platform used for developing small scale autonomous vehicles. It uses a combination of Raspberry Pi, Python as the base language, TensorFlow, keras, OpenCV, etc. to achieve self-driving.
5. Keras - Keras is a software library that is open-source. Its role is to provide a Python interface for ANN or artificial neural networks. It can also be used to interface the TensorFlow library.

C. *Result Analysis*

Fig 5: Model of the Driverless Car

In our project, we have implemented a driverless car by making a demonstration model using a Raspberry Pi 4 Model B as the main processor. It is attached to a camera that recorded images of the track while driving the vehicle in User Mode. The throttle and the steering angles were recorded as well. Our current model was trained with over 40,000 images which were recorded by driving the model on the track for 20 laps. These images from the camera and, the steering and throttle values from the controller were used to create a dataset. Out of these, 10 laps were completed by driving the model in the clockwise direction and the remaining 10 were completed by driving it in the anti-clockwise direction. We needed to do this because, for the first model we trained, we had only driven it in a clockwise direction. Due to this, it was unable to correct itself by making left turns. After training the second model, it could auto-drive and maintain lanes in Local Angle mode. It is able to complete the entire track without any human intervention.

We have trained the model on a large data-set due to which it has more accuracy. The model also exhibits real time data processing and edge computing as the data is continuously being processed by the on-board computer.

The trained car can run in 3 modes namely:

a. User: Where we have control over both the steering and throttle. The data is recorded in this mode.

b. Local Angle: In this, we first have to set the maximum throttle level that the car can use. After which the trained model controls the throttle and steering angle. The word local refers to the model which we have trained, which is locally hosted on the raspberry-pi.

c. Local Pilot: In this, the trained model will be controlling both the steering and the throttle. The maximum throttle level will not be set by us, thus giving the model complete control. This Mode is not very reliable.

Fig 6: Remote Controller

We have trained the model in User mode. We have used a web URL called as Donkey Monitor for recording the data. When we tried to train the model using the joystick controller the data being recorded was not getting stored in the database.

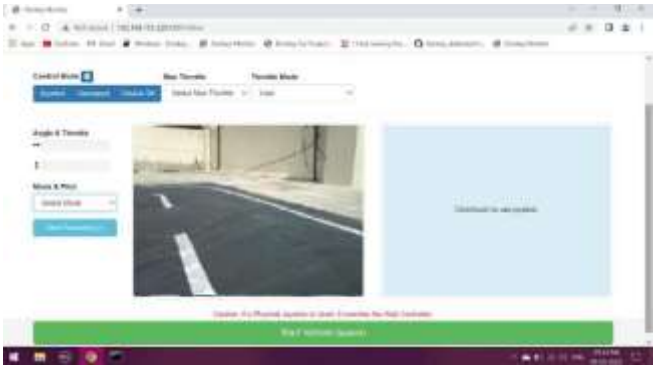
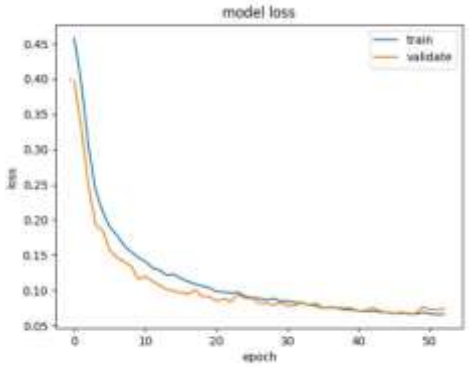


Fig 7: Donkey Monitor

The graphs given below show the model loss or the difference between training values and validate values (or prediction values) in training losses per epoch. The second model was trained for 54 epochs with a batch size of 227.



a) model 1

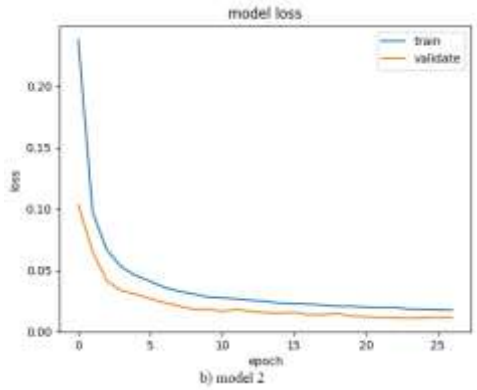


Fig 8: Graphs of model loss per epoch

V. Conclusion & Future Scope

Driver errors while driving are the most prominent cause of traffic accidents. Driverless cars have the following benefits related to road transport: reduce stress due to driving, reduce driver cost, provide mobility of non-drivers, increased safety, and decrease in traffic congestion. Even a person with disabilities can make use of autonomous vehicles the same as any other individual and can even customize the vehicles for more convenience. A self driving car not only increases safety, but it also improves efficiency and productivity of the people who spend most of their time stuck in traffic and trying to reach their work places. Donkey is a self-driving car platform which uses a high-level Python library to attain autonomous driving. We have trained the Donkeycar to drive on its own based on driving style and path. This uses a supervised learning technique often referred to as behavioral cloning. Next, we carry out steering and throttle calibration. The reason for calibrating the car is to make it drive consistently. Later, the car is driven using a web controller/ physical joystick controller. As the driving car becomes reliable, we used Keras to train a neural network to drive like a human.

We have trained the model on a large data-set due to which it has more accuracy. The model also exhibits real time data processing and edge computing as the data is continuously being processed by the on-board computer. This also helps with data security since our is not being shared with any external device or server. Some common challenges affecting the growth of the autonomous industry are misconceptions related to the technology of driverless cars and also a lack of faith in the safety and ethical liabilities of the vehicles. This issue can be overcome by educating the public on how these vehicles work and the vast testing which is carried out before making them available to the public.

We can improve the model in the following ways:

1) By including a GPS System, we can set the destination location of the vehicle. The user can just enter the destination location after which the system can automatically take the start location saving time

2) LIDAR(Light Detection and Ranging) System can be added. This would drastically reduce collision rates.

3) We can include another Camera directed backward, for parking assistance. This would be needed to train the model for driving in reverse.

References

[1] Yadollah Rasekhipour, Amir Khajepour, Shih-Ken Chen, and Bakhtiar Litkouhi, "A Potential Field-Based Model Predictive Path-Planning Controller for Autonomous Road Vehicles," IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, August 2016.

[2] Tianqi Liu, "A Review of Lane Perception and Automobile Control Based on Computer Vision," 2017 Second International Conference on Mechanical, Control and Computer Engineering, IEEE 2017.

[3] Bhaskar Barua, Clarence Gomes, Shubham Baghe, and Jignesh Sisodia, "A Self-Driving Car Implementation using Computer Vision for Detection and Navigation," Proceedings of the International Conference on Intelligent Computing and Control Systems (ICICCS 2019), IEEE 2019.

[4] Rajasekhar MV and Anil Kumar Jaswal, "AUTONOMOUS VEHICLES: THE FUTURE OF AUTOMOBILES," IEEE.

[5] Sebastian Söntges and Matthias Althoff, "Computing the Drivable Area of Autonomous Road Vehicles in Dynamic Road Scenes," IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, August 2017.

[6] Irfan Ahmad and Karunakar Pothuganti, "Design & implementation of real time autonomous car by using image processing & IoT," Proceedings of the Third International Conference on Smart Systems and Inventive Technology (ICSSIT 2020), IEEE 2020.

[7] Luis Bill and Hamid Shahnasser, "Development and Implementation of an Autonomously Driven Vehicle Prototype," 2019 2nd International Conference on Electronics Technology, IEEE 2019.

[8] Jacob Newman, Zheng Sun and Dah-Jye Lee, "Self-Driving Cars: A Platform for Learning and Research," 2020 Intermountain Engineering, Technology and Computing (IETC), IEEE 2020.

[9] Abdulrahman H. A. Widaa and Waddah Abdelbagie Talha, "Design of Fuzzy-Based Autonomous Car Control System," 2017 International Conference on Communication, Control, Computing, and Electronic Engineering (ICCCCEE), Khartoum, Sudan, IEEE 2017.

[10] Matteo Corno, Giulio Panzani, Federico Roselli, Michele Giorelli, Davide Azzolini, and Sergio M. Savaresi, "An LPV Approach to Autonomous Vehicle Path Tracking in the Presence of Steering Actuation Nonlinearities," IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, IEEE 2020.

[11] Md. Azharul Islam and Shawkh Ibne Rashid, "Algorithm for Ethical Decision Making at Times of Accidents for Autonomous Vehicles," 4th International Conference on Electrical Engineering and Information & Communication Technology, IEEE 2018.

[12] Stephen Galea, Dylan Seychell and Mark Bugeja, "A Survey of Intelligent transportation systems based Modern Object Detectors Under Night-time Conditions," Proceedings of the Third International Conference on Intelligent Sustainable Systems [ICISS 2020], IEEE 2020.

[13] Ben Lutkevich,"Self-Driving Car (autonomous car or driverless car)", techtarget.com.[Online].Available:https://searchenterpriseai.techtarget .com/definition/driverless-car(accessed Nov. 11,2021).

[14] Dipak K. Dash,"Last year's road accidents were most fatal in 5 years", timesofindia.com.[Online].Available:https://timesofindia.indiatimes.c om/india/last-years-road-accidents-were-most-fatal-in-5-years/articleshow/87574502.cms(accessed Nov. 9,2021).

[15] "USDOT Releases New Data Showing That Road Fatalities Spiked in First Half of 2021,"nhtsa.gov.[Online]. Available:https://www.nhtsa.gov/press-releases/usdot-releases-new-data-showing-road-fatalities-spiked-first-half-2021(accessed Nov. 12, 2021).

[16] "The 6 Levels of Vehicle Autonomy Explained",synopsys.com.[Online].Available:https://www.synopsys.c om/automotive/autonomous-driving-levels.html(accessed Nov. 11,2021).

[17] "Automated Vehicles for Safety",nhtsa.gov.[Online]. Available:https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety(accessed Nov. 7,2021).