AUTONOMOUS VEHICLE USING ARTIFICIAL INTELLIGENCE

[1]Suyash Chavan, [2] Chetan Ubale, [3] Divya Ingale, [4]Smita Lonkar

[1]Student, [2]Student, [3]Student, [4]Guide

[1]Department of Electronics and Telecommunications

[1]Shivajirao S. Jondhale College of Engineering, Dombivli, India

Abstract: *This project has been worked upon within the context of a time where each day we are witnessing the ever-increasing evolution in the field of Autonomous Self-driving vehicles due to an increase in the computing power and processing speeds of the modern computers. Self-driving vehicles, being the buzz of the 21st century has been steadily cementing their crucial role and more importantly, their humble contribution in reducing the fatal driving errors caused by human inputs such as reckless driving, violating lane-keeping laws, etc. ultimately leading to damage of life and property. Just as fatal as human driving errors can become, pedestrian errors similarly can yield equally life-threatening losses. Thus, Autonomous Self-driving vehicles would be playing a crucial role to curb these fatal losses and in turn, make the journey safer and relaxing for the onboard passengers. Self-driving vehicles developed by Tesla, Google Waymo autonomous vehicle project group and the modified vehicles by Nvidia have been under development stages in order to increase their reliability for common masses. This final year project depicts how a Self-driving vehicle can be devised from a simple Remote-controlled car by using a Raspberry-Pi as our Data acquisition system and using the supervised learning method of machine learning with the help of Convolutional neural network model. The goal of the project is to devise a Self-driving vehicle from a scratch with extensive training over various parameters using minimum viable compute resources. The Self-driving car was developed in two stages across a span of 6 months and then finally training the model for a further two months to acquire the desired results.*

Key Words: Self driving vehicle, Machine learning, Neural network, Raspberry Pi, Arduino.

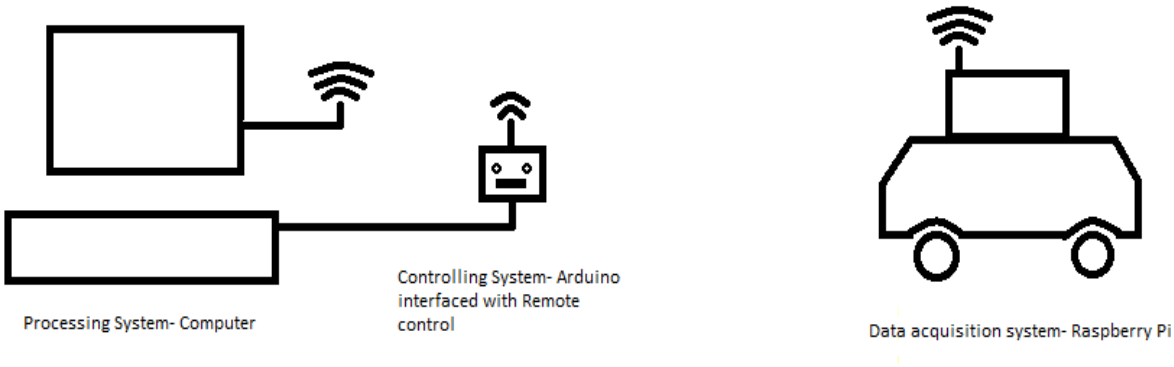
INTRODUCTION

The introduction of the System on Chip (SoC), such as the Raspberry Pi has enabled the notion of creating various “smart devices” projects and the “data acquisition systems” as a part of a bigger project. Many IoT projects and monitoring systems have been developed amongst the hobbyist communities to provide high level integration and sufficed compute resources to acquire relevant data in real-time. One such popular branch of these projects is using the SoC for the development of Autonomous driving vehicles.

An Autonomous Self-driving vehicle is a transportation vehicle with computer onboard capabilities such as decision making and constantly being aware of the surroundings. It is basically a vehicle that can make its own decision based on the real-time road and traffic conditions and help the passengers to travel safe and sound from point A to point B.

I. PROPOSED SYSTEM

Figure 1: Proposed System



The objective of implementing the Self-Driving Vehicle is to provide a safer means of transport by relying on the computational accuracy of the well-trained neural network model to drive a vehicle autonomously and in that process, minimizing the need of any manual intervention for driving the vehicle. As mentioned above in the figure, the proposed system comprises of a Data-Acquisition System, Processing System and a Controlling System. The selected data acquisition unit for the vehicle is Raspberry-Pi 3(Model B) interfaced with a RGB Pi-camera and an ultrasonic sensor (HC-SR04).

The operating system used for the Raspberry Pi is Raspbian (Stretch) which is based upon the Debian package and a Micro SD card is used for storing and running the operating system. We are including Artificial Intelligence into the Processing System i.e. a computer, with a well trained convolutional neural network model trained upon several image data sets. The predicted outputs of our model will drive the corresponding output of the Controlling system which consists of an Arduino Uno interfaced with the relevant pins of the Remote Control of the vehicle.

II. REQUIREMENT ANALYSIS

[A] Software Requirement:

1. Programming Languages:

- Python
  Python is a general-purpose scripting language with a support for various libraries. The entire AI backend has been developed in Python alongside various libraries to handle the required computations.
- C language
  We will be using Arduino Uno for controlling the remote-control inputs. Thus, to program the Arduino accordingly, we are using C language to feed in the control logic through serial communication.

2. Python Libraries:

- Open CV
  The Open CV is an open source computer vision library which has many functions essential for computer vision. We will be using it to train our Haar cascades in order to detect traffic lights and roadside stop signs.
- Numpy
  Numpy is a general-purpose array processing package which provides a high-performance multidimensional array object along with the required tools for working with these arrays. We are going to process our captured images from the real-time feed of the Pi camera and end up with a NPZ file which is used to train our Neural network.
- pygame
  pygame is an open source Python library which is primarily used for making multimedia applications like games, etc. We are incorporating the pygame library in order to map our keystrokes and drive the vehicle using the computer keyboard. The keystrokes will be accurately mapped to the relevant frames captured during the testing phases.

[B] Hardware Requirements:

1. Remote-Controlled Car:

   A fairly decent sized remote-controlled car is chosen with the idea of mounting the image acquisition system on top of it along with a portable battery for powering up the Raspberry Pi at 5V.

2. Raspberry Pi Model B:

   The Raspberry Pi is a series of small single-board computers with wireless LAN and Bluetooth connectivity. Here, we use Raspberry Pi 3 Model B which incorporates a quad-core processor and can easily handle the required heavy image processing computations. Wireless LAN functionality is used to set-up a TCP server with processing computer for easy remote access to the Raspberry Pi.

3. Raspberry Pi camera module:

   The Raspberry Pi camera module can be used to take high-definition video, as well as stills photographs. We have interfaced the Pi camera module with the Raspberry Pi 3B in order to gather the training data by capturing the relevant frames of the track while manually driving the vehicle.

4. HC-SR04 Ultrasonic sensor module:

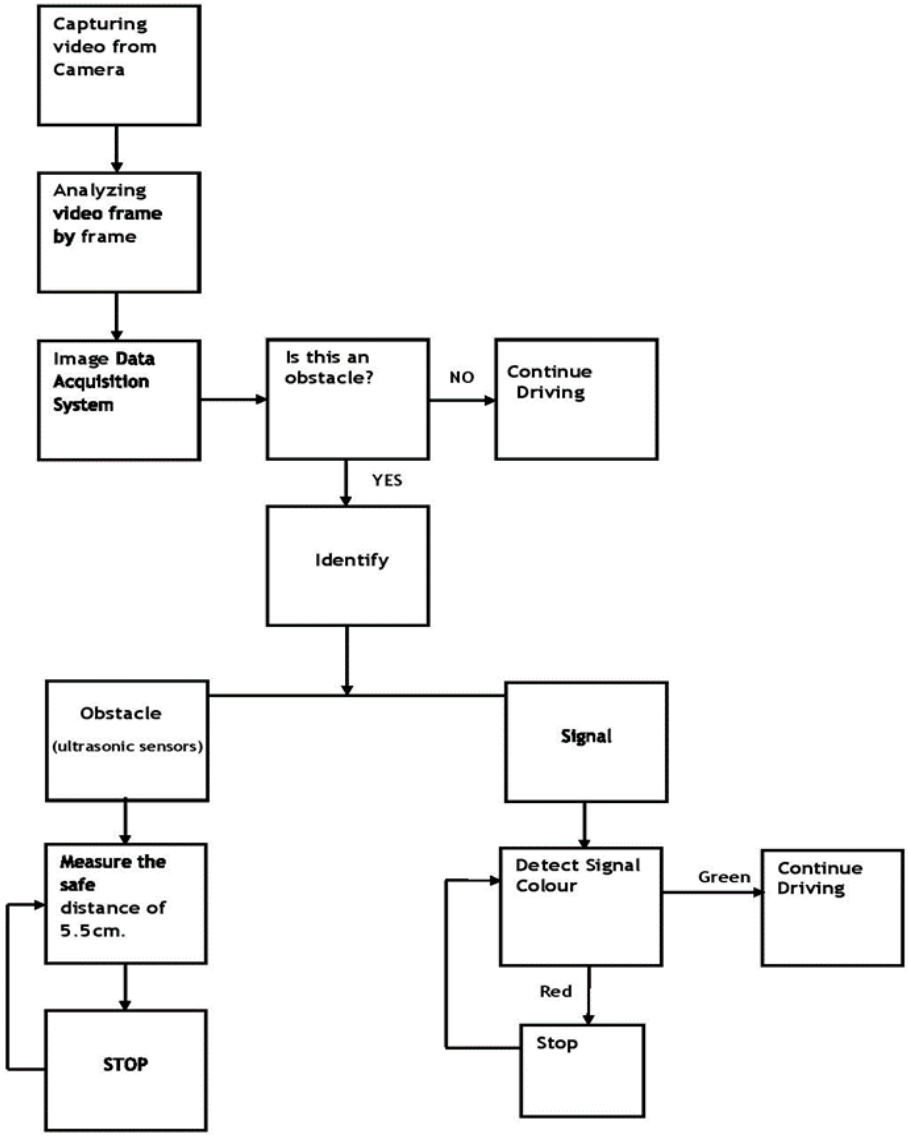
   The HC-SR04 ultrasonic sensor module is being interfaced as well with the Raspberry Pi. It will sense all the obstacles in its path and if it detects any obstruction, it will compute the distance to the obstacle and notifies Raspberry Pi regarding the same.

5. Arduino Uno:

   The Remote-Controlled car used in this project has an on/off switch type controller. When a button is pressed, the resistance between the relevant chip pin and ground is zero. Thus, an Arduino board is used to simulate button-press actions. Four Arduino pins are chosen to connect four chip pins on the controller, corresponding to forward, reverse, left and right actions respectively. Arduino pins sending LOW signal indicates grounding the chip pins of the controller; on the other hand, sending HIGH signal indicates the resistance between chip pins and ground remain unchanged.

III. PROCEDURAL DESIGN

Figure 2: Procedural Design



- Raspberry-Pi captures frames from the video camera and then sends data to the Processing system.
- The video frames received are analysed and from each frame, the image above the horizon is discarded since it will not be serving any purpose during the processing step. These modified frames are analysed using numpy to obtain NPZ file which comprises the relevant data for controlling the vehicle.
- The NPZ file is then used by the Neural network in order to detect objects and predict correct road conditions.

- If there is no change in data then the vehicle advances in accordance to the previously received significant data. If there is change in the data received then it will analyse the data to identify the change and predict whether the data received is an obstacle or a signal.
- If it's an obstacle, then the ultrasonic sensor helps in finding the distance between the obstacle and the vehicle and find safer distance to stop.
- If it's a traffic signal, then it tries to detect the colour of the signal. If its green then it continues driving but if its red then the vehicle stops and will proceed ahead only if the traffic signal changes again to green.

IV. RESULT AND CONCLUSION

The final vehicular set-up is an autonomous vehicle with self-driving ability. The vehicle with its data acquisition unit sends the collected data over the TCP server to the Processing unit in order to analyse the acquired data frame by frame. The Convolutional neural network model processes the data and successfully generates relevant output which it transmits over to the Controlling unit.

The Controlling unit sends the control data over the radio frequency to the Rf receiver onto the vehicle thus remotely controlling the vehicle without any manual intervention for driving the vehicle. The Autonomous vehicle successfully detects the presence of any obstacles as well as detects the traffic signal lights and the stop sign along its path.

With further training of the CNN model, the vehicle can be deployed in various terrain conditions and different tracks thus, enabling it to learn various driving conditions and predict the relevant actions needed to drive the vehicle.

REFERENCES

1. https://en.m.wikipedia.org/wiki/Raspberry_Pi

2. https://zhengludwig.wordpress.com/projects/self-driving-rc-car/

3. https://media.neliti.com/media/publications/93435-EN-self-driving-car-artificial-intelligence.pdf

4. Implementation of image processing on Raspberry-Pi - K.S. Shilpashree1, Lokesha.H2, Hadimani Shivkumar

5. OpenCV Documentation - Haar Feature-based Cascade Classifier for Object Detection, OpenCV Documentation - Cascade Classifier.

6. Training. NaotoshiSeo - OpenCV haartraining (Rapid Object Detection with A Cascade of Boosted Classifiers Based on Haar-like Features).

7. David J Barnes on Robotics & Mechatronics – OpenCV Haar Training - Object Detection with a Cascade of Boosted Classifiers Based on Haar-like Features - Part I. David J Barnes on Robotics & Mechatronics – OpenCV Haar Training - Object Detection with a Cascade of Boosted Classifiers Based on Haar-like Features - Part II . github.com/foo123/HAAR.js

8. Computer Vision Software - FAQ:

9. OpenCVHaartraining. StackOverflow - Haar training OpenCV