2nd National Conference on Recent Innovation in Engineering, Science, Humanities and Management
REVA University, Yelahanka, Bangalore, (India)
29th-30th March 2019 www.conferenceworld.in ISBN : 978-93-87793-79-8

Implementation of Self Driving car Emulator using Machine Learning

Hemanth K, Jayanth S, Jayasooriyan P R, J Ameer Basha
Prameela Kumari N
School of ECE, REVA UNIVERSITY

ABSTRACT

Self-driving cars is one of the most promising prospects for Machine Learning research. Some of the challenging problems in this field include Speed enhancement of Image processing in detecting lanes, Recognition of Traffic sign boards, Implementation of Vehicle overtaking, adjusting to environmental conditions etc…, Canny edge detection along with Hough Transform is a good solution for lane detection. While Nvidia is graphical simulator visualizer available open source. In this paper we discuss about the overall implementation of a self driving car emulator by combining Canny edge detection along with Hough Transform through Nvidia visualizer.

Keywords: *lane detection; Hough transform; canny edge detection; computer vision*

I. INTRODUCTION

A self-driving car is a vehicle that is capable of visualizing its environment and moving with little or no human intervention. Self-driving cars combine a variety of sensors to compute and perceive their surroundings and act accordingly. Modern high tech control systems interpret sensory information to identify appropriate navigation paths, as well as obstacles and important signboards. But to avoid massive no of sensors, a image processing based approach is undertaken to reduce production cost and enhance the portability of adding self-drive capability to any vehicle.
Self-driving car emulator gives an platform to design and develop such self-drivingcars through a software based design and analysis approach. The emulator is designed through python language. The graphical user interface used is the Nvidia graphical simulator, an open source software designed specially for self driving car design and development. Primary task of the emulator is to capture a set of three images through the front mounted camera, in our software environment we record the video of the road by manually driving the car in the Nvidia simulator, later the set of images are processed in a series of steps to detect the lanes and later use python Inbuilt libraries like socketio and flask to program the simulator to automate the car. Based on the processed image, corresponding steering angle and other attributes are altered. The image processing is done by Keras and Tensorflow ,powerful image processing and deep neural networks supported modules in python.

II. RELATED WORK

Through research of a bunch of IEEE papers and a few other articles makes it evident that Self driving car system has a great potential in automotive industry research.

1Reinhold Behringer
Daimler Protics GmbH
Reinhold Behringer is a consultant at Daimler Protics GmbH. His research interests over the past 25 years include computer vision, augmented realityand virtual , human and computer interaction, automated vehicles and intelligent systems. Behringer received a PhD in engineering from Bundes-wehr University, Munich. He is a member of IEEE

ISBN : 978-93-87793-79-8

2Mohan Trivedi
University of California, San Diego
Mohan Trivedi is a Distinguished Professor of Engineering and director of the Laboratory for Intelligent and Safe Automobiles at the University of California, San Diego. His research interests include intelligent machineand vehicles , human—robot interaction, and human-centered autonomous driving.

III. PROPOSED WORK

3.1 Finding Lanes

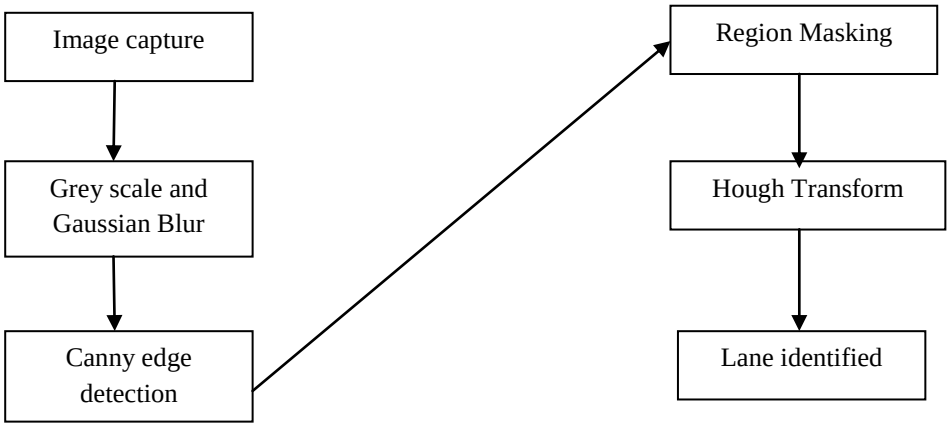

Fig 1, Flowchart for lane detection

To determine the lanes, the image is proposed in a series of steps. First image is converted into a gray image and enhanced by Gaussian blurring using openCV library in python IDE.

Fig 2, Gray scale image

Then the image is converted to a canny image using a inbuilt python function and then masked to a certain region(region of interest), so that processing becomes easy.

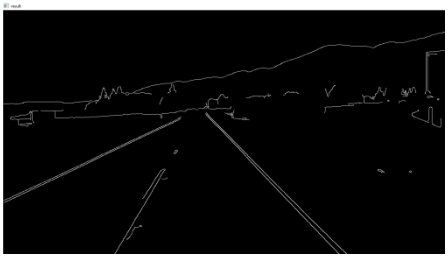

Fig 3, canny image

ISBN : 978-93-87793-79-8

Using Hough Transform we determine the gradient change corresponding to the straight line and drawing a line over the image.

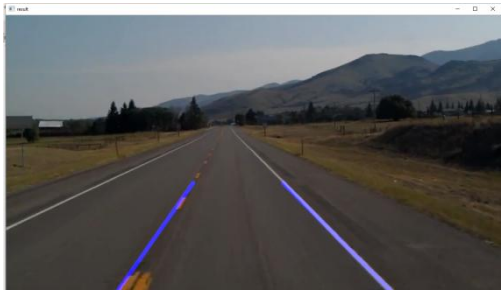

Fig 4, Lane detected image

The same process is carried out for each frame of the video to obtain continuous lane detection. The curves and turns are determined by Perceptron-based Neural Network using Tensorflow (neural network processing library in python.)

3.2 Training Model

The main objective of the project was to train a Deep Network to mimic the human steering behavior while driving, thus being able to drive autonomously on a simulator. To this purpose, the network takes the frame of the frontal camera (say, a roof-mounted camera) as input and predicts the steering direction at each point of time or instant.

Fig 5, GUI based on Nvidia's self-driving program

3.3 The Dataset

Data for this process can be gathered with the Udacity simulator itself. Indeed, when the simulator is set to training mode, the car is controlled manually by the human though the keyboard, and frames and steering directions are stored to disk.

The training set consists of large number of samples. For each sample, two main information are provided:

- three frames from the front, left and right angles from camera respectively

ISBN : 978-93-87793-79-8

- The corresponding steering angle.

3.4 Visualizing training data

We have three frames from different cameras as well as the associated steering direction.

Fig 6, Training steering angle

Each and every frame is preprocessed by cropping the upper and lower part of the frame in this way we remove information that is probably useless for the task of predicting the steering direction.

As we see, each frame is associated to a certain steering angle. The data sets are processed by keras modules and networked by the fit generator.

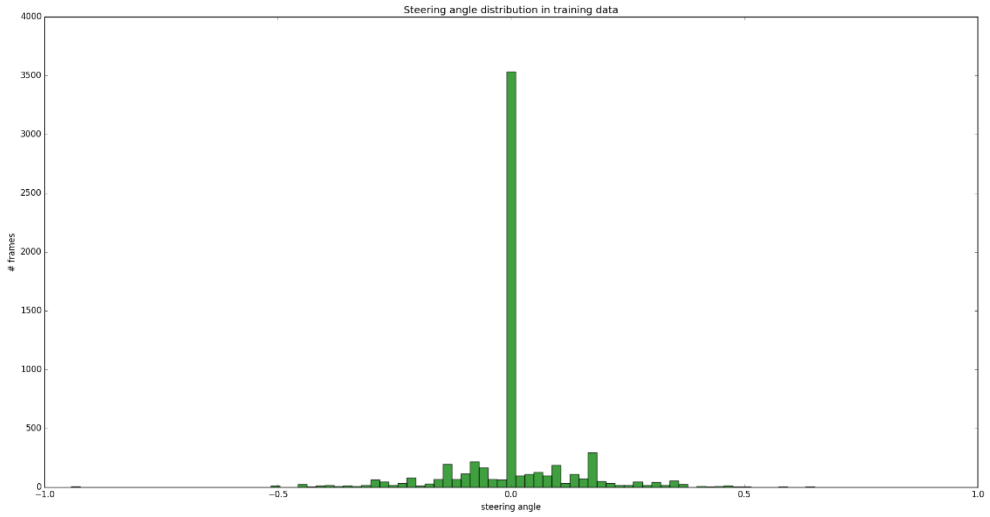

Fig 7, Steering angle v/s Frames

ISBN : 978-93-87793-79-8

Fig 8, Simulator Result

VI. CONCLUSION

The technique of combining canny edge detection and Hough transform for lane detection has been used effectively to detect the lanes in various lane-lines condition. This lane detection method along with python codes has been executed through Nvidia Nanometer visualizer for driving a car in autonomous mode.

REFERENCES

1. Huang, T. (1996-11-19). Vandoni, Carlo**"Computer Vision : Evolution And Promise"**. 19th CERN School of Computing. Geneva
2.Dana H. Ballard; Christopher M. Brown (1982). **"Computer Vision. Prentice Hall".**
3.Shapiro, Linda and Stockman, George. **"Computer Vision"**, PrenticeHall, Inc. 2001.
4. Schmidhuber, J. (2015**). "Deep Learning in Neural Networks: An Overview"**. Neural Networks.
5. Lindeberg, Tony (2001) [1994], **"Edge detection",** in Hazewinkel
6. Duda and P. E. Hart, **"Use of the Hough Transformation to Detect Lines and Curves in Pictures",**Comm. ACM, Vol. 15, pp. 11–15 (January, 1972)