International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 09 Issue: 04 | Apr 2022 www.irjet.net p-ISSN: 2395-0072

Self Driving Car

Kavya Negi[1], Kunj Patel[2], Janhavi Satam [3]

[1,2,3]UG Student, Department of Information and Technology Engineering, Atharva College Of Engineering, Maharashtra, India

---***---

Abstract – *The evolution of Internet of Things has served up the catalyst in the field of technology. Automobile manufacturers such as Ford, Audi, Hyundai, Tesla, and other companies are investing billions of dollars in autonomous vehicle driving research. According to the new information, in the next 30 years, this fast-developing industry will be worth $ 7 trillion. This will create a shift on the way cities are planned, as less parking spots will be needed, and secondly, in a developed city with most of its vehicles being connected and autonomous, traffic optimization will be able to be strongly applied by coordinating movement of the vehicles.*

1. INTRODUCTION

Over the last years the automotive industry has advanced significantly towards a future without human drivers. Researchers are currently trying to overcome the technological, political and social challenges involved in making autonomous vehicles mainstream. These vehicles need to be safe, reliable and cost-efficient. Connecting them and creating coordination mechanisms could help achieve these goals. This project proposes a self-driving Radio Control car. This car runs in an artificial environment. The environment consists of various real life obstacles, traffic signals, speed boards, etc. The car has to run and adjust according to the environment. This project focuses on lane driving, obstacle detection and road signs. This is a small scale representation of the self-driving car using various Deep Learning, Computer Vision techniques. With the increase in the size of the model the complexities increase as well. The project focuses on basic driving features considering the safety and cost of the model.

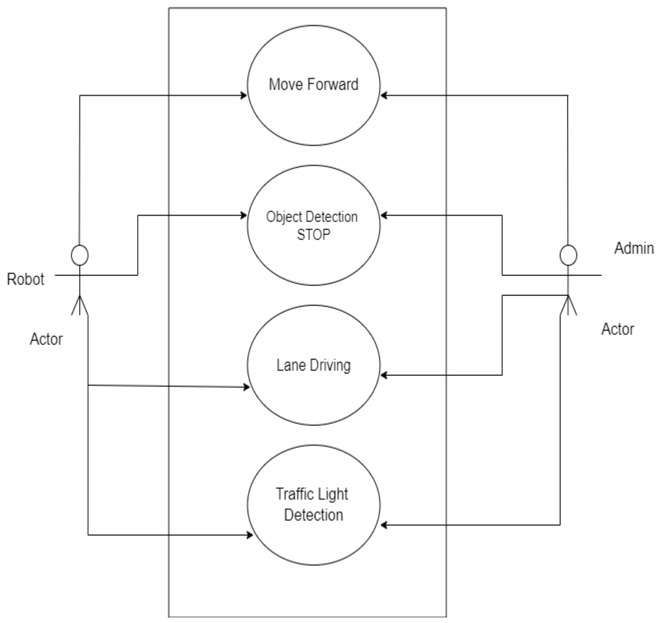
2. LITERATURE REVIEW

Mohammed A.A. Babiker et al.[1] made use of the Unity platform to build thesimulator of the car and its environment. The model uses 2 Neural Networks viz Convolutional Neural Network and VGG16 model. CNN algorithm was used to predict the output of the steering angle and VGG16 model was used to detect the traffic lights. Initially the data was collected by driving the car manually and the data obtained was used to train the neural networks. The camera collects the image which in turn acts as an input to the CNN algorithm. The predicted values help the car to move accordingly. The model had an accuracyof 86% and was able to detect different road features.

Uvais Karni et al.[2] used Image processing, Deep learning, Raspberry Pi, Arduino to build a small-scale version of an autonomous RC car. CNN was chosen by the author becauseof its high efficiency and low time taken to build the model. Neural network and Random Forest were so that the model car would run without any errors after being trained on all sets. Raspberry Pi camera captures the image and the neural network is loaded on the same Raspberry Pi Microcontroller. The model is trained several times using CNN. Sigmoid and cost functions were used to minimize the errors in the model for smooth functioning of the car. Since they loaded the model on the Micro controller itself, The future scope of this project will be loading the model on a Laptop/PC for faster outputs.

Jaychand Upadhyay et al.[3] has built a simulation of Autonomous driving car using aunity gaming module. Deep Q learning instead of Deep learning was used, Deep learning using CNN gives output as a classification whereas Deep Q learning gives the output as Q points. Q points are given when the model is built and tested successfully, if it throws some error then we get Q points in negative. The model always tries to attain a high number of Q points. The future scope of this model is implementing the same model in a real small scaled RC car so that it can be used in the future.

3. PROPOSED BLOCK DIAGRAM



The RC Car is trained in a virtual environment. The environment is created by taking into consideration the real life situations. The car is trained many times to get higher accuracy.The Pi Camera attached on top of the car captures the images. The images are processed by Raspberry Pi Microcontroller.

Image pre-processing is done using OpenCV. The image input is then taken and feature extraction takes place using CNN algorithm. The images are pooled pixel by pixel manner. This gives higher accuracy to the model.

After pooling the images are classified and divided into classes, the final output is obtained. The output is then sent to the Arduino Microcontroller. The microcontroller reads the NN output and simultaneously triggers the car according to the output. The car moves according to the signal received from Arduiono.

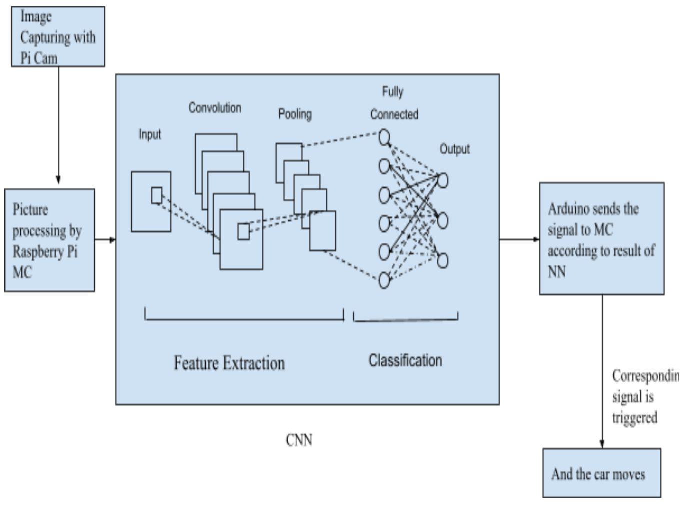
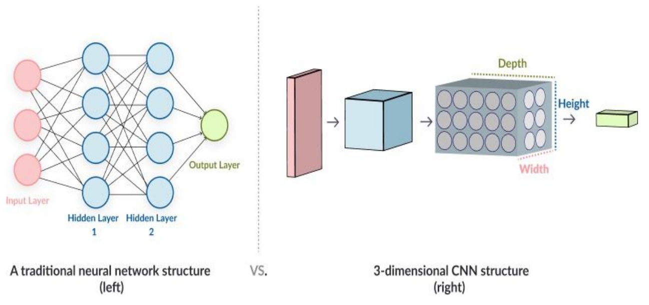


Image Capturing with Pi Cam
Picture processing by Raspberry Pi MC
Input
Convolution
Pooling
Fully Connected
Output
Feature Extraction
Classification
CNN
Arduino sends the signal to MC according to result of NN
Correspondin signal is triggered
And the car moves

4. IMPLEMENTATION

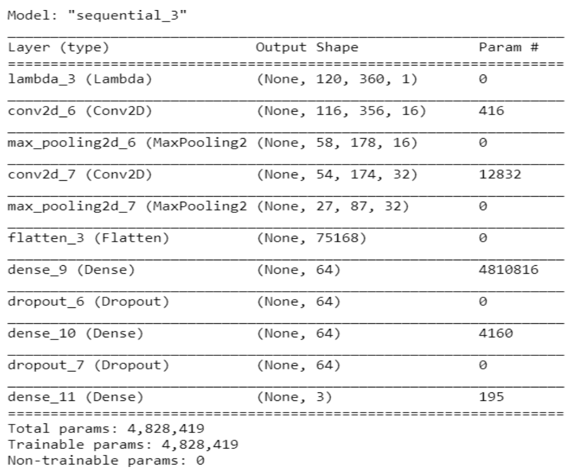
4.1 Algorithms

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithmwhich can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. . The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.


Input Layer
Hidden Layer 1
Hidden Layer 2
Output Layer
Depth
Height
Width
A traditional neural network structure (left)
VS.
3-dimensional CNN structure (right)

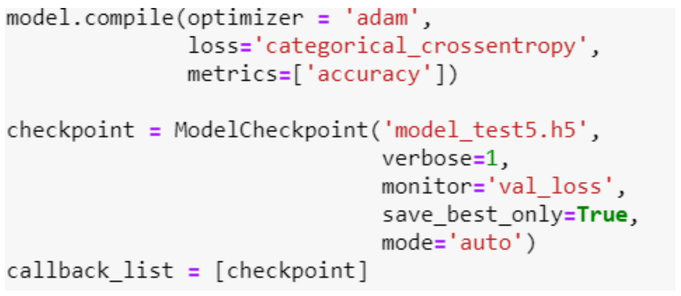
4.2 Working of the Project

- First step of our project was to load the training data. The dataset consists of 67 images of the environment that was created artificially. It has several images of roads that go straight, towards the left and towards the right.
- The training data is stored as an npz file (Numpy array zip) and is given as an input tothe Model.
- We have used the sequential model that CNN provides. 'Relu' and 'Softmax'activation functions were used.



```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lambda_3 (Lambda)            (None, 120, 360, 1)       0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 116, 356, 16)      416
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 58, 178, 16)       0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 54, 174, 32)       12832
_________________________________________________________________
max_pooling2d_7 (MaxPooling2 (None, 27, 87, 32)        0
_________________________________________________________________
flatten_3 (Flatten)          (None, 75168)             0
_________________________________________________________________
dense_9 (Dense)              (None, 64)                4810816
_________________________________________________________________
dropout_6 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_10 (Dense)             (None, 64)                4160
_________________________________________________________________
dropout_7 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_11 (Dense)             (None, 3)                 195
=================================================================
Total params: 4,828,419
Trainable params: 4,828,419
Non-trainable params: 0
```

To train the model we used 'adam' optimizer.

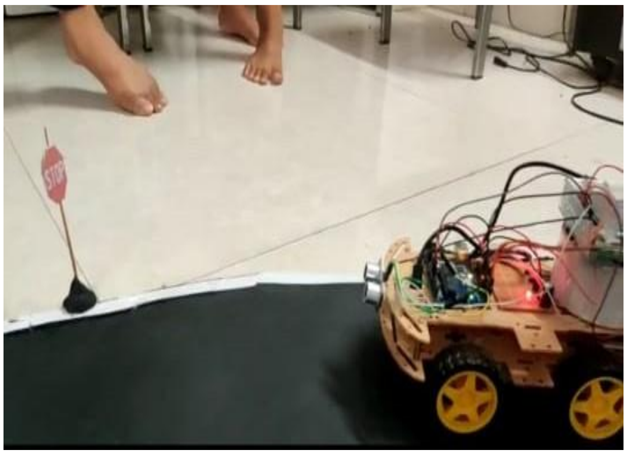


```
model.compile(optimizer = 'adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

checkpoint = ModelCheckpoint('model_test5.h5',
                             verbose=1,
                             monitor='val_loss',
                             save_best_only=True,
                             mode='auto')
callback_list = [checkpoint]
```

- After 50 epochs our model was ready to be tested.

- In order to run the Self driving car, we have used 3 Server socket connections which will collect the data from raspberry pi. They are:
  - A connection where the raspberry pi acts as client and the computer acts as a host. This is done to send the image stream that is captured from the Raspberry Pi camera.

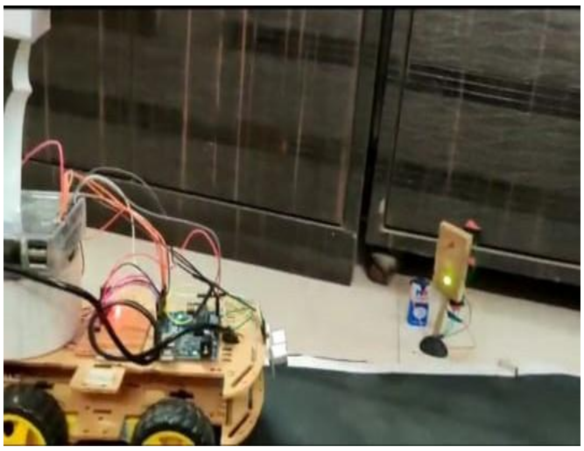


  - A connection where the computer acts as a client to the raspberry pi. Once the image stream is loaded in the computer, the image is sent to the trained model we have made (shown above). The predictions from the computer are sent to the raspberry pi.
  - A connection where the raspberry pi acts as a client to the computer in order to send the details that are collected by the Ultrasonic HC-SR04 sensor.

4.3 Implementation of Features:

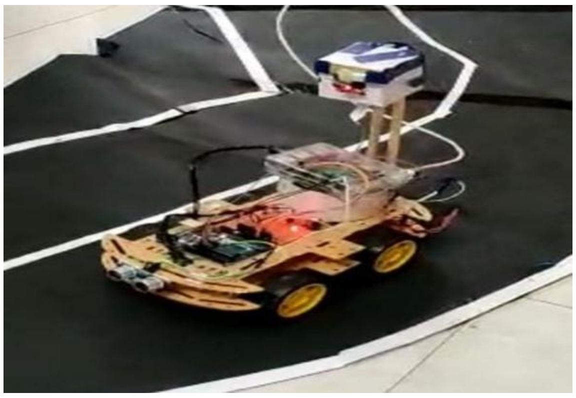
- **Obstacle detection:** Using the Ultrasonic HC-SR04 sensor the car detects the obstacle at the distance of 30-40cms and stops for 2 seconds and then changes its lane and moves forward.

- **Traffic Signals & Road Sign Detection:** In order to detect the signals wehave a pre-trained cascade classifier model in OpenCV. A cascading classifier detects a frame as positive or negative so each classifier will check each framefor positive samples, if the sample is not found the sliding window moves to the next frame. The next frame is checked again for a particular classifier. Thisgoes on till the entire frame of the video is covered.



- **Lane Detection:** A trained Convolutional Neural Network model is used to predict the next move of the car. If the road is moving towards the left, the prediction given by CNN will be a left turn. During a left turn, the Model sees whitespace of the floor on the right and black chart paper on the left, which triggers a Left turn prediction.

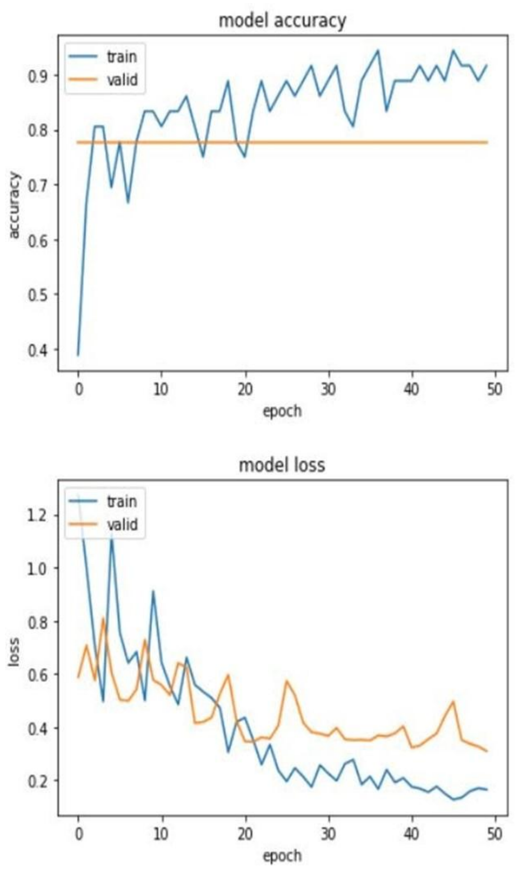
5. RESULTS



The picture depicts the hardware prototype of our self-driving car model made using CNN and IOT devices. The black chart paper here acts as an artificial environment on which the prototype will work abiding by rules that we have stated.

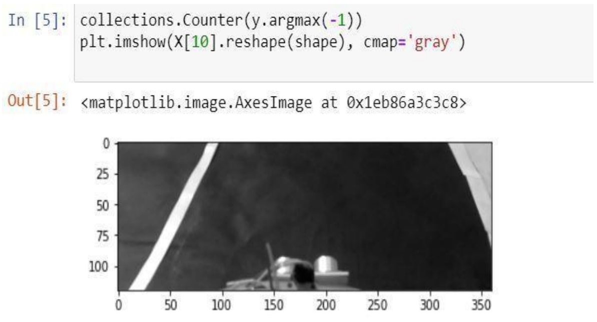
Hardware	Functionality	Voltage Specification
Raspberry Pi 3B+ (with plastic cover)	Used as a chip computer	5 V (>2.1 A)
Raspberry Pi camera v2	Used as a camera to capture Images	Works along with RaspberryPI
DC Motor	Used to rotate the wheels of the car	5V
Arduino Uno	Used as a microcontroller that sends analog signals to DC Motor	5V
Power Bank *2	Used to provide electric supply	Output: 5V 2.4A
Ultrasonic HC-SR04 Sensor	Used to detect the distancebetween the sensor and theobstacle	5V

To automate the car following hardware specifications are required:



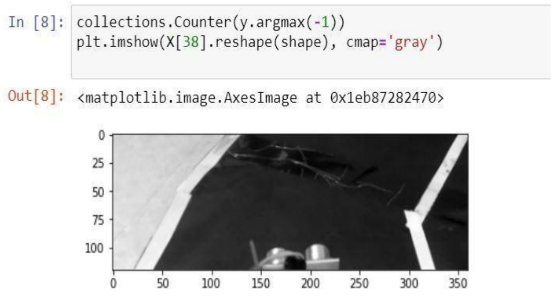
Validation Accuracy

Above is a picture which displays the training and validation accuracy. The training data which consisted of images of the environment was given to the CNN model. The training accuracy increased over time i.e. the model kept on learning on itself in the 50 steps.Validation data was passed over the testing dataset and the validation accuracy was acquired. In the second image both the training and testing errors were less than 0.4 which proves that the model is accurate in predicting the directions in which the car should travel.



```
In [5]: collections.Counter(y.argmax(-1))
        plt.imshow(X[10].reshape(shape), cmap='gray')

Out[5]: <matplotlib.image.AxesImage at 0x1eb86a3c3c8>
```

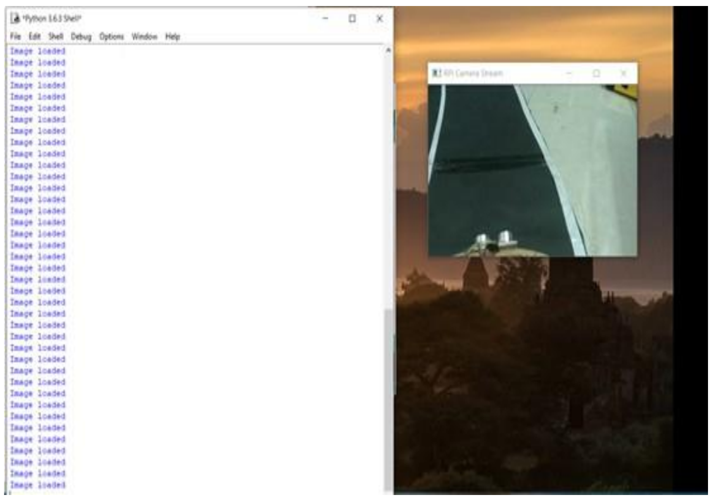


```
In [8]: collections.Counter(y.argmax(-1))
        plt.imshow(X[38].reshape(shape), cmap='gray')

Out[8]: <matplotlib.image.AxesImage at 0x1eb87282470>
```

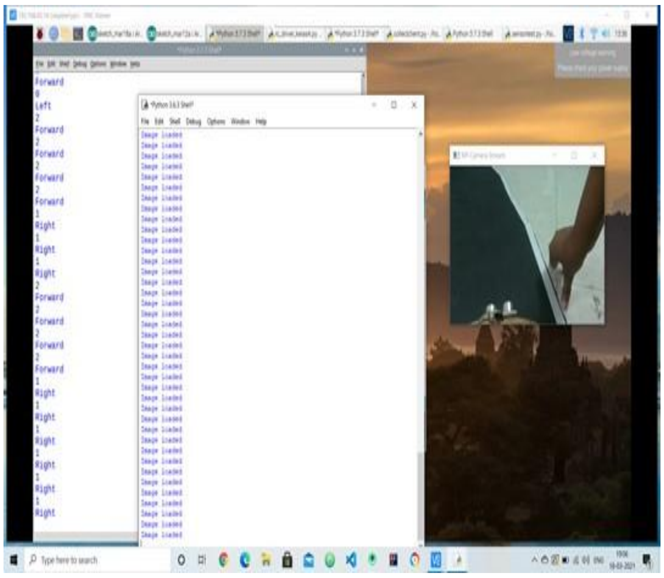
Above two images show an example of what the Raspberry Pi camera records. The first one perceives a straight road whereas the second picture sees the road take a right turn.

The pictures are recorded at every click from the keyboard given to the model.



```
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
```

This figure has two components, the one on the left is the Image being transmitted from the raspberry pi cam to the computer and hence the image loaded notification pop ups in every second. On the right is the normal image stream that the camera has recorded.



```
Forward
0
Left
2
Forward
2
Forward
2
Forward
2
Forward
1
Right
1
Right
1
Right
2
Forward
2
Forward
2
Forward
2
Forward
1
Right
1
Right
1
Right
1
Right
1
Right
1
Right
```

```
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
Image loaded
```

Figure _ is the main implementation of the codes that we have used. Server Sockets are used to transmit data between the Raspberry Pi and the Host computer over the same network.

As you can see, the picture on the right shows the road going straight and then turning left. Inthe same way we can see the model predicting forward first and then a left.(on the top left of the image).

6. CONCLUSION

Self-driving cars are the future of the automobile industry. In our proposed model, implementation of self driving cars can be cost efficient and still give higher accuracy rates. The car is trained to move in a virtual environment which is similar to the real life environment and functions just like the human mind does while driving. Lane driving, lane changing, traffic rules, etc are the major factors this model is focusing upon. This model can be implemented on a large scale though the complexities will increase with the increase in thesize of the project. Hence, a cost efficient model with high accuracy rates is what theproposed model is all about.

7. REFERENCES

[1] Mohammed A.A. Babiker, Mohammed A. O. Elawad, Azza H. M. Ahmed,"Convolutional Neural Network for Self – Driving Car in a virtual Environment", 2019 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCC EEE19).

[2] Uvais Karni, Shreyas Ramachandran, K Sivaraman,"Development Of Autonomous Downscaled Model Car Using Neural Networks And Machine Learning", 2019.3rd International Conference on Computing Methodologies and Communication (ICCMC)

[3] Jaychand Upadhyay, Takafumi okayama, "Autonomous Driving System based on Deep Q Learning" ,2018. International Conference on Intelligent Autonomous Systems(ICoIAS)

[4] Nischal Sanil, Pasumarthy Ankith Naga venkat, Rakesh V, Rishab Mallapur, Mohammed Riyaz Ahmed,"Deep Learning Techniques for Obstacle Detection and Avoidance in Driverless Cars",2020 International Conference on Artificial Intelligence and Signal Processing (AISP).

[5] Bhaskar Barua, Clarence Gomes, Shubham Baghe, 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)

[6] Dong Dong 1 , Xiao-Yang Li2 , XiaoTong Sun3,"A Vision-Based Method For Improving the Safety Of Self-Driving",School of Reliability and Systems Engineering, Beihang University Science and Technology on Reliability and Environmental Engineering Laboratory Beijing, China

[7] Gowdham Prabhakar, Binsu Kailath,Sudha Natarajan, Rajesh Kumar,"Obstacle detection and classification using Deep learning for tracking in high-speed Autonomous driving", IEEE Region 10 Symposium (TENSYMP)

[8] Mohammad Rubaiyat Tanvir Hossain, Md. Asif Shahjalal, Nowroz Farhan Nur,"Design of an IOT based Autonomous vehicle with the aid of computer vision",International Conference on Electrical, Computer and Communication Engineering (ECCE), February16-18, 2017, Cox's Bazar, Bangladesh