ISSN XXXX XXXX © 2018 IJESC

Research Article Volume 8 Issue No.4

Driverless Automation Vehicle

Manoj Sabnis[1], Siddhesh Deepak Hindalekar[2], Sahil Anil Chanchlani[3], Kunal Ashok Dayaramani[4]
Associate Professor[1], BE Student[2, 3, 4]
Department of Information Technology
V.E.S. Institute of Technology, Chembur, Mumbai, India

Abstract:
The day-by-day increasing demands on the airports due to ever increasing traffic, to fulfill these increasing demands, driverless cars as an alternative to the human driver assisted cars is imperative. Sometimes it becomes a very difficult task for the airport authorities to provide all the services to thousands of passengers at once. This leads them to face flight delays and again taking a step behind to satisfy their customers/passengers. One of the reasons for delay is inconsistency of manpower to provide timely services. Minimizing this human intervention on the automation can help these delays reduce to a lot extent. This paper introduces a pilot model of driverless vehicle that will replace the manpower and the human intervention in the vehicles used on the airport for the transportation of passengers, luggages,etc. from the hub or the waiting room to the aircraft. The system is computed in three modules: the object detection module, camera calibration and neural network training. The integration of the above models is a scope model.

Keywords: Python, OpenCV, Raspberry Pi, Arduino, RC Car, Numpy

1. INTRODUCTION

An automobile is a major expense for most, but driverless automobile is on a wink of day-to-day use. Current trend of automation is of service-oriented architecture where there are a number of services to be offered, integration of all the services are proposed to form a single working service domain or such service domains of interest of the automation research work. Driverless automation works under various domains and services. The location based service is implemented where objects which can move are tracked using the GPS in major domains.This location based tracking as a service is used to match the position of the vehicles so as to be guided accurately under complete control from the source to destination. The working domain selected is an Airport Application where number of vehicles of different type like passenger, load, baggages, etc. are moving on their specified tracks. The project is undertaken to implement the automation on these vehicles and make them “driverless”. However, concerning about the luggage carrier, the work test bed is suggested for loaded carrying vehicles which are to be guided perfectly under complete control from its source to destination on their predesigned path. System designed approach is been selected so as form the pilot platform where the output so obtained is tested for its accuracy and evaluated with comparison with the peer existing system. Finally the conclusion is then followed by requirements of the system domain and the working of the automation; which suggests the future enhancements as per the need.

1.2 System Design Approach:

The paper consists of three units namely: the input unit which includes the camera and the ultrasonic sensor sub-units[10], the processing unit (host/computer) and the Remote Controlled car control unit (wireless RC car)[11].

1.2.1 Input Unit:

The main aspect of the input unit revolves around a Raspberry Pi board (model B+)[1], attached with a pi camera module and an HC-SR04 ultrasonic sensor is used to collect the input data. There are two client programs that run on the Raspberry Pi processor for streaming video and ultrasonic sensor data to the host via a local Wifi connection[4]. The collected video is scaled down to QVGA (320×240) resolution to achieve the low latency of the video streaming[6].

1.2.2 Processing Unit:

The processing unit is the host computer itself that handles important multiple tasks such as receiving the data from Raspberry Pi, training data from the neural network and it’s prediction (steering), object detection, and sending instructions to Arduino through a USB connection[2].

1.2.3 TCP Server:

In this unit, image frames are converted to gray scale and are decoded into numpy arrays. A TCP server program follows a multithreaded approach that runs on the host computer to receive streamed image frames and the ultrasonic data from the Raspberry Pi [9].

1.2.4 Object Detection:

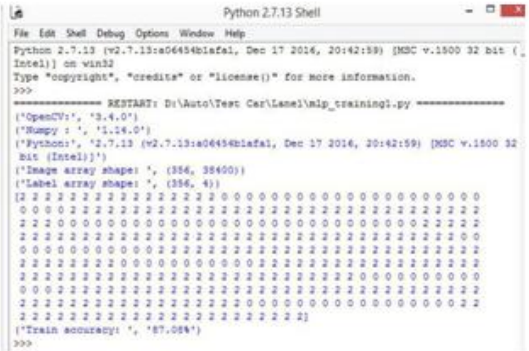
The object detection unit gets simplified to a certain extent with the use of Haar feature-based cascade classifiers for object detection. The Ultrasonic sensor mounted on the front-base of the car detect the object distance with the help of the transceiver mechanism. This economical ultrasonic sensor gives a range of 2 cm to 400 cm of the measurement from the object. It has a ranging accuracy that can reach approximately up to 3mm. This HC-SR04 ultrasonic sensor includes an ultrasonic transmitter, a receiver and a control circuit.[4]

1.2.5 RC Car Control Unit:

The Remote Controlled (RC) car has a switch type wireless remote controller. It has four switch options of forward, reverse, left and right. An Arduino board is used to simulate these kind of button-press actions. The ground pin is kept zero and as soon as any of the button is pressed, the resistance between the relevant chip pin gets active(high). The Arduino board has four pins which are chosen to connect four chip pins on the remote controller, corresponding to forward, reverse, left and right actions corresponding of the button clicks. Arduino pins initially sends LOW signals which indicates the grounding of the chip pins of the controller. Moreover, sending HIGH signal indicates the resistance between chip pins and the ground pin remains the same. Whole of the Arduino Board along with the remote controller circuit is connected to the host computer via USB [2]. The host computer outputs the commands to the Arduino using serial interface, and then the Arduino reads the commands and writes out LOW or HIGH signals, simulating button-press actions to eventually drive the RC car.

1.2. Neural Network

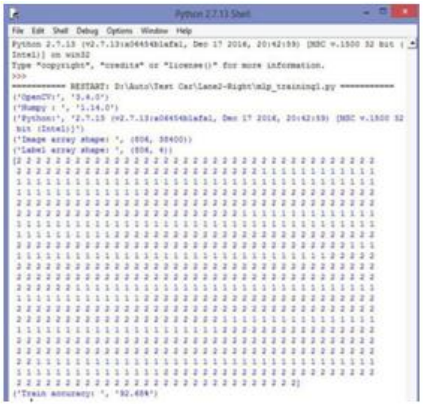
Neural network plays a vital role in training the remote controlled car and to adapt the dynamic varied situations. The main advantage of using neural network is that once the network is trained, it only needs to load trained parameters afterwards, thus prediction becomes very quick[14]. The neural network is fed with the continuous video calibration from the camera mounted on the RC car. However, only the lower half of the input image is used for training and prediction as the upper half is not vital for detection and calibration from the angle of the RC car’s perspective. There are 38,400 (320×120) nodes in the input layer and 32 nodes each in two hidden layers. The number of nodes in the hidden layer is chosen fairly arbitrary. Eventually, there are four nodes in the output layer where each node corresponds to the RC car’s steering control instructions: forward, reverse, left and right. (Note that the reverse control click is not used in the project, but it is still included in the output layer).[13]



```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: D:\Auto\Test Car\Lane1\mlp_training1.py ==============
('OpenCV:', '3.4.0')
('Numpy : ', '1.14.0')
('Python:', '2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32
 bit (Intel)]')
('Image array shape: ', (356, 38400))
('Label array shape: ', (356, 4))
[2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0
 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0
 0 0 0 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
('Train accuracy: ', '87.08%')
>>>
```

Figure: 1.2

Figure 1.2 shows the training output and the accuracy of the data set (video calibration from the camera mounted on the RC car)using neural network. This training accuracy is only for the forward and the left direction functionalities of the RC car)[11].



```
Python 2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32 bit (
Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=========== RESTART: D:\Auto\Test Car\Lane2-Right\mlp_training1.py ===========
('OpenCV:', '3.4.0')
('Numpy : ', '1.14.0')
('Python:', '2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 20:42:59) [MSC v.1500 32
 bit (Intel)]')
('Image array shape: ', (806, 38400))
('Label array shape: ', (806, 4))
[2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2]
('Train accuracy: ', '92.48%')
```

Figure 1.3

Figure 1.3 shows the training output and the accuracy of the dataset using neural network. The training accuracy shown above is only for the forward and the right direction functionalities of the RC car.[11]

2. SOFTWARE APPROACH:

2.1 Algorithm:

STEP 1: Raspberry Pi (Processor) is mounted on the RC car.
STEP 2: Ultrasonic sensor and the camera is mounted at the front base of the car.
STEP 3: Video Calibration capture program is initialized.
STEP 4: Object detection program is initialized.
STEP 5: Video captured images (in frames) are sent to the host computer.
STEP 6: Obstacle detection program starts continuously sending the distance measure between the car and the obstacle (if any) to the host computer.
STEP 7: Host computer will process the image frames of the inputs sent by Raspberry pi and generate the required predicted control signals.
STEP 8: Host computer will then send the corresponding signals to the arduino which is connected to the host computer via USB connection.
STEP 9: Arduino will receive the corresponding signals from the host computer and further will send the processed and corresponding signals to theremote controller of the RC car.
STEP 10: The RC controller (remote) will then eventually send signals to the car through the infrared transmitter for its directions whether forward, reverse, left or right.

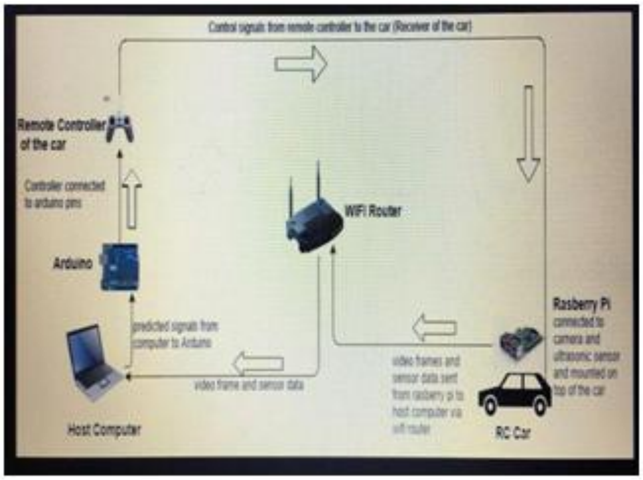

Control signals from remote controller to the car (Receiver of the car)
Remote Controller of the car
Controller connected to arduino pins
Arduino
WIFI Router
predicted signals from computer to Arduino
video frame and sensor data
Host Computer
video frames and sensor data sent from rasberry pi to host computer via wifi router
Rasberry Pi connected to camera and ultrasonic sensor and mounted on top of the car
RC Car

Figure 2.1

The 10-step algorithm mentioned above sums up the working and coordination of the RC car with its hardware and software components. Figure 2.1 shows the pictorial representation of the above mentioned algorithm. Raspberry pi and ultrasonic sensor being mounted on the RC car and the data wirelessly being transmitted to the host computer using a wifi module (router). The Arduino board with the controls of the controller of the RC car via the breadboard is connected to the host computer via an USB. RC car’s remote controller eventually giving the signals to the RC car.

2.2 Testing

2.2.1 Ultrasonic sensor Calibration

The HC-SR04 ultrasonic sensor is mounted at the front base of the RC car to continuously measure the distance between the RC car and the obstacle in the path. The ultrasonic sensor provides accurate distance measurement results, the distance is then sent to the host computer. In fact, the correspondence number to the actual distance, we know when to stop the RC car. Experimental results of detecting distance using ultrasonic sensors are shown below.

2.2.2 Arduino Board circuit and Control Calibration Arduino board is connected to the host computer via the USB to receive the control signals from the raspberry pi via the host computer and eventually signal the car's remote control to give the corresponding directions to it. Basically, Arduino board is used to simulate button-press actions. Figure 2.3 shows the four Arduino pins which are chosen to connect four chip pins on the controller, corresponding to forward, reverse, left and right actions. The remote controller sends the signals to the RC car via the infrared medium. Connections using jumping wires are mounted on a breadboard which acts as a medium between the arduino and the RC remeote. Arduino gets its supply through the host computer and is the ventual controller of the remote of the RC car.[2]

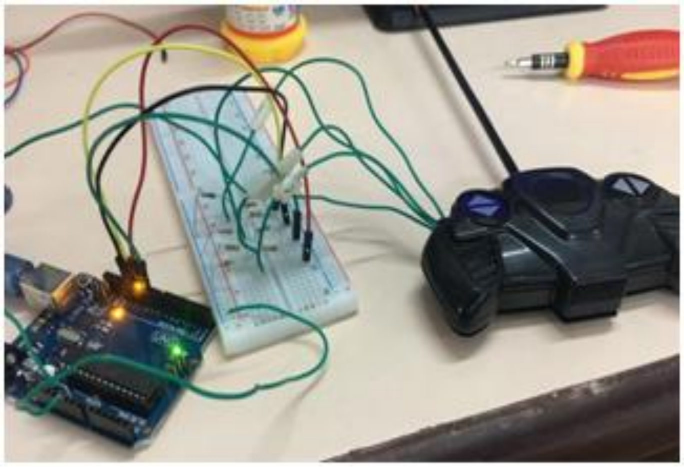
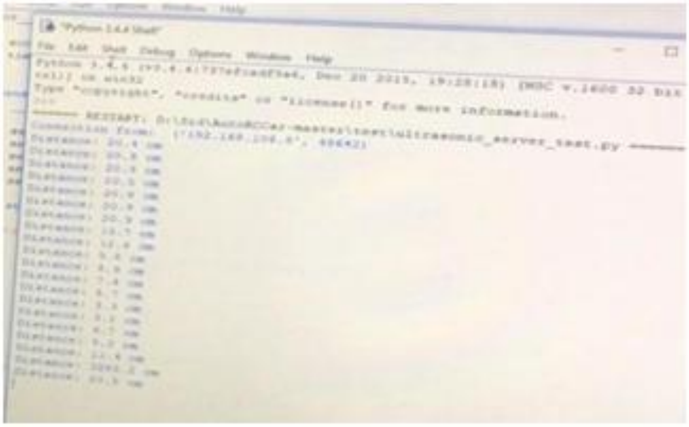


Figure 2.3

3. EVALUATION



```
*Python 3.4.4 Shell*
File  Edit  Shell  Debug  Options  Window  Help
Python 3.4.4 (v3.4.4:737efcadf5a4, Dec 20 2015, 19:28:18) [MSC v.1600 32 bit
(Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
====== RESTART: D:\Sid\AutoRCCar-master\test\ultrasonic_server_test.py ======
Connection from:  ('192.168.106.4', 49642)
Distance: 20.4 cm
Distance: 20.8 cm
Distance: 20.9 cm
Distance: 20.5 cm
Distance: 20.9 cm
Distance: 20.9 cm
Distance: 20.9 cm
Distance: 15.7 cm
Distance: 12.8 cm
Distance: 5.6 cm
Distance: 4.9 cm
Distance: 7.4 cm
Distance: 6.7 cm
Distance: 3.3 cm
Distance: 3.2 cm
Distance: 4.7 cm
Distance: 5.2 cm
Distance: 11.4 cm
Distance: 3291.2 cm
Distance: 20.5 cm
```

Figure 2.2

Figure 2.2 is the output displayed on the host computer outsourced by the ultrasonic sensor of the RC car. The distance measurement is a continuous dynamic process by the ultrasonic sensor to the host computer[4].

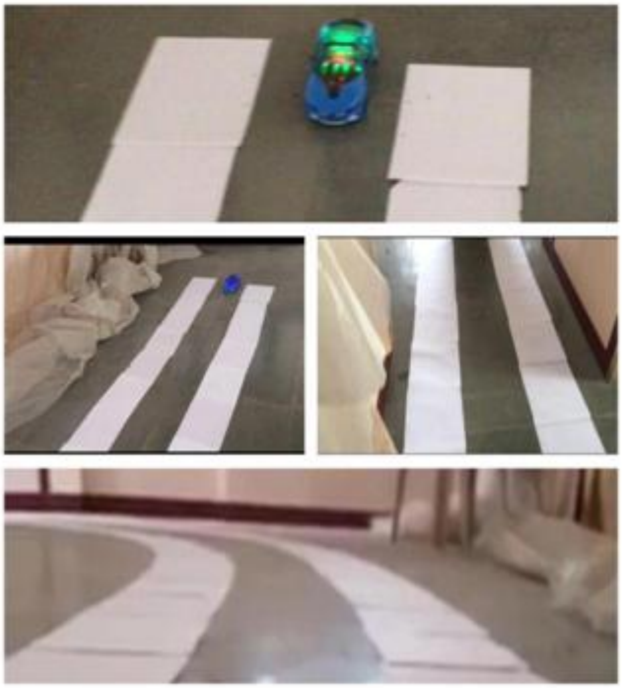


Figure 3.1
Figure 3.1 shows the lane/path of the RC car. The RC car detects the white pages on its either side and runs between those white pages (lanes).

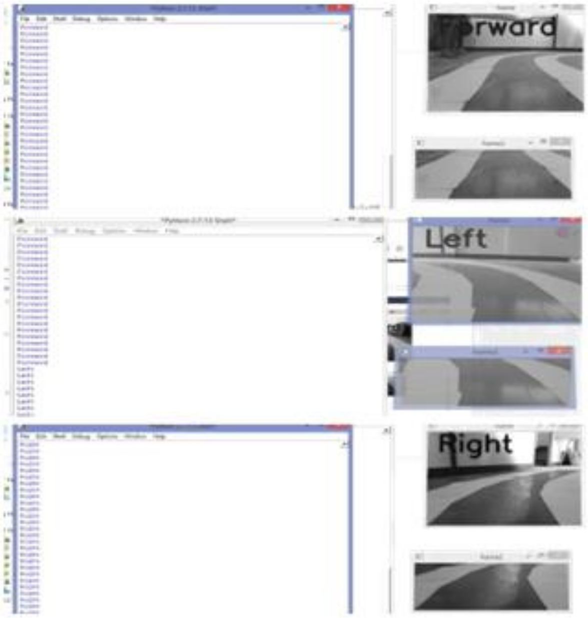


Figure 3.2
Figure 3.2 shows the camera calibration video on the host computer. The RC car continuously detects the white lines and its corresponding functionalities (whether forward, right or left) can be seen on the host computer as shown. Commands of directions and the gray scaled video is dynamic on the host computer.

4. FUTURE SCOPE

Lane detection, object detection, machine learning and neural network play a vital role in the smooth functioning of the RC car on its track, stopping on detecting an object in front of it and moving along the boundaries of the white paper. Integration of the three models (the passenger model, the luggage carrier model and the staircase mode) is a future scope.

5. CONCLUSION

In this paper, a prototype of a self driving robot car is presented. The different hardware components used in the process and their assembly are clearly described. The main feature is to train the RC car and to detect its road lanes indicated by white lines on both the sides, eventually make appropriate prediction for the direction which the RC car takes (ie forward, left or right). Using the ultrasonic sensor, collision with the obstacles in front of the RC car is detected and avoided. The neural algorithms mentioned in the paper have been successfully implemented .

6. REFERENCES

(1) Hardware Components:

[1] .Raspberry Pi. Rohit Tiwari Dushyant Kumar Singh Vehicle Control Using Raspberry Pi and Image Processing, Lovely Professional University, Punjab. --https:// www .pyimagesearch .com/2015/02/23/install-opencv-and-python-on-your-raspberry-pi-2-and-b/

[2]. Arduino Board Establishing the Connection between Control Theory Education and Application: An Arduino Based Rapid Control Prototyping Approach Hongtai Cheng, Lina Hao, Zhong Luo, and Fei Wang Department of Mechanical Engineering, Northeastern University, Shenyang, China

[3].Object detection using Haar-cascade Classifier - Sander Soo: Institute of Computer Science, University of Tartu

[4]. Ultrasonic sensor. Interfacing hc-sr04 ultrasonic sensor with raspberry pi[internet]. [cited 2016 November 14]. Available from:https://electrosome.com/hc-sr04-ultrasonic-sensor-ras pberry-pi/. Johann Borenstein & Yoram Koren, Obstacle Avoidance with Ultrasonic Sensors, IEEE JOURNAL OF ROBOTICS AND AUTOMATION, VOL. 4, NO. 2, APRIL I988, pp. 213-218

[5]. OpenCV Python Tutorial. H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski, Self-supervised monocular road detection in desert terrain. G. Sukhatme, S. Schaal, W. Burgard, and D. Fox, editors & Proceedings of the Robotics Science and Systems Conference, Philadelphia, PA, 2006.

(2) Software Approach

[6] .Videos (OpenCV with python for video and image processing). OpenCV Python Tutorials Documentation Release 1, Alexander Mordvintsev & Abid K, Nov 5, 2017

[7]. Raspberry pi and CV2 update/install commands. --https://www.pyimagesearch.com/2015/02/23/install-opencv-and-python-on-your-raspberry-pi-2-and-b/

[8] .Python basics and sockets - python 3.4 documentation [online]. --https://www.ibm.com/developerworks/linux/tutorials/l-pysocks/index.html

[9] .TCP/IP Client and Server. Socket Programming, Limi Kalita, M. Tech Student, Department of Computer Science and Engineering, Assam Down Town University, Guwahati, India. --http://www.bogotobogo. com/python/python_ network_ programming_ server_client_file_transfer.php

[10]. Camera in raspberry pi. [online] --https:// www.pyim agesearch. com/2016/01/18/multiple-ca meras-with-the-rasp berr y -pi-and-opencv/--https://www.raspberrypi.org/docu mentat ion/usage/webca ms/ --http://www.chioka.in/python-live-video-streaming-example/

[11] .Wirelessly streaming a video from a Raspberry to a remote laptop [online] --https:// www.youtube. com/watch?v =sYGdge 3T30o

[12] Camera Calibration - opencv documentation [online] --http://opencv-python-tutroals.rea dthedocs.io/ en/latest/py_ tutorials/py_calib3d/py _calibration/py_calibration.html--https:// stackoverflow.com/questions/42308270/python-numpy-mgrid-and-reshape

(3) Machine Learning and Neural Network [online]

[13] OpenCV Neural Networks Documentation --http://search networking.techtarget.com/definition/neural-network--http:// do cs. opencv.org/2.4/modules/ml/doc/neural_networ ks.html

[14]. Training and algorithms. CRISTIAN-TUDOR TUDORAN, VICTOR-EMIL NEAGOE, A New Neural Network Approach forVisual Autonomous Road, Depart. Electronics, Telecommunications & Information Technology Polytechnic University of Bucharest Splaiul Independentei No. 313, Sector 6, Bucharest ROMANIA