International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

AN IN-DEPTH STUDY OF LANE DETECTION FOR AUTONOMOUS CARS USING COMPUTER VISION TECHNIQUES

Henil Gajjar
Department of Electronics and Communication Engineering,
Institute of Technology, Nirma University, Ahmedabad - 382 481, India

Stavan Sanyal
Department of Electrical Engineering,
Institute of Technology, Nirma University, Ahmedabad - 382 481, India

Abstract: **The article discusses the importance of self-driving cars to improve road safety and reduce the number of accidents caused by human error. Self-driving cars not only reduce human error but also help reduce driver fatigue. We further explore the use of computer vision in autonomous cars, with previous research relying on deep learning algorithms with LiDAR sensors which can be expensive. The authors propose a more cost-effective approach using simple computer vision algorithms such as color space transformation, Canny edge detection, and Hough line transformation to detect lane lines and steer the car accordingly. This approach requires less operational hardware and can be implemented using affordable boards like Raspberry Pi and Nvidia Jetson Nano. The article also highlights the reconstruction of a remote-controlled car that had a 95% accuracy using a certain set of parameters was a tool for understanding autonomous cars better.**

Keywords: **Computer Vision; Autonomous car; Lane detection.**

I. INTRODUCTION

As the pace of global progress quickens and technology becomes more integrated into our daily lives, the risk of forgetting key details looms larger than ever, with potentially disastrous repercussions. While some events, including accidents brought on by unforeseen reasons, may be beyond our control, computational help can be used as a powerful tool for reducing such incidences. The automotive research sector has switched its focus towards the creation of electric autonomous automobiles as a result of the rising popularity of electric vehicles, which outperform their combustion-engine counterparts in terms of technology and the environment(Brandies et al, 2020)
The self-driving car phenomena has two dual fundamental goals. The first is to lessen driver mistake, which will improve personal safety, save lives, and ultimately lower the frequency of collisions. The second is to ensure that vehicles are used to their fullest potential and increase overall transport efficiency. Additionally, self-driving vehicles give people with physical disabilities—including those who are visually impaired—a better sense of independence, enabling them to live their lives as they see fit. Self-driving cars have also made a substantial contribution to cutting carbon emissions, which has increased the economic attraction of electric vehicles. For instance, the most recent autonomous Teslas have successfully avoided accidents brought on by human mistake in numerous instances, thanks to the superior hardware and software capabilities of Tesla vehicles. The use of automatic cruise control systems has helped drivers feel less fatigued, while autonomous vehicle technology has enhanced accessibility and convenience for those with disabilities.
Six levels of automation have been established by the Society of Automotive Engineers (SAE), with Level 0 representing a complete absence of automation. With the assistance of active safety technologies, the driver is in charge of completing the driving task at this level. On the other hand, Level One Automation entails an Automated Driving System (ADS) that controls the vehicle longitudinally or laterally, but not the entire Operational Environment Detection and Response (OEDR) system. The development of ADS has made significant progress in recent years, and a number of related products are anticipated to be introduced in the upcoming decade. At the moment, a great number of researchers are concentrating on the creation of intelligent automobiles with the goals of reducing traffic accidents and ensuring safe driving. Various methods for alerting drivers to the structure of lanes and the presence of other vehicles in those lanes, as well as potential lane departures or crashes, have been developed as a result of technological advancements.
Pilot driving, Traffic assist, Unmanned parking assistant, Highway assist, Fully assisted parking, Super Cruise, and Multiple Lane Control are some of the technologies that a number of businesses, including Audi, Tesla, Nissan, Ford, and Bosch, are actively implementing to enhance the customer experience with driverless vehicles. On the off chance that these innova-

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

tions are effectively carried out, they will altogether affect society in more ways than one. From a vehicle client viewpoint, there will be less car accidents because of decreased human mistake, which will make rides not so much upsetting but rather more agreeable for travelers, including the people who are impaired or youngsters. People will no longer have to worry about owning a car as a result of this, and event management will be improved in the event of vehicle failure or driver incapacity.

From the point of view of transportation operations, there will be less congestion, better traffic management, and dynamic routing that will lead to more effective real-time navigation and an effective framework because vehicles will be under more control and operations will be coordinated better. Additionally, this will improve infrastructure asset utilization, such as the development of parking lots and roads (JiLindholm, 2013). Technologies for automated driving will set a new standard for user experience. According to (Yue et al., 2020), artificial intelligence (AI) has made a significant impact on our day-to-day lives and has fundamentally altered the automation industry. Computer Vision is one of its subsets, and it is concerned with image processing and serves as a medium for converting any digital image into a format that can be read by machines. Computer Vision is an essential tool for the automation and autonomous-self-driving car fields because it enables image, pixel size, and feature analysis.

Autonomous Self-Driving Cars use computer vision to improve their efficiency and safety in daily use. It can be used to create 3D maps, detect and classify objects, and collect data for autonomous vehicles, among other applications (Srivastava, 2019). Maps can be made that can identify stationary objects like buildings and trees using Lidar (Li et al., 2016) and camera inputs. Another use for Computer Vision is object recognition, which involves object detection, object localization, and image classification. In addition, cars can be tracked using computer vision to improve road safety by monitoring and anticipating the patterns of other drivers. Self-driving cars also use Lane Detection, which enables them to follow a particular lane and recognize road turns and curves. Constant traffic sign discovery and acknowledgment frameworks in light of GPUs are likewise utilized utilizing multi-class learning and multi-object following, and person on foot identification for independent vehicles utilizing multi-otherworldly cameras. All of these Computer Vision applications help autonomous vehicles become aware of their surroundings and act accordingly.

Comparing and learning from self-driving cars on a smaller scale is essential as technology advances and becomes more advanced. Image processing, high-frequency steering actuation, machine learning, and object detection are among the similar technologies utilized in Tesla, Mercedes, and Volkswagen vehicles. Tesla, which has been designing and producing self-driving vehicles for half a decade, is the leader of this generation. They are nearly a decade ahead of the competition with their level 4 driving automation technology and have a reliable and human-like performance due to their advanced technical team. However, self-driving cars have a significant disadvantage in that they are prohibitively expensive due to the need for high-end hardware and software. Additionally, it is challenging for businesses to accurately predict human-like behavior. India and Argentina, two developing nations with low rates of per capita income and market opportunities, must lower the price of automobiles for the greater good and profit. To address these difficulties, we have fostered a minimal expense, 18:1 limited scope self-driving vehicle that can recognize paths and drive inside their limits with the assistance of item recognition that stops the vehicle assuming an item is distinguished in front. This vehicle depends entirely on PC Vision to identify paths, subsequently decreasing the computational power required. Such little activities can spur and rouse more youthful ages to investigate and foster more productive codes and calculations for comparative results.

II. RELATED WORKS

(Newman et al., 2020) scaled down a real life implementation of modern self-driving cars of the world. The research was conducted using five vibrant colored cars that had premium hardware including Intel's RealSense D435i RGB-D Camera Module interfaced with Nvidia's Jetson TX1 Carrier Board and a Proportional Integral Derivative controlled steering actuators. It used OpenCV and YOLOV3 algorithm to understand the lane lines and send the actuating signals to the attached Arduino Nano that finally actuated the servo using PID. These Cars were made to loop a predetermined track that was monitored using two cameras mounted on the ceiling forming a solid Indoor Positioning System (IPS). Cars observed the lane lines and based on that, PID actuators controlled the servo motors. With the attached RGB-D Camera module, cars were able to measure the depth ahead and stop or slow down if there was an obstacle or another car in-front. This implementation directly reflects a real life simulation where the cars use various GPS and depth sensors along with several cameras to navigate the car amidst other cars properly in the real world.

(Pannu, G. et al, 2015) aimed to fabricate a monocular vision self-driving car model with Raspberry Pi used as the Processing chip. Other accessories which were used in the research included a camera module, a motor driver, Wifi-dongle, Ultrasonic sensors, AAA batteries and Servo motor. Introduction of lane detection, object detection and other existing algorithms has been done in order to control and make the car capable of reaching its destination safely and effectively while eliminating the factor of human errors and any kind of unwanted risk. The camera input is directly fed to the Rpi where it performs lane detection and object recognition using OpenCv and YOLO and accordingly the motor driver drives the servo motor. Secondly the project provides an option of Remotely accessing the car via mobile application thus enhancing user experience. The overall research work can be enhanced by the implementation of machine learning as the efficiency of the algorithms can be improved significantly.

(Barua, B. et al., 2019) suggested a safe, quicker and efficient prototype for navigating by implementing a virtual environment using computer vision. The focus was on providing a cheaper alternative for achieving an autonomous vehicle by introducing camera sensors instead of lidars which are expensive and not feasible for mass production. Four cameras had been implemented, - one camera front-facing, one camera rear-facing and one wide-angle camera mounted on each side of the car but they heavily rely on the camera facing front side. All the cameras had been coupled with stereo cameras allowing it to virtually create human binocular vision thus providing it with the ability to capture 3-D images and this is used to accomplish stereo vision which has then been used to calculate the depth of each object. After performing object detection and lane detection on the images 3-D and Frenet coordinates have been obtained that have been used later for trajectory planning. The simulation has been tested and the results show that it has performed significantly well and may be used as a model for autonomous car projects in the future.

In (Yaovaja, K. et al., 2019) implementation of autonomous tracked mower, a Vision based remote controlled system was used to identify the difference between cut and uncut grass. A tank-like continuous track was designed with a nylon line based cutter rotating at around 90 rpm. National Instruments' NI myRIO was used as a primary controller that sent PWM signals to the Mower's H-Bridge Motor Driver for driving the wheels at required torque. A 32-bit single camera system was used inorder to achieve a proper image segmentation. A Template matching was done in order to guide and control the mower and to differentiate between cut, uncut, and the boundary of the grass with the help of Convolutional and Deep Neural Networks. This implementation is an exceptional use in daily life of a human being as it becomes relatively easy to cut the lawn whilst having a cup of tea during a bright sunny morning.

(Secuianu, F., Lupu, C., 2018) projected an autonomous self charging robot that uses computer vision along with the 3 axial digital compass system to guide itself automatically towards the charging dock and recharge its lost power. Equipped with a couple of HC-SR04 Ultrasonic Sensors, a RaspiCamV2 along with 3-axis Digital Compass operating on I2C interface, this self charging robot is able to sense the environment, align its position according to the dock, and charge itself automatically without any human interaction. Working on A*, an updated version of Dijkstra's algorithm for grid maps, this car forms a virtual grid en route for the dock. Working on a compass where only four directions are possible, the car is limited to travelling in the form of a grid which is not that useful in the real world where a car is not always allowed to move in a grid. But with more advanced sensors and GPS, this self-docking concept is a great utilization of computer vision's application that can easily modernize earth with the growing demand for electric vehicles around the world.

(Tian, H., Ni, J., 2018) proposed the work of manufacturing an autonomous system that can detect and track paths for a formula student self-driven race car. A LIDAR-vision assisting method has been used for detecting the traffic cones that have been used as track marks. The detection algorithm also incorporates GPS-INS data and LIDAR odometry. The environment perception has been divided into three parts - Laser Obstacle Detection, Vision Based Obstacle classification and Cone color detection. Laser Obstacle Detection has been achieved by Point cloud preprocessing and accumulating multiple frame point data. After gathering all the points of objects, the Euclidean cluster extraction method is used for dividing and then grouping the point clouds that are reflected by the same object followed by calculating the centroid location. Vision Based Obstacle classification is done with the camera by calibrating it with LIDAR, followed by cone detection based on SVM. During the starting lap, the vehicle operates over LIDAR and visual systems, and the motion is controlled based on the logic to follow the middle line. During this mapping node and odometry is run simultaneously so as to track the trajectory and create a map of the track. In the next lap, GPS-INS is used by the vehicle for positioning and tracking the trajectory recorded in the commencement lap. As the track is now known the tracking algorithm calculates speed for each waypoint, thus improving the lap time greatly.

Obstacle Avoiding unmanned aerial drone which (Devos, A. et al., 2018) mentioned, used LiDAR, GPS and Gyroscope and Barometer equipped Pixhawk Flight controller to control its path. MAVLink protocol based communication between Raspberry Pi Pico and the flight Controller was observed in order to send signals to the drone's four Individual BLDC motors. Based on a two-neutron recurrent network with synaptic plasticity (E. Grinke et al., 2015), the drone uses two simple LiDAR sensors to detect obstacles and intelligently avoid them during the course of its flight. A surroundings map through Simultaneous Localization and Mapping Technique was created and the drone was allowed to run the course according to the distance and position values received from the LiDAR sensors. Pixhawk, SLAM and LiDAR based Unmanned vehicles are used in the cases where predetermined paths are available and the vehicles are only meant to course the given paths. However, predicting information from the LiDAR values and SLAM can be a task within itself and these advanced technologies are greatly being used in our day-to-day lives.

(Coelho, L. et al., 2021) formulated a project that is capable of predicting position and orientation of visual-beacons from the images taken with the camera. The equipment used in the project include a blimp controlled via remote, provisioned with a camera of 24 bits with a resolution of 640x480 resolution, RGB, per pixel. The system has two basic tasks, first is to determine the pixels that correspond to characteristic points of the beacon when the given image has known geometrical properties of the visual bacon. Second is to acquire the camera's orientation and position, thus that of blimp in order with the visual beacon. The image processing is done with the help of Segmentation, Labeling and Identification of markers. Then geometrical methodologies are applied to compute the points and then a calculation of the solution is done using these input

points and thereafter solving the nonlinear system and then local search is performed with these as initial conditions. The experimental results show accurate, methodical and efficient working of the system and thus can be used in future for various other purposes.

(Vladimirovich, P. et al., 2015) published a unique concept of three wheeled autonomous vehicle equipped with a custom low level control board having two of Atmel Atmega128 AVR microcontrollers. With hall sensor equipped BLDC motors on both the back wheels of the car, it is easily maneuverable. A Dual Camera equipped three wheeled car is used to perceive depth as well as the information correctly with the help of OpenNI and OpenCV libraries. The data from the Depth Camera perceived using OpenNI library is then sent to OpenCV library based modules where other information such as color and geometry of the object is identified. Based on the set color and geometry patterns the coordinates and the depth of the obstacle is identified and the information is then processed and transmitted using I2C protocol to another Atmel Atmega128 AVR controller that gives PWM signals to IR2132 motor driver that actuates the motor mounted in the rear wheels. This implementation might be a great use in the current market as a very cheap controller board is used to process the data and actuate the signals.

(Manigel, J., Leonhard, W., 1992) described guidelines for an unmanned vehicle to run along roadways by following a white guideline as directed by the visual signals sensed by Computer Controlled Display(CCD) camera which focuses ahead and down, hence an area of 5 to 25 m is detected ahead as the windscreen has been mounted in front of it. The line coordinates are transmitted further that have been detected by the image processor, via transputer links to other transputer networks. The relative position of the vehicle and curvature of the road are identified by algorithms that use geometric coordinate-transformation and a dynamical model (Kalman filter). Lateral deviation, yaw-angle deviation, road curvature and velocity of the vehicle is used to compute the reference angle of the servo actuated electrically. Putting all these together, it took around 70ms per image for control and recognition, and all these have been tested on simulations and with a VW Bus Caravelle on typical Autobahn scenes at a velocity of up to 120 km/h. Thus, suggesting that this methodology is suitable and provides a pragmatic approach for implementation in near future.

So far, a common pattern was observed when steering actuation is considered in an autonomous car. Either an independent motor controller driven steering control system or a PID based actuation was observed. As far as processing is concerned, a variety of interfacing methods were seen with LiDAR and camera modules. A custom CNN model or a YOLO algorithm was also observed in many of the research. Implementing a CNN or YOLO not only increases code complexity but also makes the overall project shift to the expensive side of the budget friendly curve. This paper tries to make the car less complicated, decreasing the code complexity and the use of expensive hardware by implementing a simple yet powerful OpenCV based lane-detection algorithm on a budget friendly computer that monitors the lane lines and actuates the steering using a slave microcontroller controlling the servo motor coupled to the front wheel axle.

III. DATASETS AND METHODOLOGY

The primary objective of the project was to detect lanes and adjust the car's steering accordingly. To achieve this, the approach was divided into two main blocks: lane navigation and command actuation. The methodology can be viewed as two distinct parts, with the first involving detecting and navigating lanes and the second involving carrying out the necessary steering commands.Fig. 1 shows the flowchart for the process.

The camera input is fed to the Raspberry Pi that is the processing chip. The RPi then performs various processing algorithms over each frame fed to it (Maksimovic et al., 2014). Each frame is then scaled down to a resolution of 320 x 240 as processing a lower resolution is more feasible. The image is by default in BGR(Blue, Green, Red) format which is then converted to HSV(Hue, Saturation, Value) format as HSV is better for object detection because it is more resilient to changes in external colour thus reducing the risk of failure in detecting shades of a particular color. Thereafter an image mask of green(track color) color is created that is then superimposed over the actual frame so as to detect only the track that is in front of it. Following this, edge detection is performed over this masked image that provides only the edges of the track, this image is then cropped to get the Region of Interest (ROI). This is the region on which further processing is done as this is the primary region in front of the car and plays a major role in steps involved further in the methodology. It is important to perform edge detection before getting the ROI considering the fact that if ROI is computed first then followed by edge detection, this will lead to false introduction of 2 more lines that are not a part of the track but are the boundaries of the ROI which further will lead to inaccuracy in lane detection and provide undesirable results.

After the ROI has been determined, Hough Line Transformation is performed over this that gives all the lines it detects as per the parameters provided to it. Using these lines, the actual lane is detected which is then represented by two green lines imposed over the actual track. Once the track has been detected, the two green lines are then used for calculation of the steering angle and then a red line has been used to indicate the direction that the car is supposed to follow. The slope of this red line denotes the angle at which the car is supposed to steer.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

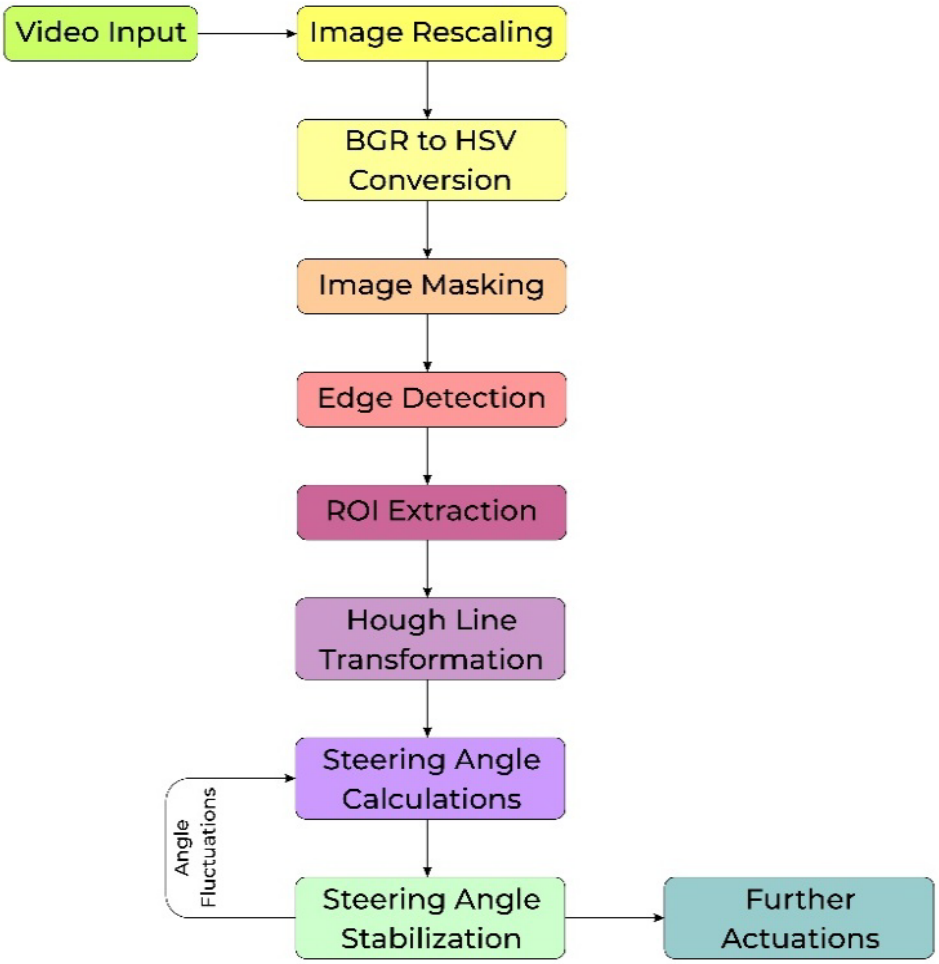

Fig. 1. Flowchart for the Algorithm

Once the RPi is done with calculating the steering angle, the angle is then stabilized by performing an algorithm that we have developed. The need for steering angle is to prevent any kind of gross change in the angle that may cause the Servo motor any kind of damage and lead to unwanted results making the whole system less efficient. This stabilization is done by comparing the current steering angle to the new calculated steering angle, then their difference is calculated. Any difference greater than 5 degrees will be considered as 5 only, hence avoiding any sudden change in the steering angle. Now this final angle is then fed to the Arduino via I2C protocol through a USB cable to I2C bus. This is done by using 'Pyfirmata' library that allows the user to code Arduino in python without having to use the Arduino IDE.

The Arduino further actuates the Servo motor it controls thus changing the steering angle of the car. A motor driver has been interfaced with the Arduino that controls the rear motor of the car and is provided with a constant PWM value so as to keep the speed of the car constant and to achieve better results without compromising the speed one desires. This PWM can be varied as per our convenience, for testing phase the speed is kept low such that one can monitor each frame and debugging becomes easy. Once the testing phase is done with, the PWM value is increased in order to see how accurate the algorithm is at high speeds.

IV. RESULTS AND DISCUSSIONS

1.1 Autonomous Car Assembly

For the purpose of developing a prototype while keeping cost efficiency in mind, the whole project has been developed around a pre-build RC car that is easily available in the market. Initially, when the factory manufactured RC car arrived, it was powered by Nickel-Cadmium batteries(rechargeable) rated at 4.2V and 0.9A and it was an all wheel drive system with dc motors coupled to the front and rear axle of the wheels. Another Dc Motor was mounted at the front axle that helped the car to turn, acting as a spring based steering actuator for the car.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

The remote controller gives a signal to the motor in which direction to turn, i.e. anticlockwise or clockwise. However, one of the major drawbacks of this method is that the turning angle is not under one's control, that is one cannot turn the steering wheel to the desired angle rather than the extremities. This method does not provide the user with flexible turning angles.

In order to overcome this limitation, the car was dismantled and the whole powertrain had been modified as per the flexibility of providing high frequency customized steering actuation and making the car similar to the real world. Firstly, the car is powered by a 18W rated Power Bank that is acting as a direct source of supply to the Raspberry Pi and L298N DC Motor Driver. The in-box batteries had been replaced by the power bank as they were unable to meet the requirements of the RPi and the motor driver. For data acquisition Logitech C920 HD Pro Webcam was used as it records frames in 1080p at 30fps and also has low light feature along with image stabilization. The camera is interfaced with RPi via USB port. The RPi further actuates an Arduino Uno microcontroller board that is used to give signals to the Servo motor and the motor driver that controls the rear wheels.

A L298N DC Motor Driver has been used instead of the famous L293D as the former can draw up to 2A from both the channels whereas the later can only draw a maximum of 0.6A thus more suitable for high torque and high RPM motors. The front wheels have now been coupled with the Servo motor. Introduction of the Servo motor has provided the car with flexibility in terms of turning angle, as every time the degree of turn changes, the car has to turn accordingly, which was not possible in the default assembly. A 3D printed mounting had been made that integrated the shaft of the servo to the front wheel axle that directly steered the car without the introduction of a spring.

Thus, making it a modular design car as each section is independent of the other and can function while the other sections have been removed or disconnected for a while. This has been done by providing supply to the RPi and Motor Driver separately through the power bank and connecting the Arduino to the RPi via USB port. For instance, if one wants to only take the camera input and not perform any steering actuation, it can be done by simply plugging out the Arduino from the RPi. To sum it up, for data acquisition we have the camera module, for image processing and all other calculations Raspberry Pi has been employed and the Arduino is responsible for command actuation. Fig. 2 shows the fully assemble PiCar

Fig. 2. Fully Assembled PiCar

1.2 Lane Navigation System

The Lane Navigation System used in the car can be broken up into two steps namely Insighting (Lane Detection) and Inclination Outlining (Steering Actuation). Insighting metamorphoses the raw input image of the road from the webcam to cartesian coordinates of detected lane line(s) while the Inclination Outlining reshapes the identified lane lines into an angle that is further transformed to a PWM signal before fetching it to the microcontroller for steering actuation. Before lane line detection in a video, a thorough implementation of it on a still image is important after which the process can be repeated in a loop to achieve a stable video processed lane lines.

Lane Detection

As mentioned earlier, Insighting is a method of metamorphosing the camera feed of the road into coordinates of the lane lines detected. Insighting lets an image go through a number of steps before getting a stable coordinates of lane lines.

Color Abstraction

The first and foremost step of the Lane Navigation System is Color Abstraction. Isolating a certain color from the rest of the image in order to avoid unwanted noise and errors for a proper processing of an image is the motto of this step. A non-sticky green tape was used for marking a formidable track for the PiCar. Green seemed to be a unique color and was not merging with the other objects in the background. Fig. 4 is the stable output from the webcam of the PiCar.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

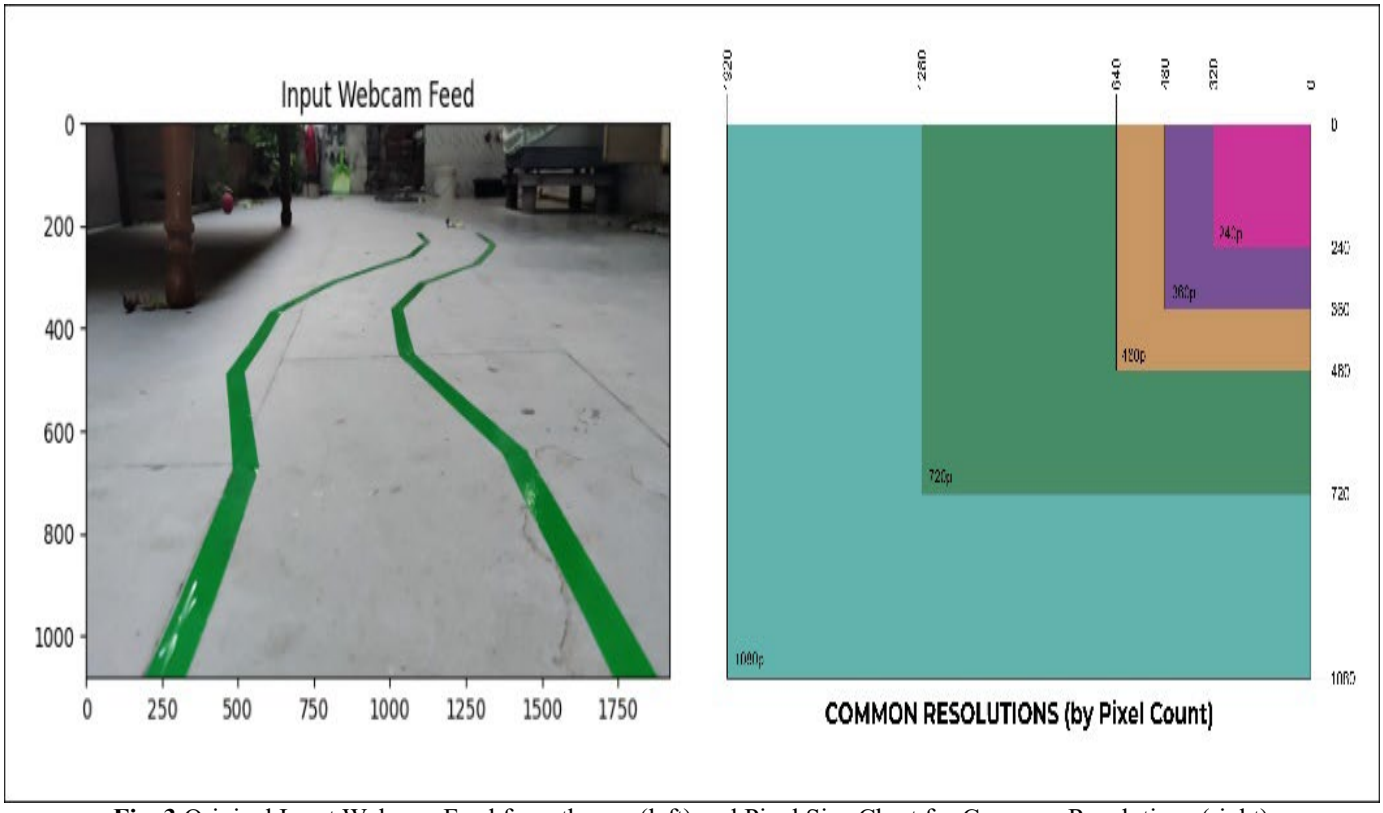

Fig. 3. Original Input Webcam Feed from the car (left) and Pixel Size Chart for Common Resolutions (right)

As different colors are interpreted by the RGB color model at different parts of the green tape, HSV color model renders the green tape as a single color irrespective of the brighter or the darker shades of green that is captured by the webcam. It is best explained by the following Fig. 4. It is noticed that the entire green tape is rendered as a one neon colored strip in the HSV color model while the green tape in the input webcam image is not well lit in some areas causing the tape to be light green at some parts and darker green at another. The OpenCV library due to its legacy reasons detects and processes the images in the BGR color space that is slightly different than the RGB color space; just the sequence order of the model is swapped.

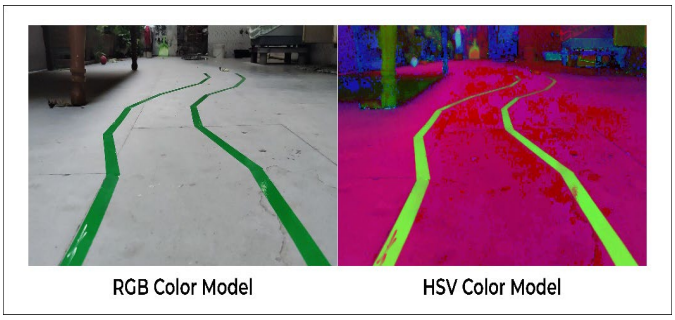

Fig. 4. Comparison between the RGB and HSV colored Images of the road

Once the image is converted into the desired HSV model, the color lifting process can be started. The color lifting is basically the process of focusing on the desired color and removing the unwanted colors from the image. This is done by specifying a range of the green color to the processor. For isolating a color from the HSV color space, a limit to bound the color space is necessary. A Green color is in about the 60-180 degree range of the Hue color spectrum. As far as the Saturation and Value are concerned, 40-255 ranges help in eliminating any shades in the color that have occurred throughout the green-taped lane line. Hence, it is clearer to bound the converted HSV image between the limits [60, 40, 40] and [180, 255, 255] acting as upper limit and lower limit respectively. The process of bounding an image within the given limits is quite simple; all the pixels that are within the limits of the set range are given the value 1 and the pixels that are not within the limits are set as 0. Thus, Fig. 5 shows a binary mask that displays white portions where the green areas were present in the actual image. This step concludes the Color Abstraction, the masked binary image formed is further used in the upcoming steps to get more information out of the image.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

Fig. 5. Input and Color Masked Image Side by side

Edge Detection

After masked image is formed, the most crucial step for the future lane line detection algorithm is to identify the edges of the color masked image. Demarcating the edge of an image is one of the most fundamental and significant step in image processing. Edge of an image carries a lot of information with it. Edges are very important in reckoning the structure and Photometrical, Geometrical and Physical properties of an object in the given image. Such properties give birth to variation such as discontinuities, local extrema, and 2D features at the junction of two edges. Hence the plan of edge detection is to localize the variations formed due to the Photometrical, Geometrical and Physical properties of an object in the given image. To efficiently disclose edges from an image, John F. Canny in 1986 developed a sufficient algorithm that is widely executed in the modern world. Canny's Edge Detector has three objectives, namely minimizing errors, a good localization and Minimal Response to Single edges of an image. In Fig. 6, it is observed how real and efficient edges are formed from the input color masked image. This edge detected image is now further processed and enhanced in order to extract more details like lane line coordinates from it.

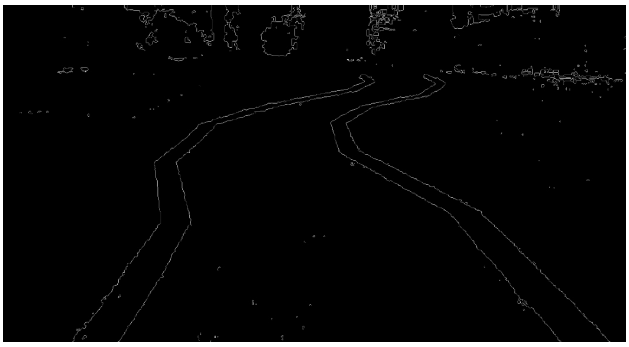

Fig. 6. Edge Detected Road Image

Region of Interest Segregation

After the edge detected image of the green lane line, it is observed that there are unwanted noises present in the upper part of the image. Here it is necessary to focus only on the track rather on the entire image. This segmentation of a certain part of an image in order to process the important and useful part of the image is called Region of Interest Segregation and the Segregated part is called the Region of Interest of the input image. According to(Md Jan et al., 2020),A User Defined Mask containing the area to be marked as a Region of Interest undergoes bitwise-and operation with the Edge Detected image from the previous step. As the Top 3/5th part of the Image contains unwanted elements, User-Defined Mask has white pixels on the bottom 2/5th part of the image which after undergoing bitwise-and operation, would give a clean, soft edge detected image as shown in Fig. 7of the part of input image which is required for smooth processing of the image.

Fig. 7. Region of Interest Retrieved Image

Line Segment Detection

After the Region of Interest Retrieval, it is easily possible for humans to identify four lines representing two main lane lines in the image but that is not the case when it comes to machine understanding. For the computer, the image is just a batch of white pixels scattered randomly over a black surface. Hence, In order to extract relevant information out of the image, a famous algorithm patented by Paul V. C. Hough called Hough Transformation is applied. Hough Transformation is widely used in computer vision to identify various shapes like circles, lines, ellipses, etc. present in the image. Thus, Hough Line Transformation would be used to extract lines from a batch of white pixels scattered over an image shaped like a white line. Before understanding how Hough Line Transformation works, knowledge of Hough Space and the Mapping of edge points onto it is necessary.

Probabilistic Hough Transformation requires 5 parameters to effectively identify line segments from the edge detected image. These Parameters include, which sets precision in distance of the pixel, acts as angle precision factor of a pixel, Minimum Threshold Value holds for minimum number of votes before calling it a line, whereas Minimum Line Length and Max Line Gap are used to set the minimum required length and maximum gap between the two lines. After passing all the required parameters to the algorithm, the probabilistic Hough Line Transformation returns both endpoint coordinates of the detected line segments. For self-driving cars, finding accurate line segments is necessary as all the actuation relies on this step. Hence, fine tuning these parameters is absolutely critical in order to get an optimum result out of the algorithm. After set-

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

ting accurate parameters, line segment disclosed image is obtained as shown in Fig. 8after passing the edge detected image of green taped lane line through Probabilistic Hough Line Transformation Algorithm.

Fig. 8. Line Segment Disclosure through Probabilistic Hough Line Transformation

Lane Line Formation

When the Hough Line Transformation is done, end coordinate points of the line segments (x1, y1) and (x2, y2) are handed over by the algorithm. It can be observed through Fig. 8that each lane has a significant number of line segments associated with it. For an effective lane detection, working with one line segment for each lane is essential both in increasing accuracy of the algorithm as well as memory saving. Thus, converting these miniature line segments of each lane line into one is the next step of the Insighting algorithm. As each line segment has a distinct characteristic, in order to categorize them as the left and the right lane line there is one thing that is common between them, and it is their slope. It is observed that every line segment on the left lane has an upward sloping positive slope while the line segments on the right lane have a downward sloping negative slope. Now based on the slope, every line segment can be categorized into two parts; Left Lane and Right Lane. However, if a line segment has positive slope and is in the left part of the image only then it is categorized as a left line segment. Similarly, if a line segment has a negative slope and is situated in the right part of the image, it is categorized as a right line segment. It is done in order to avoid any errors in the process to achieve smooth functioning of the algorithm as it can be seen from the Fig. 8that the top left part of the left lane has a negative slope apart from being on the left side. After all the line segments are categorized into left and right. The next step of Lane Line Formation is to take a mean of the slope and intercept of all the line segments in one group thus obtaining an average slope and intercept of each group. After obtaining average slope and intercept of left and right lane lines, they are converted back to line segments using reverse plotting and the final two lane lines are obtained. The final lane line detected image can be seen from the Fig. 9.

Fig. 9. Webcam Feed (left) and Lane Line Detected Image (right)

Steering Calculations

Once the required lane lines are detected, bounding of the vehicle within those lane lines is essential. To do that, the steering angle is required to be calculated with every changing lane line. Accurate steering angle is required to keep the car in the middle of the detected lane lines.

Heading Estimation

Before calculating steering angle, an estimation of where the car must head-on the next is essential. There are two major cases to look for in the heading estimation of the car.

a. Two Lane Lines

When there are two lane lines captured by the webcam, calculating heading estimation becomes much easier. Assuming the camera is mounted rigidly at the center of the car and the near (bottom) endpoint of the heading line is always in the middle and pointing onwards, heading direction can be calculated by taking a mean of the far (top) endpoints of the lane lines. In Fig. 10, the thin red line is the heading line of the car and shows where the car is headed from the given frame.

b. One Lane Line

As discussed earlier, faulty logics or lane crooks on one side can result in the capturing of one lane line by the webcam. During such a case, it is not possible to determine the heading estimation by averaging two endpoints. When only a left lane is detected, it is assumed that the car is turning hard towards the right and vice versa. Hence, one alternative to the issue is to follow the single lane line i.e., the slope of the heading line must be the same as the detected lane line. In doing so, the car undergoes hard steer to one direction up to the moment where two lane lines are again detected by the webcam thus continuing the process. Fig. 10 shows how the track undergoes a sharp right turn. Hence, by following the lane line (making the slope of the heading line equal to the lane line) it becomes easier to overcome the challenge before the second lane is again detected.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 230-242
Published Online June 2023 in IJEAST (http://www.ijeast.com)

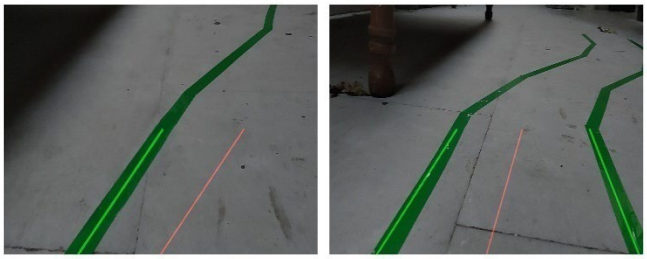

Fig. 10. Heading Line during One Lane Line (left) and Two-Lane Lines (right)

Steering Angle Calculations

Once the heading estimation is done, the penultimate step of the lane navigation algorithm is to calculate the angle at which the steering wheel needs to turn for maintaining the car bound to the center of the lane lines. Steering Angle can be calculated using a simple trigonometric equation. It is known that the bottom end of the heading line is at the middle of the frame and the angle is to be calculated between the heading line and the vertical axis passing through the bottom endpoint of the heading line. Thus, angle is calculated using tan. An inverse tan of X offset (Perpendicular) by Y offset (Base) of a formed triangle given in Fig. 11yields the required steering angle for the next frame.

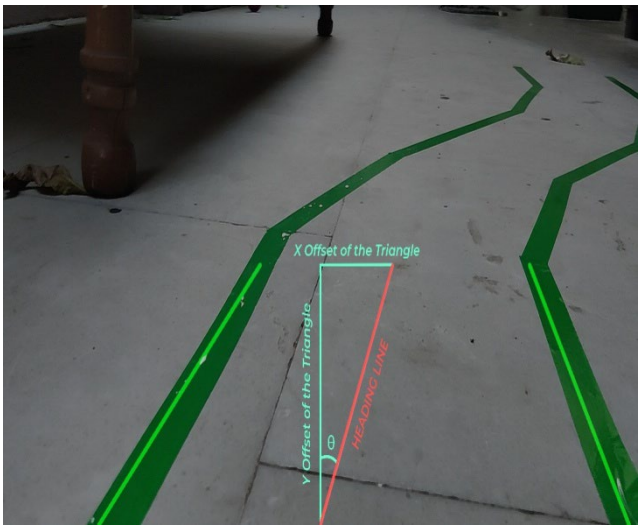

Fig. 11. Steering Angle Calculations

The servo motor mounted on the axle of the has an offset of 90 degrees. That means when the car is heading straight the steering angle is set 90 degrees. Every car in the world has different steering offsets and maximum steering angle limit parameters to be set before the actuation of the steering wheel. This car has a maximum turning angle of about 30 degrees on both sides making up a total of 60 degrees steering field to work with. Thus, the car turns between 60- and 120-degree steering angles.

Steering Angle Stabilization

After steering angle is calculated, one way is to directly feed it to the servo motor for actuation. But by doing so, the car would animate left and right when a sharp change in the steering angle is detected. For example, if the car is heading straight and there is a sudden turn of 110 degrees (to the right), the car would bounce out in the right direction. When this happens frequently (during successive sharp turns), the car slips off the track and cannot detect further lane lines. So instead of a discrete steering angle curve i.e., having 90-degree angle at one millisecond and 110 at the other, a continuously stable steering angle curve is to be formed. To do that an algorithm is set up that continuously tracks any change in steering angle. If the upcoming steering angle differs more than 5 degrees from the current steering angle a gradual 5-degree change would be implemented to reach the desired angle. For example, if the current angle is 90 degrees and the car can see that the upcoming angle is to 110 degrees so instead of jumping to 110 degrees, the car would gradually increase the angle from 90, 95, 100, 105 and finally to 110 degrees. This stabilization helps in elimination of any unwanted off-track slip and zigzag driving of the car. The final obtained Stabilized angle is now passed on to the command actuation department of the car for the desired inclination of the steering wheels.

Command Actuation

Once the steering angle is calculated and stabilization is done, the new stabilized angle has to be fed to the Servo motor by the Arduino. Arduino works over serial communication and fetches its data from the Raspberry Pi. The main challenge to be overcome was communicating the Arduino with the RPi, which has been done by implementing 'Pyfirmata' library. It is a prebuilt library package of python that allows serial communication between a python script and Arduino. Thus, by using 'Pyfirmata', a python program can be run on the Arduino via Raspberry Pi as it acts as the host for Arduino.

The Arduino acts as the command center of the car as it now gives commands to the Servo by feeding the new angle after every frame has been processed as well as providing signals to the motor driver in order to control the speed of the car as the rear wheels are coupled to the dc motor that is integrated with the motor driver. The Arduino also has the hold over the car's braking system. Braking System is basically an automatic shutdown feature where when the car does not detect any lane lines, the power to the motor gets cut off making the car stride forward with inertia for a couple of seconds. Even for that buffer amount of time if the lane lines are not detected, then the car comes to a force stop by applying reverse potential to the motors and stopping them.

Testing

The algorithm is tested on two different sets of parameters for each image i.e., rhom angle, minThreshold, minLineLength, and maxLineGap.The first training set of the algorithm was completed by passing the parameters such as minThreshold, minLineLength, and maxLineGap as 1, π/180, 20, 10, and 5

respectively and the second training set was completed by passing parameters as 1.5, π/180, 40, 20, and 10 respectively. The car was tested on the track multiple times and was observed that the car performed better on the set-2 parameter having an accuracy (the percentage times the car successfully completed entire track without deviating) of 95% as compared to 90% using set-1 parameters.

V. CHALLENGES AND FUTURE SCOPE

Almost every readily available remote-control car in the market comes with a DC motor coupled directly to the front wheel axle of the car. The problem with this setup is one cannot set the car's steering angle precisely in the degrees as the motor is coupled in such a way that it only deviates to the extremities. To avoid that, the Servo motor shaft was mounted directly to the front wheel. Apart from the fact that the 3D printed mounting was quite complex in design, the Servo motor precisely turned the steering wheel to the given output by the microprocessor. This setup helped the car not only improvise its precision but also helped the car achieve high frequency steering actuation in the case of tight corners. Another drawback of this system is at night when there is not enough light for the camera to capture the information from the world. When the sun sets, the algorithm fails drastically with a steep decrease in the overall efficiency. Usually while an image is captured, all the light from the spectrum is captured by the camera. This includes the light from the infrared spectrum which is responsible for diminished contrast, abnormal color tone, and other image quality issues. To avoid such image quality issues, infrared light is needed to be filtered out before passing through the lens and it is done through an infrared filter attached with the camera. This infrared filtered enabled camera is good for capturing images during the day but at night, this infrared filter filters out all the light and completely blackens an image. Using a camera without infrared filter alongside the normal camera solves the problem as during the night the camera without infrared filter captures the infrared light carrying some information on it. This little information capture is sufficient enough to detect lanes and carry out the algorithm. As the lane line used in this research is green, the car detects and remains bound to the green lane line only. This setup takes a huge setback when it comes to real life problem solving as the lane lines on the roads of the world are anything but green. Problems can be solved by manually changing the hue, saturation and value parameters according to the required color matching to the road. But this solution is time consuming and moreover with every changing road condition the parameters need to be altered. Research at Nvidia has shown excellent results in dealing with this issue. They have come up with a supervised machine learning algorithm that works on Convoluted Neural Networks (CNN) which considers video images as features and steering angle as labels to train the model. This model is trained based on the given input-output pairs provided by the user. A specific image and steering data of a pre-existing track with various parameters is to be fed to the model and it trains itself based upon the scenario. This improves the code efficiently and avoids the hassle of manually overriding all the parameters for different algorithms at each contrasting track with a little increase in the processing power.

In order to amplify things further and make the car more leaned towards the real world, an end-to-end deep learning algorithm can be implemented where the car teaches itself using a deep learning algorithm how to drive autonomously on the road just by observing a remote-controlled driving patterns of a human being. By doing so, all the hassles of color transformation, edge detection, and Hough transformation can be avoided and more efficient autonomous driving can be set up using fewer lines of code. This method of training a model is called supervised learning. In supervised learning, machine produces output based on the learnings from trained input-output pairs. In this case, a few laps around a predetermined track to be run with a joystick controller helps the car learn all the patterns of human behavioral driving and based on those patterns it learns and simulates the results itself. With supervised learning, few codes can powerfully impact how autonomous cars can efficiently imitate human-like driving. Most modern self-driving cars use either LiDAR or ultrasonic sensors for distance detection. Same thing can also be implemented on an autonomous RC car; Using either LiDAR or Ultrasonic sensors, the car can easily determine distance in front of it. This exponentially helps in replicating how real-life Teslas and Mercedeses of the world implements adaptive cruise control systems in their Electric Vehicles. A LiDAR sensor using its pulsed light waves scattered in all directions helps in developing a virtual 3D environment around itself. Using this 3D environment, a car can visualize where, how and how far the other cars behave around itself. Using these technologies, an adaptive cruise control system can be implemented in the autonomous RC car that deviates, brakes and accelerates automatically by observing where and how far the other objects are within its environment. Thus, making the car behave much more like a modern autonomous car of the world. Implementing an IoT based telemetry system is also another feature that can be implemented for data actuation purposes. Most modern autonomous cars of the world have an IoT based app that tracks the movement of the car, helps the user to control the car remotely through a mobile phone, and it also helps an individual to access its cameras in case of a theft or emergency. All the data of the car from battery capacity to camera feed would easily be accessible to an individual residing anywhere in the world. A personal virtual assistant can also be installed in the car that can access each parameter through voice commands of individuals. A person with his voice would be able to either start or stop the car using voice commands. Using features like supervised learning, LiDAR, IoT based Telemetry and Voice Assistant, a closer and more advanced versions of modern autonomous cars of the world could be portrayed making any individual or a growing student understand the technology better with each progressing step.

VI. CONCLUSION

This paper aimed to provide a cheaper, effective, and simpler way to employ autonomous systems in cars by only implementing computer vision while avoiding the use of other expensive sensors such as LiDAR. Keeping cost effectiveness in mind, the whole project has been performed over a pre-build RC car. The paper also provides a detailed comparison between different papers regarding how they have implemented self-driving cars, their limitations and what scope they have in the near future, compared to what this paper suggests. The paper initially discusses the assembly of the RC car, how it was dismantled and what all changes were then introduced in order to employ all the accessories necessary for the project. The front wheel assembly has been completely changed as Servo motor replaced the DC motor coupled to the wheels. Secondly a power bank is substituted for the Ni-Cd batteries. RPi along with Arduino perform all the processing and actuation parts respectively. The camera input is fetched to RPi where it first performs color abstraction followed by edge detection. Once all the edges have been detected, the focus is laid over the ROI that contains the important information for further processing. Now, Hough Line transformation is performed that detects all the line segments in the ROI using which the track lines have been calculated. After the track has been detected, steering angle is derived by using a self-developed algorithm and stabilization is performed to prevent sudden strain over the servo motor. This angle is then transmitted to the Arduino that actuates the servo and DC motor coupled to the motor driver accordingly. This paper has achieved the aim to implement an autonomous system with the help of Computer Vision only, without using external sensors. Although there are some limitations to this model which can be rectified by implementing advanced technologies, this model has provided sufficient evidence that by introducing few changes, this method can be adopted for self-driving cars in future.

VII. REFERENCES

[1]. Maksimovic, M., Vujovic, V., Davidović, N., Milosevic, V., & Perisic, B. (2014). Raspberry Pi as Internet of Things hardware: Performances and Constraints.

[2]. Md Jan, M., Zainal, N., & Jamaludin, S. (2020). Region of interest-based image retrieval techniques: A review. IAES International Journal of Artificial Intelligence (IJ-AI), 9(3), 520. https://doi.org/10.11591/ijai.v9.i3.pp520-528

[3]. Newman, J., Sun, Z., & Lee, D.-J. (2020). Self-Driving Cars: A Platform for Learning and Research. In 2020 Intermountain Engineering, Technology and Computing (IETC). 2020 Intermountain Engineering, Technology and Computing (IETC). IEEE. https://doi.org/10.1109/ietc47856.2020.9249142

[4]. SinghPannu, G., Dawud Ansari, M., & Gupta, P. (2015). Design and Implementation of Autonomous Car using Raspberry Pi. In International Journal of Computer Applications (Vol. 113, Issue 9, pp. 22–29). Foundation of Computer Science. https://doi.org/10.5120/19854-1789

[5]. Barua, B., Gomes, C., Baghe, S., & Sisodia, J. (2019). A Self-Driving Car Implementation using Computer Vision for Detection and Navigation. In 2019 International Conference on Intelligent Computing and Control Systems (ICCS). 2019 International Conference on Intelligent Computing and Control Systems (ICCS). IEEE. https://doi.org/10.1109/iccs45141.2019.9065627

[6]. Yaovaja, K., Bamrungthai, P., & Ketsarapong, P. (2019). Design of an Autonomous Tracked Mower Robot using Vision-Based Remote Control. In 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE). 2019 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE). IEEE. https://doi.org/10.1109/ecice47484.2019.8942741

[7]. Secuianu, F.-D., & Lupu, C. (2018). Implementation of a home appliance mobile platform based on computer vision: self-charging and mapping. In 2018 22nd International Conference on System Theory, Control and Computing (ICSTCC). 2018 22nd International Conference on System Theory, Control and Computing (ICSTCC). IEEE. https://doi.org/10.1109/icstcc.2018.8540685

[8]. Tian, H., Ni, J., & Hu, J. (2018). Autonomous Driving System Design for Formula Student Driverless Racecar (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1809.07636

[9]. Devos, A., Ebeid, E., & Manoonpong, P. (2018). Development of Autonomous Drones for Adaptive Obstacle Avoidance in Real World Environments. In 2018 21st Euromicro Conference on Digital System Design (DSD). 2018 21st Euromicro Conference on Digital System Design (DSD). IEEE. https://doi.org/10.1109/dsd.2018.00009

[10]. Grinke, E., Tetzlaff, C., Wörgötter, F., & Manoonpong, P. (2015). Synaptic plasticity in a recurrent neural network for versatile and adaptive behaviors of a walking robot. In Frontiers in Neurorobotics (Vol. 9). Frontiers Media SA. https://doi.org/10.3389/fnbot.2015.00011

[11]. de Coelho, L. S., Campos, M. F. M., & Kumar, V. (n.d.). Computer vision-based navigation for autonomous blimps. In Proceedings SIBGRAPI'98. International Symposium on Computer Graphics, Image Processing, and Vision (Cat. No.98EX237). SIBGRAPI'98. International Symposium on Computer Graphics, Image Processing, and Vision. IEEE Comput. Soc. https://doi.org/10.1109/sibgra.1998.722762

[12]. Alexander Yur'evich, K., Fedor Valer'evich, D., Sergeevich Ivanov, V., Sergeevich Yegorov, A., &

Viktorovich Men'shikov, V. (2018). Methods for the Preparation of Modified Polyorganosiloxanes (A Review). In Oriental Journal of Chemistry (Vol. 34, Issue 2, pp. 612–622). Oriental Scientific Publishing Company. https://doi.org/10.13005/ojc/340202

[13]. Manigel, J., & Leonhard, W. (1992). Vehicle control by computer vision. In IEEE Transactions on Industrial Electronics (Vol. 39, Issue 3, pp. 181–188). Institute of Electrical and Electronics Engineers (IEEE). https://doi.org/10.1109/41.141618

[14]. Utesch, F., Brandies, A., Pekezou Fouopi, P., & Schießl, C. (2020). Towards behaviour based testing to understand the black box of autonomous cars. In European Transport Research Review (Vol. 12, Issue 1). Springer Science and Business Media LLC. https://doi.org/10.1186/s12544-020-00438-2

[15]. Lindholm, M. (2013). Urban freight transport from a local authority perspective-A literature review. European Transport - TrasportiEuropei, 54.

[16]. Li, B., Zhang, T., & Xia, T. (2016). Vehicle Detection from 3D Lidar Using Fully Convolutional Network (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1608.07916

[17]. Shen, Y.-T., Chen, L., Yue, W.-W., & Xu, H.-X. (2021). Artificial intelligence in ultrasound. In European Journal of Radiology (Vol. 139, p. 109717). Elsevier BV. https://doi.org/10.1016/j.ejrad.2021.109717