SELF DRIVING CAR USING LANE DETECTION SYSTEM

Dr.I.Poonguzhali[1], R.Indira[2], R.Vignesh[3*], V.Yuvaraj[4], N.Suryaa[5]

[1]Associate Professor, [2]Assistant Professor, [3,4,5] UG Scholar, Department of Electronics and Communication Engineering, Panimalar Institute of Technology, Chennai, India

**Corresponding author: maheshvicky001@gmail.com*

ABSTRACT - *The autonomous car or the self driving car is able to sense the environment, navigate & can fulfill the transportation capabilities of human without human input. However, vehicle crashes still holds the leading cause of accident death of millions of people every year. The main reason for building an intelligent vehicle is to elevate the safety conditions by either entire or semi automation of driving tasks. Among these tasks, the lane detection part has an important role in driving assistance systems that acquires information such as lane structure and car's position respective to the lane. Therefore, a system that assists a driver is necessary. One of the main technology involves in these tasks is Computer Vision which is a powerful tool for sensing the environment and has been widely used in many applications by the Intelligent Transportation Systems (ITS).*

Key Words: Self driving car, Lane detection system, Computer vision

1. INTRODUCTION

Deep learning approaches for autonomous driving have been split into two models: 1. Mediated perception uses the driving markers in the image to create a world representation surrounding the car. Items such as traffic signs and obstacles in the road are classified to determine a driving action 2. Behavior reflex directly maps the input data to a driving action [1, 2, 3]. The entire image is converted to steering and velocity commands. Both models have issues such as requiring a multitude of sensors such as GPS, radar, and accurate maps for mediated perception, while behavior reflex methods can be confused by different human decisions made for similar events [4, 5, 6]. A new approach that maps only specific inputs to steering angle and speed is called the direct perception approach which can be considered a mix between the first two autonomous driving approaches.

The organization of the paper is as follows. Section 2 conveys the research description. Section 3 deals with methodology of the project. Section 4 and Section 5 describes the output and conclusion respectively.

2. RESEARCH DESCRIPTION

Autonomous driving is one of the next frontiers for the Deep Learning and machine learning. The purpose of this research was to produce a deep neural network using Tensorflow python framework to correctly identify the current user lane, throttle and the steering angle that drives the car without any input. A successful model should be able to navigate through curves of the lane given that no car blocks the vision of the target vehicle.

2.1 Sequential Neural Network (SNN):

In the deep learning architecture, Neural Networks build high level accessibilities sequentially through their subsequent layers segmentation. When a data is processed as input, at each layer, one mapping within these layers is selected according to a sequential decision process. The resulting model is developed according to a DAG like architecture, so that a path from the source to a destination node defines a sequence of transformations. Instead of assuming global transformations, similar in traditional multilayer networks, this model allows the user to learn a set of local transformations. So it can able to process input data with different characteristics through defined sequences of such local transformations and by amplifying the expression power of the designed model w.r.t a traditional multilayered network [7, 8, 9].

2.2 Tensorflow:

TensorFlow is a free and open-source library software for machine learning developed by the Google Brain team for internal Google use. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Its flexible architecture allows for the easy deployment of computation across a variety of platforms such as CPUs, GPUs and TPUs, and from desktops to clusters of servers to mobile and edge devices [10, 11, 12].

2.3 Raspberry pi:

Raspberry pi is a tiny computer capable of delivering low level machine learning- neural network outputs is used in this project. Real time interfacing and computations are faster for low level neural network problems.

3. METHODOLOGY

Initially, the dataset needed for training the sequential neural network model is collected using the mini model vehicle which has raspberry pi running on it. It collects and saves the data as the user manually drives through the path several times.

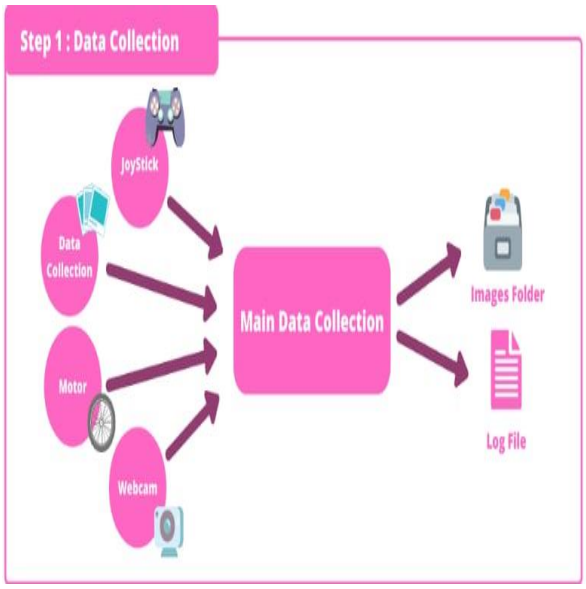

Fig. 1 Step (1) of the process is collecting the required data.

The collected data is stored in two files

(i).Images Folder

(ii).Log file

The Images Folder consist of the collected images named as per the timestamp in sequential order. The Log file has all the steering angle values collected while the user manually drove the vehicle.The steering angle values are also sequential respective to order of the images in the Images Folder. Fig. 1 shows the collection of data in simplest form.

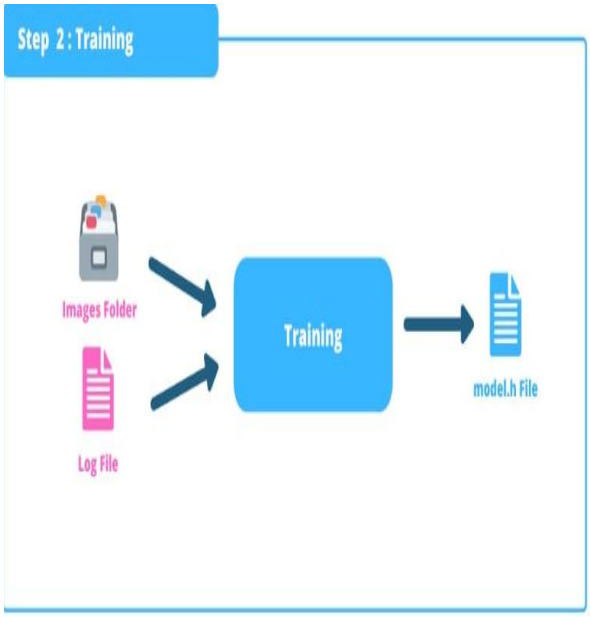

Fig. 2 Step (2) of the process is training the collected data

The collected data is trained using the Sequential Neural Networks Algorithm model. The data is first splitted into two such as

(i) Training set data

(ii) Validation set data.

The training set data consists of nearly 80% of the collected data and the remaining 20% of data is for validating the trained data. Convolutional 2D layer, Flatten layer, Dense layer are added to the model for generating the trained file. Fig. 2 intimates how the collected data will be trained. Fig. 3 will explain about creating the SNN model using ADAM.

Creating a Sequential Neural Network model:

```
#### STEP 7 - CREATE MODEL
def createModel():
  model = Sequential()


model.add(Convolution2D(24, (5, 5), (2, 2), input_shape=(66, 200, 3),
model.add(Convolution2D(36, (5, 5), (2, 2), activation='elu'))
model.add(Convolution2D(48, (5, 5), (2, 2), activation='elu'))
model.add(Convolution2D(64, (3, 3), activation='elu'))
model.add(Convolution2D(64, (3, 3), activation='elu'))


  model.add(Flatten())
  model.add(Dense(100, activation = 'elu'))
  model.add(Dense(50, activation = 'elu'))
  model.add(Dense(10, activation = 'elu'))
  model.add(Dense(1))

  model.compile(Adam(lr=0.0001),loss='mse')
  return model
```

Fig. 3 Adding layers and compiling using ADAM.

Adam method is used for training the required model. Adam is known as an optimization algorithm with adaptive learning rate that has been designed especially for training deep neural networks. It is an adaptive learning rate method, which means, Adam computes individual learning rates for different parameters. Its name is derived from adaptive moment estimation, and it is called as Adam because it uses estimations of 1st and 2nd moments of gradient to adapt the learning rate for each and every weight of the neural network.

The Accuracy and the Validation loss is shown after training the model. The validation loss is computed with the help of the Validation set data. The accuracy of the trained model should be reasonable and the loss should be minimum or else the input data should be changed to reduce the loss.

Once the model is trained and if the accuracy and the validation loss is reasonable then the model is ready to be implemented in action. The live images feed from the integrated webcam is processed through the created model. The model outputs the steering angle based upon the feeded input image. The generated steering angle is used for driving the vehicle to maintain its track in its lane. The accuracy can be improved if more amount of data is collected for training.

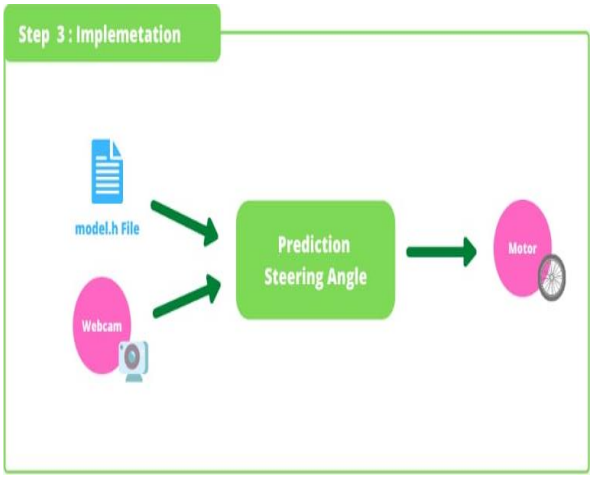

Fig. 4 Step (3) of the process is implementing the trained model.

4. OUTPUT

```
Epoch 1/10
WARNING:tensorflow:Callbacks method `on_train_batch_end` is slow compared to the batch time (batch time: 0.2619s vs `
on_train_batch_end` time: 1.8150s). Check your callbacks.

Epoch 00001: saving model to mnist_digits/
100/100 - 38s - loss: 0.6056 - accuracy: 0.8134 - val_loss: 0.1790 - val_accuracy: 0.9435
Epoch 2/10

Epoch 00002: saving model to mnist_digits/
100/100 - 36s - loss: 0.1892 - accuracy: 0.9431 - val_loss: 0.1056 - val_accuracy: 0.9660
Epoch 3/10

Epoch 00003: saving model to mnist_digits/
100/100 - 36s - loss: 0.1350 - accuracy: 0.9609 - val_loss: 0.0769 - val_accuracy: 0.9749
Epoch 4/10

Epoch 00004: saving model to mnist_digits/
100/100 - 37s - loss: 0.1036 - accuracy: 0.9675 - val_loss: 0.0634 - val_accuracy: 0.9786
Epoch 5/10

Epoch 00005: saving model to mnist_digits/
100/100 - 37s - loss: 0.0981 - accuracy: 0.9716 - val_loss: 0.0598 - val_accuracy: 0.9817
Epoch 6/10

Epoch 00006: saving model to mnist_digits/
100/100 - 36s - loss: 0.0783 - accuracy: 0.9757 - val_loss: 0.0437 - val_accuracy: 0.9848
Epoch 7/10

Epoch 00007: saving model to mnist_digits/
100/100 - 36s - loss: 0.0630 - accuracy: 0.9820 - val_loss: 0.0490 - val_accuracy: 0.9834
Epoch 8/10

Epoch 00008: saving model to mnist_digits/
100/100 - 37s - loss: 0.0636 - accuracy: 0.9814 - val_loss: 0.0430 - val_accuracy: 0.9854
Epoch 9/10

Epoch 00009: saving model to mnist_digits/
100/100 - 37s - loss: 0.0556 - accuracy: 0.9844 - val_loss: 0.0411 - val_accuracy: 0.9864
Epoch 10/10

Epoch 00010: saving model to mnist_digits/
100/100 - 38s - loss: 0.0600 - accuracy: 0.9818 - val_loss: 0.0460 - val_accuracy: 0.9855
training took: 387.0842 secs.
```

Fig. 4 Training Steps with accuracy and validation loss.

The Fig. 4 shows the output process of training the collected data using SNN model. The Figures 5 and 6 is the designed model that we have used in this project.

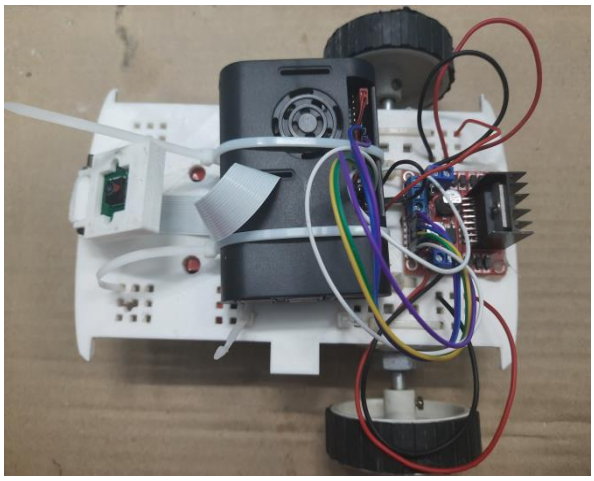

Fig. 5 Designed model of the project

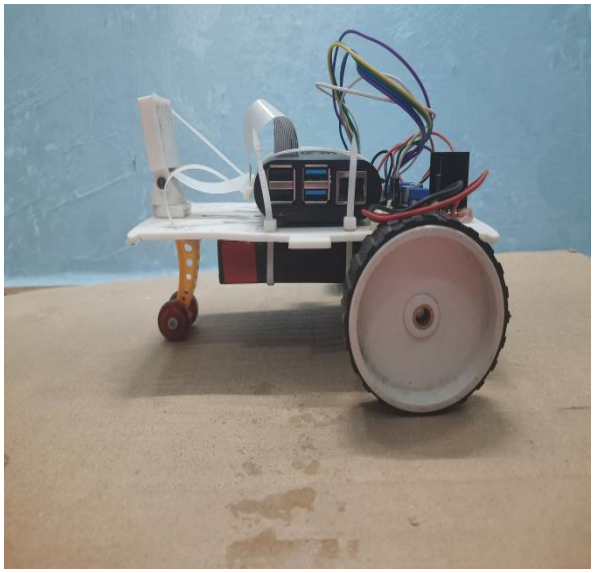

Fig. 6 Designed model of the project

5. CONCLUSION

Thus using deep learning techniques such as SNN, Adam, and tensor flow framework, the research shows that autonomous driving can successfully be simulated in a mini environment using the deep learning architecture presented in this paper. The results of this experiment show that these principles are ready to be applied to a real world platform in order to experimentally verify its ability to become an everyday part of our lives. As autonomous vehicles are embedded with Internet of Things (IoT) data gathering, decision making and interactive driving experience will emerge in future.

REFERENCES

[1] *Apollo: An open autonomous driving platform*, 2018, [online] Available: https://github.com/ApolloAuto/apollo.
[2] *Autoware: Open-source to self-driving*, 2018, [online] Available: https://github.com/CPFL/Autoware
[3] J. Long, E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation", *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431-3440, 6 2015.
[4] *Opencv*, September 2018, [online] Available: https://opencv.org/.
[5] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," DTIC Document, Tech. Rep., 1989.
[6] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deep driving: Learning affordance for direct perception in autonomous driving," in Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2722–2730.
[7] C. Chen, "Extracting cognition out of images for the purpose of autonomous driving," Ph.D. dissertation, PRINCETON UNIVERSITY, 2016.
[8] R. Shanker, A. Jonas, S. Devitt, K. Huberty, S. Flannery, W. Greene, B. Swinburne, G. Locraft, A. Wood, K. Weiss et al., "Autonomous cars: Self-driving the new auto industry paradigm," Morgan Stanley Blue Paper, Morgan Stanley & Co. LLC, 2013.
[9] R. Polishetty, M. Roopaei, and P. Rad, "A next-generation secure cloud-based deep learning license plate recognition for smart cities," in Machine Learning and Applications (ICMLA), 2016 15th IEEE International Conference on. IEEE, 2016, pp. 286–293.
[10] M. Roopaei, P. Rad, and M. Jamshidi, "Deep learning control for complex and large scale cloud systems," Intelligent Automation and Soft Computing (AUTOSOFT).
[11] P. Sheelarani, S. P. Anand, S. Shamili and K. Sruthi, "Effective car parking reservation system based on internet of things technologies," 2016 World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), Coimbatore, India, 2016, pp. 1-4, doi: 10.1109/STARTUP.2016.7583962.
[12] Gowtham S U, Gokulamanikandan M, Pavithran P, Gopinath K , "Interactive Voice & IOT Based Route Navigation System For Visually Impaired People Using Lifi", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 2, Issue 2, pp.323-327, March-April-2017. |