IMPLEMENTATION OF DRIVERLESS CAR

Project Reference No.: *45S_BE_4553*

College ***:*** *Reva University, Bengaluru*
Branch ***:*** *Department of Electronics and Communication Engineering*
Guide(s) ***:*** *Prof. Md. Tauseef*
Student(S) ***:*** *Ms. Alisha Fatima*
Ms. Ankitha Kavya Gowda
Ms. Bhavitha Y
Ms. Chandana C U

Keywords:

Driverless Car, Donkey Car, TensorFlow, Python, CNN (Convolutional Neural Network), OpenCV

Introduction:

There has been a large spike in the number of automobile accidents these days. 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. Hence, the number of deaths in 2020 is more than the number of deaths in the last 5 years.

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 the research and development of semi-autonomous and fully autonomous cars. Some prominent companies that are currently developing and/or manufacturing autonomous cars are given below along with the name of their automation technologies:

1. Tesla – “Autopilot Mode” and “Full-Self Driving Mode” are currently available in all their car models.
2. Waymo – It has been developing “Self-Driving Automobile Technology”, a project formerly known as “Google Autonomous Automobile Project”.
3. Apple – It is working on self-driving technology under the name of “Project Titan”.
4. Audi – Their automation technology is called “Autonomous Intelligent Driving”.

Objectives:

Our main objective is to implement a Driverless Car to perform the following actions:

1. To detect and identify traffic lights and road signs.
2. To detect and avoid the obstacles, pedestrians, and other vehicles.
3. To record user inputs like steering angles, throttle levels and video inputs.
4. To train the DonkeyCar deep learning model using the user inputs.
5. To travel to a set location without any accidents and without any human intervention.

Hence, a driverless car will be very useful for aged or physically challenged people as they can travel from one place to another independently and safely

Methodology:

The Working of Drowsiness Detection System Can Be Divided into Block Diagram And The Software Working.

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.

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.

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. 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.

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.

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.

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.

The working of the Driverless Car can be explained using two flowcharts that explain the two phases in making it. 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 three inputs from the data are taken: 1. The image. 2. Steering Angle 3. 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 Convolutional Neural Network 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.

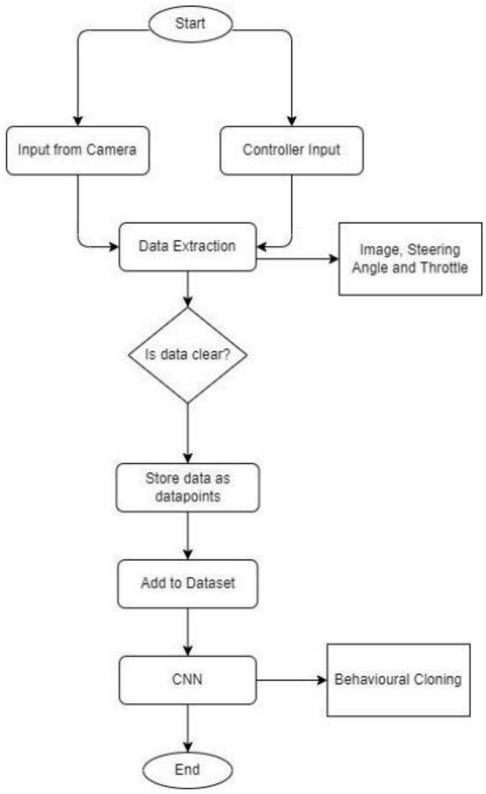

Fig 1: Flow diagram for the Training Phase

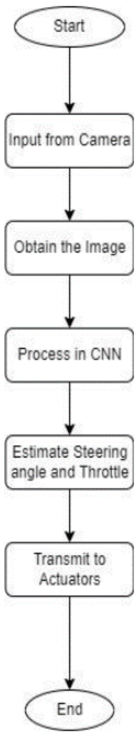

Fig 2: Flow diagram for the Trained Phase

Conclusion:

In our project we have implemented a driverless car by making a demonstration model. We have used a Raspberry Pi 4 Model B as the main processor. It is attached to a camera which 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 clockwise direction and the remaining 10 were completed by driving it in anti-clockwise direction. We needed to do this because, for the first model we trained, we had only driven it in 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 lane in Local Angle mode. It is able to complete the entire track without any human intervention.

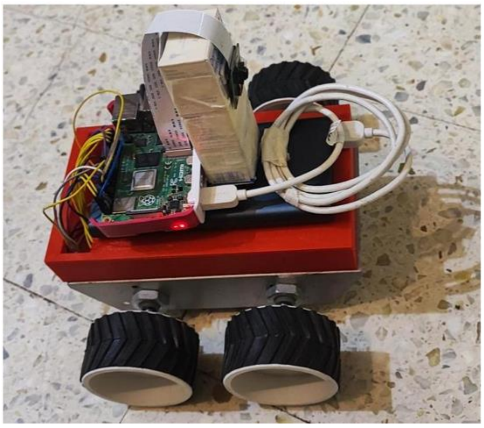

Fig 3: Model of the Driverless Car

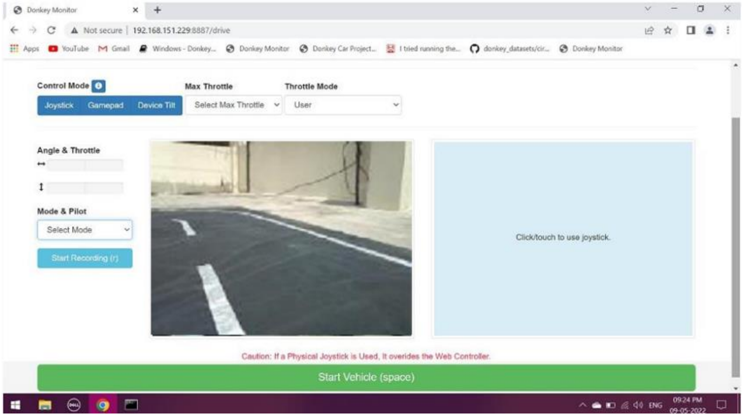

Fig 4: Donkey Monitor

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 which 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. 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 wasn’t getting stored in the database.

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.

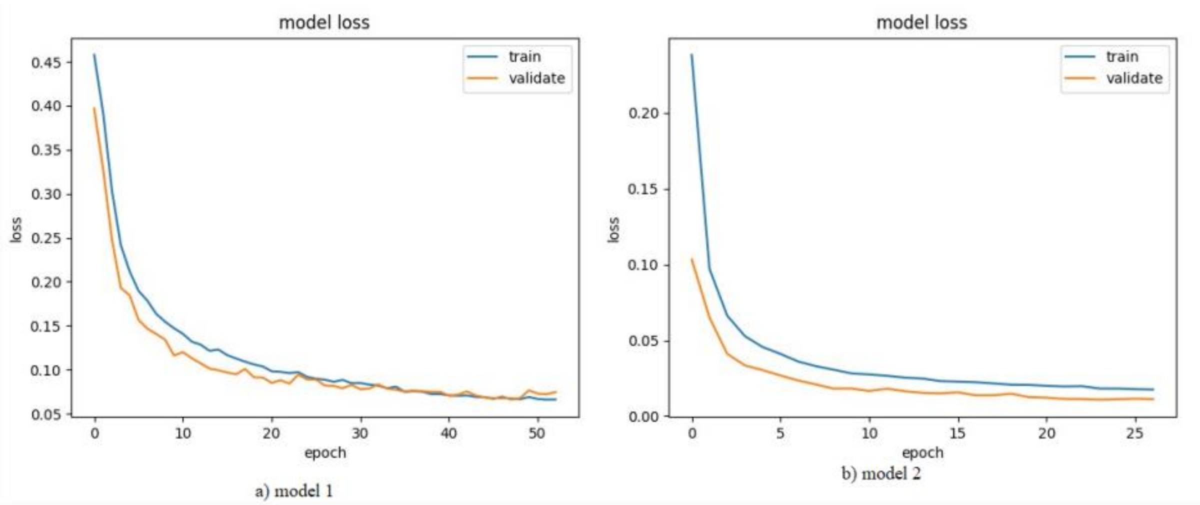

Fig 5: Graphs of model loss per epoch

Scope for future work:

Driver error is the most common cause of traffic accidents. The autonomous vehicle is one of the greatest technological developments of the next decade. It will increase productivity, especially for people who work from the road (e.g., real estate, sales). Donkey is a Self-Driving Car Platform, made up of several components. It is a high-level self-driving library written in Python. 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 web controller/physical joystick controller. As driving car becomes reliable, we used Keras to train a neural network to drive like a human.

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. This can be used to estimate travel time which in turn will help us save time.
2. LIDAR System can be added. Light Detection and Ranging, or LIDAR is a technology in which single light particles or photons are emitted. These Photons strike surrounding objects and are reflected back. Based on the time after which they come back to the sensor, the distance of the objects from the vehicle can be calculated. This would drastically reduce collision rates.
3. We can include another Camera directed backwards, for parking assistance. This would be needed to train the model for driving in reverse.