CNN based Curved Path Detection and Obstacle Avoidance for Autonomous Rover

Sudeep Aryan G [1], Bhanu Meher Srinavas R [2] ,Neethika K [3],Dr.Jayabarathi R [4]
Department of Electrical and Electronics Engineering Amrita School of Engineering,Coimbatore
Amrita Vishwa Vidyapeetham, India
cb.en.u4eee19151@cb.students.amrita.edu, r_jayabarathi@cb.amrita.edu

Abstract—**For the new industrialization, automation is an uprising field which is evolving everyday. This can also be noticed in the Transportation industry, which is moving towards self-driving vehicles. But complete self-driving vehicles are not yet achieved. To achieve a self-driving vehicle, algorithms like lane detection, obstacle detection, traffic sign recognition etc.., can be used. If a full fledged self-driving vehicle is achieved, there will be less potential for road accidents. This paper explores both hardware and software elements of self-driving vehicles that includes the use of deep learning technique, Convolutional Neural Networks (CNN) for lane detection. Software components such as Udacity simulator, along with hardware components like raspberry Pi 3b+, pi Camera, and ultrasonic sensors, are used. In this paper, raspberry pi 3B+ is used as a controller for the four wheel rover. Images from the pi camera are delivered to the raspberry pi. A trained CNN model will forecast the steering angle. CNN algorithm will give steering direction as output by taking images from pi camera as input. This model differs from Nividia's model in its complexity, a large parameter count, training data dependency, and algorithmic flexibility. Another key aspect is that this model's output is the mean value of the steering value generated from Pixel Summation and the steering value generated by Nvidia's CNN model. It provides the final output to the motor drivers, causing the motor drivers to deliver appropriate signals to the relevant motors. The outcome will be a self-driving rover which will follow an instructed path and be able to detect an obstacle if present and will be able to continue on the lane.**

Index Terms—**Convolutional Neural Network, Lane Detection, Self Driving Vehicle**

I. Introduction

Automation is one of the most booming fields in the new industrialization today. Transportation industry has also noticed this in self-driving vehicles. However, complete self-driving vehicles have not yet been adopted. To overcome this, optimization of algorithms like lane detection, obstacle detection, etc are used. As a result, the occurrence of road accidents due to human error won't be an issue since the system is fully automated. The self-driving vehicles are expected to work in almost all transportation areas especially in industries using machineries and the use of these in general transportation for domestic users to avoid errors and also be helpful for the blind or physically challenged or senior citizens. The rover is automated and is able to detect any obstacle in its path and continues to move in the best possible track by avoiding the obstacle. But the unique and interesting point behind this paper is the path that the rover travels can be both linear or circular with the accuracy is nearly equal.

Motivated by the approach of enhancing transfer architecture with additional features, as seen in models like ChatGPT, Brad and other AI Tools, our research extends the NVIDIA architecture. We introduce groundbreaking improvements, such as pixel summation, which leads to a model that outperforms current solutions in terms of efficiency.

In recent years, there has been a significant development in the field of autonomous vehicles, especially self-driving cars. One of the architectures used for comparison is the architecture proposed by Nvidia described in [1] which does not factor in Pixel Summation for generating steering angle value while, the other is a lesser complex CNN architecture (with a total of 7 layers) .Udacity simulator is a popular tool used for comparing the accuracy of different CNN architectures in predicting the steering angle for a car. [2] provides an in-depth overview of the Udacity simulator and how it can be used to collect data in training mode, where the car is controlled manually. The data collected can then be used to train a CNN to predict the steering angle, which is used in autonomous mode. Several research papers have been published on the topic of self-driving rovers using a Raspberry Pi controller and deep learning algorithms. [3] describes a custom CNN model with a total of 9 layers that predicts both steering angle and speed for an autonomous car in the Udacity simulation platform, which was trained three times with different numbers of epochs. [4] discusses the working of a self-driving four-wheeled rover using a deep learning algorithm with a CNN that predicts the direction for the rover.

The use of Raspberry Pi as the main processor is common in the development of self-driving rovers, as described in [5], that uses a prototype with a camera for lane detection using a CNN that sends signals to an Arduino to drive the rover's motors . [6] describes a rover capable of multi-mode autonomous driving using four IR sensors for lane detection, four ultrasonic sensors for object detection, and a camera for object detection and traffic sign recognition .[7] compares the datasets of Udacity and Nvidia on the CARLA simulation platform, a CNN architecture similar to the one proposed by Nvidia but with fewer layers was used, with Raspberry Pi used for image processing and CNN operation and Arduino mega for interfacing ultrasonic sensors and motors. Additionally, [8] uses a remote control car with a Pi camera connected to a Raspberry Pi was used to train a CNN to understand traffic signs (only for direction signs) to predict the steering angle and control the motors using an Arduino mega in a lane .

[11] discusses the reliability of the rover depends mainly on its mechanical design, including the various methods for movement and the torque selection of the motors used, as proposed by Zhang Guowei in a journal . Finally, the convolutional neural network architecture used for the project was proposed by Nvidia in [12], which also used a real car, and to control the steering angle, used Nvidia Drive PX as their motor driver.

II. Self Driving Rover

A. *OBJECTIVE*

To simulate and implement a CNN based self driving rover model with raspberry pi which is able to avoid obstacles in its path and able to move in the given lane. Raspberry pi 3B+ is used as a controller for the four wheeled rover. Images from the pi camera are delivered to the raspberry pi. Following image augmentation, photos will be submitted to CNN, which allows to forecast the direction, where an ultrasonic sensor will be used to identify the objects at a distance. An algorithm developed from the data given by the CNN model and ultrasonic sensor helps the rover to avoid the obstacle and move in its defined path. It provides the final output to the motor drivers, causing the motor drivers to deliver appropriate signals to the relevant motors, helping the rover to become an autonomous moving vehicle.

B. *OUTLINE OF THE PAPER*

Nvidia proposed architecture of CNN model is trained using the dataset which was collected from training mode of the Udacity simulator. The training is done in two different paths provided by the Udacity simulation platform and in testing the path chosen is random among the two trained paths. Similar paths are used in testing and training that increases the accuracy of the CNN model. Steering angle will be the output of the trained CNN model. Raspberry Pi 3B+ is used as the controller used for the four wheel rover and Pi-cam is used to capture the objects. The robot designed is a four wheeled Rover built using 4 high torque DC motors which are controlled by a motor driver(L298N). The raspberry pi will be controlled by a connecting to a laptop/desktop. Ultrasonic Sensor is used to detect the obstacle. NVIDIA's proposed CNN model is used for lane detection and lane/curvature keeping (and ensures it is in the right path). An algorithm that takes input from both CNN and ultrasonic sensor, so it should be able to select a path without obstacles while still being on the lane.

III. SIMULATION

The Simulation of a CNN model Used for self driving cars. For simulation, Udacity simulator is used. A CNN architecture that is proposed by Nvidia specially for Self driving cars. The required model should be capable of mapping the image input

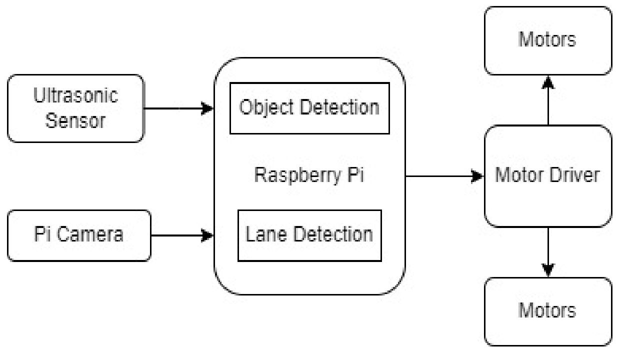

Fig. 1. Block Diagram of Self Driving Rover

to steering angle as output. A CNN model is created, and trained using the data set that is collected during training mode of the Udacity simulator. Then it is tested back in autonomous mode. In this paper, architecture proposed by Nvidia for Self driving cars for CNN model is used.

A. *METHODOLOGY*

1) Data Collection: Using Training mode in the Udacity simulation platform , the data is collected (for 10 laps). This dataset is stored in the form of images , and a '.csv' file which contains the path of images and corresponding steering angle, throttle, brake, speed. From the data collected from Udacity, consider only the center images and steering angle and ignore left and right images and throttle, brake, speed for training the further steps from the data.

2) Visualize and Balance Data: The need for the visualization of data is to check the steering angle. For example, if the steering angle is too much left then the rover moves left . And to avoid this , balancing the data by making it continuous between -1 to +1. This is achieved by creating bins which are odd in total so that zero is in the middle, this bins consist of 1000 samples. This data is adjusted in a way so that the car has highest zero angles while training.

3) Split for Training and Validation: The data has been loaded from pandas(csv file) to two lists, one for the image path and other for steering angle. Split data for training and validation, after every epoch. Validation data is needed to check the performance of the model for unknown data or while testing.

4) Augmentation: Image Augmentation is used to alter existing data and create new data for model training to increase the dataset size. The augmentation techniques used are Pan, Zoom, Brightness, Flip.
Pan
From the Fig. 2 and Fig. 3, Shifting of image to right can be observed. similarly left or up, down can also be done.
Zoom
In the Fig. 4, the image is Zoomed
Brightness
From the Fig. 2 and Fig. 5, it can be observed that the brightness of the image can be increased or decreased.

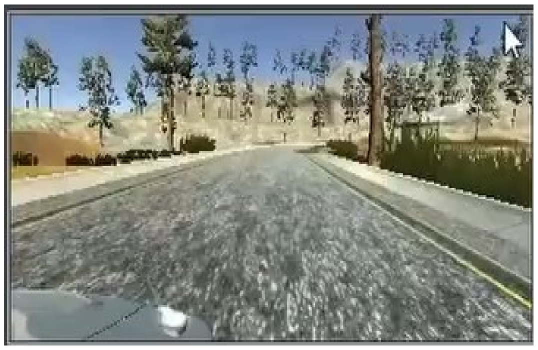

Fig. 2. Image before Augmentation

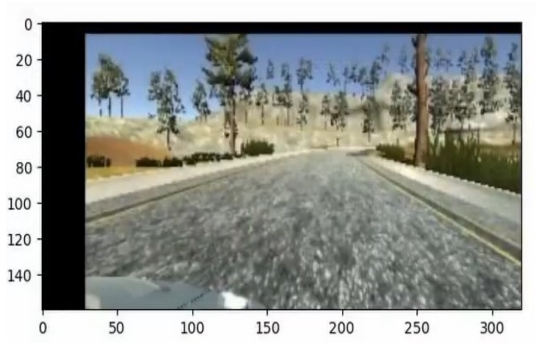

Fig. 3. Image after Panning

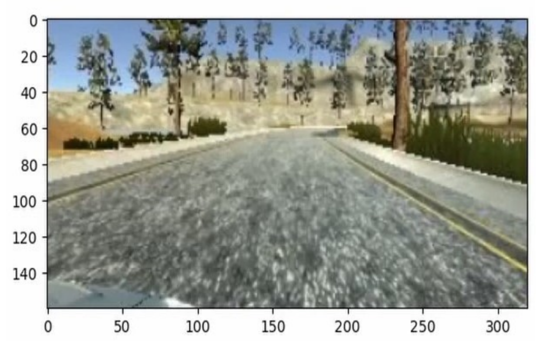

Fig. 4. Zoomed Image

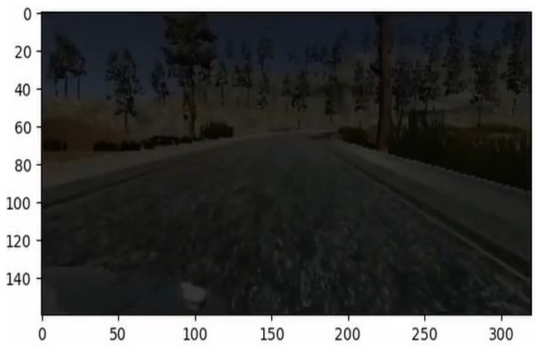

Fig. 5. Image after decreasing the brightness

Flip :
From the Fig. 2 and Fig. 6, it is observed that the image is flipped.

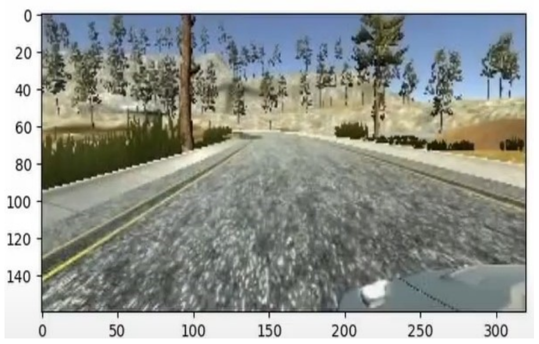

Fig. 6. Flipped image

5) *Pre-Processing:* These techniques below are performed while training, not to increase the dataset but to increase the model efficiency.
Cropping
In the Fig. 7, Image is cropped for the required region of the image(Road).

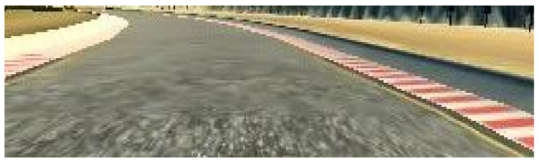

Fig. 7. Cropped Image

YUV Color Space
Fig. 8, shows that the YUV is used rather than RGB to reduce the bandwidth which makes the model more efficient. So that path can be easily identified. In YUV color space Y indicates Brightness ,U indicates Blue Projection ,V indicates Red Projects.

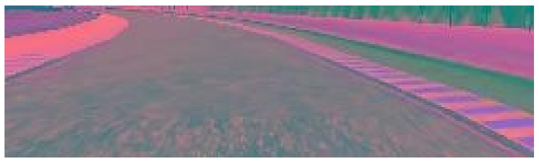

Fig. 8. YUV Image

6) *Batch Generator:* All the images are not sent to the model at a time, the images are sent in batches. So the function "Batch Generator" is created to send images in batches.

7) *Creating the Model:* Fig. 9, shows the architecture of the Nvidia Proposed CNN model for Self driving cars, that is used in this paper.
Model is trained using the data batch generator generates and then ,validate the model. The model is trained for 10 epoch.

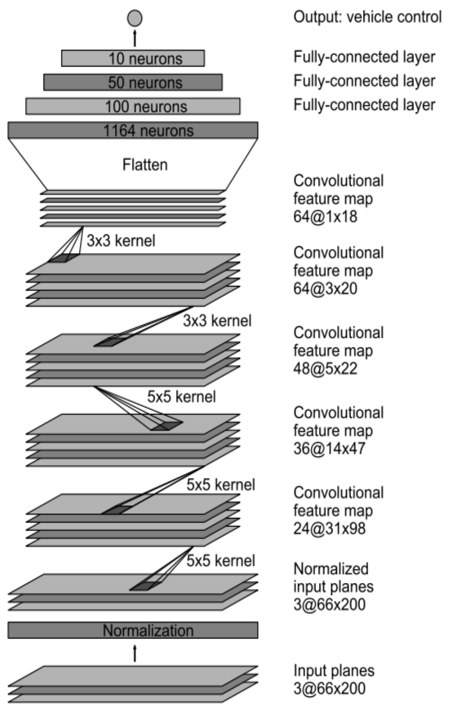

Fig. 9. CNN model [11]

B. TESTING

The trained model is tested in autonomous mode in the Udacity simulator to check whether the model is trained well. If the model is trained well,it will drive well at high speeds and different types of tracks in autonomous mode. As the final output,the Rover is driving well in autonomous mode using trained CNN.

C. SIMULATION RESULT

Nvidia proposed architecture of CNN model is trained using the dataset which was collected from training mode of the Udacity simulator. In a similar manner, VGG16 model also has been trained. The accuracy of Nvidia model and VGG16 was compared. The accuracy of the Nvidia model is 94% and the VGG16 model is 82%. Also, the VGG16 model has more parameters than Nvidia model. As it is evident from the above results, the Nvidia model outperforms the VGG16 model because of its higher accuracy and less parameters. We improved the existing nividia core CNN architecture by reducing the excessive hyperparameters hence it performs in lesser duration of time. With the help of steering angle the CNN model is being trained by doing various operations on python. The training is done in two different paths provided by the source i.e, Udacity and in testing the path chosen is random among the two trained paths. CNN model is tested in the Udacity simulator. During training, our CNN model was specifically trained on one path. Since the paths are similar in testing and training the accuracy is high for the CNN model. However, during testing, we successfully implemented it on different paths, and the model performed effectively.

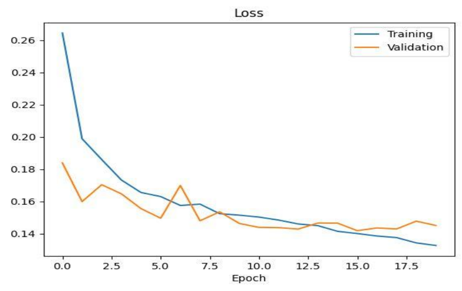

Fig. 10. Simulation Result

IV. IMPLEMENTATION

Calibration of different components and integrating them together with the Rover's body. The calibrations is done for ultrasonic sensor and motor driver. The Raspberry pi camera is used to observe the surroundings in front of the rover. In this paper, four high torque DC motors are used. The motor driver is used to control the motors. The 12V battery is used to power the motor driver. The ultrasonic sensor is used to sense the obstacle in the path in which the rover is moving. Raspberry Pi has the capability to interact with the outside world and has been used widely for digital projects.

A. INTERFACING WITH MOTORS

The motors used are of high torque medium speed applications.These motors are used for the movement of the rover in desired direction and desired speed. There are four motors which are controlled by the motor driver which provides the supply for each motor.

The motors of the rover can be modeled as the working of two motors. The motors on the right side of the rover are connected in parallel. Similarly the motors on the left side are also connected in parallel. Both connected sides are connected to the motor driver and the signals of the motor driver is transmitted from the raspberry pi.

GPIO PIN USAGE

The GPIO pins used are configured as output.

Motor 1A 1B= GPIO pin 5 and 6

Motor 2A 2B= GPIO pin 13 and 19

The LED can be driven ON or OFF using a raspberry pi, because 20 mA current is enough for an LED. But it cannot handle more than that. Therefore the L298N motor controller is used to provide supply for ET-PGM36 motors which requires 12V supply.

The L298N motor driver functions are based on the H-bridge configuration, which makes it easy to control the direction of rotation of a DC motor. H- bridge is used to run motors both in a clockwise and anti-clockwise direction. As mentioned earlier, it is capable of running two motors in any direction at the same time. The number of switches are four namely J1, J2, J3, J4 respectively.

When the switches J1 and J4 are closed, then a positive voltage is applied across the motor, and the motor rotates. When J1 and J4 are ON, the left side terminal is more positive than the right side terminal, and the motor rotates in a particular direction.

When J2 and J3 are closed, J1 and J4 are opened, then a negative voltage is been applied across the motor as a result the motor rotates in an inverted direction. When J2 and J3 are ON, the right side terminal is more positive than the left side terminal, making the motor rotate in the other direction. When all the terminals are shorted, the motor stops.

For L298N IC, H-bridge circuit is the best combination for driving the DC motor. Generally, these kind of circuits are used in Robots for controlling the motor. In this paper, the rotation of the motor is controlled by left input pins where the motor is connected across and in the right-hand side for the motor right inputs are connected. Therefore motors rotating direction is decided by the switches. On the other hand, L298N motor driver can also perform speed control. To achieve this the PWM signals should be provided in motor input pins. The width of the pulses determines the speed of the motor. If the pulse is wider, then the motor rotates faster. There are two input pins Vss and Vs for L298N motor driver. The Vss receives power to drive the motor from 5V to 25V whereas the Vs receives power to drive the logic circuitry from 5V to 7V

Whenever the forward button is clicked , it will call the forward. cgi script which has the wiring pi commands to directly control the GPIO pins. The same procedure will happen if left, right, reverse and stop buttons are clicked. In the H-bridge circuit there is a internal voltage drop in the switching transistors. Due to this the L298N motor driver has a voltage drop of 2V.

B. *INTERFACING ULTRASONIC SENSOR*

Trig is given to Rpi GPIO pin since the trigger is input to sensor it does not affect the GPIO but Echo pin is output from sensor and input to the Raspberry pi. Vcc=5V, ground to be grounded. The ultrasonic sensor will give a digital high of 5V, whereas the GPIO pins of raspberry pi will work at 3.3V. So a voltage divider is used to give only a digital high of 3.3V to the raspberry pi. Here the Vcc is connected to the 5V pins of Raspberry Pi, ground can be connected to ground, the trigger is connected to GPIO 20 and echo is connected to GPIO 21. Every time an object is placed in-front of the sensor the distance is calculated it will give a pulse when the triggering signal is given to it. The GPIO pins of number 20 will be set as "Trigger", and number 21 is set as "Echo". A signal of 10µs is developed from the trigger and is given to raspberry pi. The HC-SR04 will give a digital high when it has a triggering pulse and will generate a 40kHz pulse. When it again senses the 40kHz signal, the digital high signal from the sensor will go low. The on time of the cycle is taken by the pulse to travel two times the distance between sensor and surface. The start time and stop time is calculated by using time function. This difference will give them time. As the speed of sound is 340m/s and the distance covered is twice between the sensor and surface. The distance is calculated by, Distance=(stop time - start time)*17000 cm

The implementation of the hardware components have been explained for each component. The calibration for different components for different components have been made considering their performance limit. While doing the calibration the values have to be carefully noted. The Raspberry pi is the one which interfaces the total components with each other and itself. During the interfacing the wiring should be done carefully and properly so that its good while the rover is in motion.

V. LANE DETECTION USING IMAGE PROCESSING

It deals with the Lane detection module of the paper. For lane detection, Raspberry Pi V2 Camera module, Raspberry Pi 3b+ as controller and a Four Wheeled Rover. A CNN is trained to predict steering angle for the lane. For training CNN, Image processing techniques like Thresholding, wrapping and pixel summation are used to find curves in the lane. And major Python libraries like Opencv and Tensorflow are used. CNN architecture used was proposed by Nvidia [12], which was used for a real car. First by driving the rover manually through a keyboard, using Image processing techniques mentioned, data was collected . And then, this data is used to train the neural network.

A. *THRESHOLDING*

Thresholding is a common image processing technique used to segment images as foreground or background regions based on threshold values of colour. Here RGB or HSV colour space image is converted into binary image by applying a range of colour to find. It can select what is useful and what region is useless. The original image is converted into 8-bit binary image with white as 255 and black as 0.

B. *WARPING LANE*

Warping lane is an image processing technique where image is converted into a new coordinate system, so image alignment is changed as required. The curve value of lane right now is required rather than the whole lane. So cropping of the lane can be done. Bird eye view is used on the images to get a top view, which is important and easy to analyse where the curve is present and how much curve it is.

C. *PIXEL SUMMATION*

The summation of these pixels basically helps in finding the histogram, not only the direction of the curve but also the amount of the curve is required. To get this value all the indices of the curve have to be found then these indices values are to be averaged.

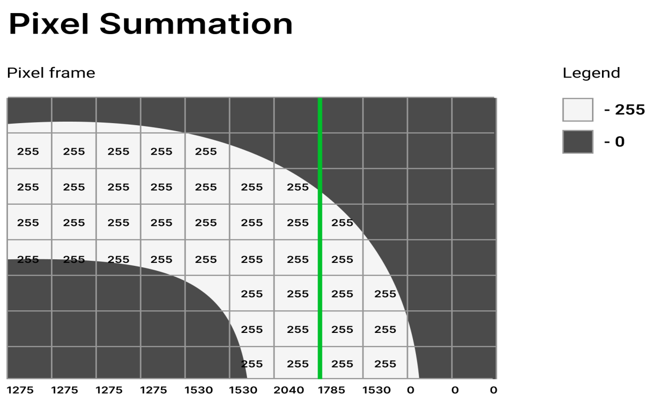

Fig. 11. Pixel Summation

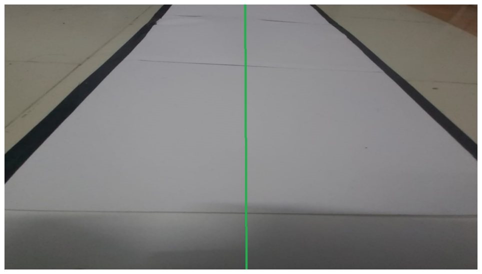

Fig. 12. Straight

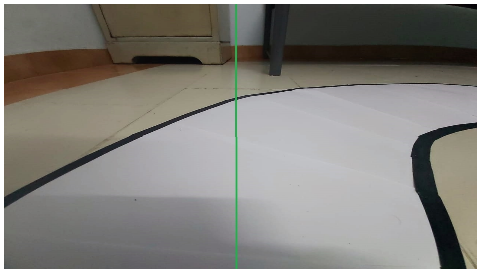

Fig. 13. Right

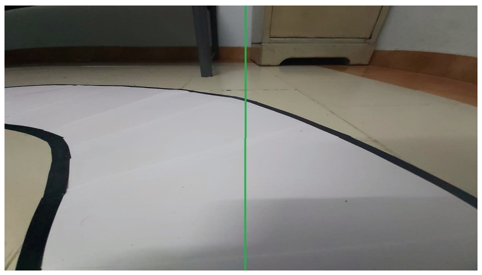

Fig. 14. Left

D. *IMAGE PROCESSING RESULTS*

Finally as a result the curve value of lane at a particular instant is acquired. Using this, a dataset to train our CNN model can be made.The sum of pixels in each column plays a crucial role in determining the direction and degree of a curve for a rover, considering the rover's center as a fixed reference point.

If there are more pixels detected on the right side of the pixel frame, the rover will turn to the right Fig. 13. Conversely, if more pixels are detected on the left side of the frame, the rover will turn to the left Fig. 14. If there is an equal number of pixels on both sides, the rover will continue straight Fig. 12.

VI. DATA COLLECTION AND TRAINING

Data collection is the process of gathering and compiling data that is used to train the neural network. Data collection involves capturing images and their corresponding steering angles from the rover while it is in operation. These images and steering angles are then utilized to train a CNN model that predicts the appropriate steering angle for a given image.

The analog values from '-1' to '+1' used to represent the steering angles provide a more precise and nuanced control over the rover's movements. This means that the machine learning model can predict a range of steering angles that fall within this range, leading to smoother and more natural movements for the rover. Overall, the data collection process and the different modules within the rover's controller work together to enable the successful operation of the machine learning-based rover.

Training a model involves few steps same as mentioned earlier in Methodology of Simulation. After all the steps we finally get a model.h file after training process is completed. This model.h file along with the webcam allows us to make a prediction and using this prediction when the steering angle is obtained this angle will be sent to the motor to run the rover. For all this required is an extra file, webcam module and motor module. In the hardware implementation, the calibrations

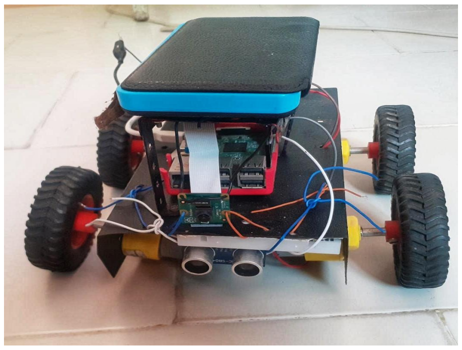

Fig. 15. Autonomous Rover

for the sensor have been made and the calculation for the motors and other required components have been made. The Raspberry pi is the main unit of the rover which controls other components. Raspbian operating system is the OS used in the raspberry pi. The ultrasonic sensor HC-SR04 is used for sensing the surroundings. The motors used in this are high torque DC motors, ET-PGM36. And the motor driver is L298N for controlling the motors.

VII. CONCLUSION

In this paper, the simulation of self-driving cars and the hardware implementation of the rover have been mentioned. In the simulation, the Udacity platform has been used to simulate. The simulation utilized a CNN architecture proposed by Nvidia, achieving an accuracy of 94%. The performance of Nvidia model was compared with that of VGG16 model and it was found that the Nvidia model was more efficient. The input dataset for training the model has been taken from the same platform in training mode. For training the model autonomous mode had been used. For training and testing the paths used are inbuilt in Udacity. Since the paths are similar in testing and training the accuracy is high for the CNN model. Additionally, the output is more accurate since it is derived from the mean values of steering angle generated by Pixel Summation and the steering angle generated by Nvidia CNN Architecture.

In the hardware implementation, the calibrations for the sensor have been made and the calculation for the motors and other required components have been made. The Raspberry pi is the main unit of the rover which controls other components. Raspbian operating system is the OS used in the raspberry pi. The ultrasonic sensor HC-SR04 is used for sensing the surroundings. The motors used in this are high torque DC motors, ET-PGM36. And the motor driver is L298N for controlling the motors. In conclusion, our model outperforms the Nvidia core architecture with enhanced efficiency, including improved accuracy, while requiring fewer hyperparameters, making it a highly time-efficient solution for autonomous systems.

REFERENCES

[1] S. Lade, P. Shrivastav, S. Waghmare, S. Hon, S. Waghmode and S. Teli, ”Simulation of Self Driving Car Using Deep Learning,” 2021 International Conference on Emerging Smart Computing and Informatics (ESCI), 2021, pp. 175-180, doi: 10.1109/ESCI50559.2021.9396941.

[2] A. Khanum, C. -fY. Lee and C. -S. Yang, ”End-to-End Deep Learning Model for Steering Angle Control of Autonomous Vehicles,” 2020 International Symposium on Computer, Consumer and Control(IS3C),2020,pp.189-192, doi:10.1109/IS3C50286.2020.00056.

[3] M. -T. Duong, T. -D. Do and M. -H. Le, ”Navigating Self-Driving Vehicles Using Convolutional Neural Network,” 2018 4th International Conference on Green Technology and Sustainable Development(GTSD),2018,pp.607-610, doi:10.1109/GTSD.2018.8595533.

[4] A. K. Jain, ”Working model of Self-driving car using Convolutional Neural Network, Raspberry Pi and Arduino,” 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2018, pp. 1630-1635,doi:10.1109/ICECA.2018.8474620.

[5] P. G. Chaitra, V. Deepthi, S. Gautami, H. M. Suraj and N. Kumar, ”Convolutional Neural Network based Working Model of Self Driving Car - a Study,” 2020 International Conference on Electronics and Sustainable Communication Systems (ICESC), 2020, pp. 645-650, doi:10.1109/ICESC48915.2020.9155826

[6] M. Azab and B. M. A. Moustafa, ”Multi-Mode Self-Driving EV Control for Industrial Sites and Logistic Stores,” 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), 2020, pp.41-46, doi:10.1109/NILES50944.2020.9257878.

[7] A. Agnihotri, P. Saraf and K. R. Bapnad, ”A Convolutional Neural Network Approach Towards Self-Driving Cars,” 2019 IEEE 16th India Council International Conference (INDICON), 2019, pp. 1-4, doi:10.1109/INDICON47234.2019.9030307.

[8] T. -D. Do, M. -T. Duong, Q. -V. Dang and M. -H. Le, ”Real-Time Self-Driving Car Navigation Using Deep Neural Network,” 2018 4th International Conference on Green Technology and Sustainable Development (GTSD), 2018, pp. 7-12, doi:10.1109/GTSD.2018.8595590.

[9] J.Borenstein and Y. Koren, ”Obstacle avoidance with ultrasonic sensors,” IEEE Journal on Robotics and Automation, vol. 4, no. 2, pp. 213-218, April 1988, doi:10.1109/56.2085.

[10] Zhang Guowei et al., ”Development of robotic spreader for earthquake rescue”, 2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014), 2014. doi 10.1109/ssrr.2014.7017679

[11] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, Karol Zieba NVIDIA Corporation End to End Learning for Self-Driving Cars

[12] R. Chinmayi, Yogesh Kumar Jayam, Venkatesh Tunuguntla, Jaideep Venkat Dammuru, Harshith Nadella, Sai Sri Krishna Anudeep Dulla, Leela Sathya Kartheek Raja, Jayachandran G Nair ., ”Obstacle Detection and Avoidance Robot,” 2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), Madurai, India, 2018, pp. 1-6, doi: 10.1109/ICCIC.2018.8782344.

[13] K. Merugu and S. Adarsh, ”Multi lane detection, curve fitting and lane type classification,” 2022 IEEE 19th India Council International Conference (INDICON), Kochi, India, 2022, pp. 1-6, doi: 10.1109/INDICON56171.2022.10039722.

[14] R. K. Harini, J. Hari Krishna, A. Haneela Reddy, A. Sneha and R. Jayabarathi, ”A Comparative Study of Different MPPT Algorithms and Implementation of Single Axis Solar Tracker,” 2022 International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), Shivamogga, India, 2022, pp. 156-161, doi: 10.1109/DISCOVER55800.2022.9974619.

[15] Bhavani, M., Peeyush, K.P., Jayabarathi, R. (2023). Plant Health Analyzer Using Convolutional Neural Networks. In: Bindhu, V., Tavares, J.M.R.S., Vuppalapati, C. (eds) Proceedings of Fourth International Conference on Communication, Computing and Electronics Systems . Lecture Notes in Electrical Engineering, vol 977. Springer, Singapore. https://doi.org/10.1007/978-981-19-7753-$4_2$6.