ARTIFICIAL NEURAL NETWORK BASED AUTOMATED ESCALATING TOOLS FOR CRISES NAVIGATION

A PROJECT REPORT

Submitted by

GOKUL.S (410814104022)

MURUGAN.V (410814104035)

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

GKM COLLEGE OF ENGINEERING AND TECHNOLOGY
CHENNAI-63

ANNA UNIVERSITY::CHENNAI 600 025

APRIL 2018

ANNA UNIVERSITY::CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**ARTIFICIAL NEURAL NETWORK BASED AUTOMATED ESCALATING TOOLS FOR CRISES NAVIGATION"** is a bonafide work of **Mr. S.GOKUL (410814104022)** and **Mr. V.MURUGAN (410814104035)** who carried out the project work under my supervision.

SIGNATURE

Mr. S.THIRUMAL, M.E.
HEAD OF THE DEPARTMENT

Department of Computer Science and Engineering

GKM College of Engineering and Technology,

GKM Nagar, New Perungalathur, Chennai- 600 063

SIGNATURE

Dr.R. INDRA GANDHI, MCA., Ph.D., M.Tech., Ph.D.
SUPERVISOR
PROFESSOR

Department of Information Technology

GKM College of Engineering and Technology,

GKM Nagar, New Perungalathur, Chennai-600 063

GKM COLLEGE OF ENGINEERING AND TECHNOLOGY

CHENNAI-600 063

ANNA UNIVERSITY::CHENNAI-600 025

PROJECT VIVA-VOCE EXAMINATION

The viva-voce Examination of the project work was submitted by,

GOKUL.S	**410814104022**
MURUGAN.V	**410814104035**

is to be held on ..... at Department of Computer Science and Engineering, GKM College of Engineering and Technology, Chennai-63.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are grateful to our **Chairman Dr. G.Kathamuthu, M.A.,** who took keen interest in our studies and encouraged us in all our efforts throughout this course.

We are grateful to our **Advisor Mr. C.Balasubramanian, M.P.Ed.,** for his kind cooperation, moral support and facilities given.

We express our deep gratitude to our **Chief Executive Officer Dr. Sujatha Balasubramanian, M.B.A, Ph.D.,** who was the main inspiration behind our project.

We would like to express our sincere thanks to our **Principal Dr. C.Chellappan, Ph.D.,** for his support he extended to us for our project.

We would like to express my sincere thanks to our beloved **Head of the Department Mr. S.Thirumal, M.E.,** who encouraged us for the completion of my project successfully.

We are highly indebted to our **Project Guide Dr.R.INDRA GANDHI, Ph.D,** for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

And we also thank my department staff in developing the project and people who have willingly helped me out with their abilities.

ABSTRACT

Autonomous driving technology has made significant advances in recent years. In order to make self-driving cars more practical, they are required to operate safely and reliably even under adverse driving condition. The object detection based on deep learning is an important application in deep learning technology, which is characterized by its strong capability of features learning and feature representation compared with the traditional object detection method. After analyzing the characteristics of videos shot by the camera, we choose to use deep learning to train a vehicle detection model to detect targets in video. In the end, we use trained data set to control the speed and navigate the vehicle in crises situations. Conversely, not much research is going on of usage such networks for elaborating of real time data. The goal of this work is exploring, experimenting and providing new approaches of classification non-stationery data using neural network.

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
ReLU	RECTIFIED LINEAR UNIT
CNN	CONVLOUTIONAL NEURAL NETWORK
HMM	HIDDEN MARKOV MODEL
PNN	PROBABILISTIC NEURAL NETWORK
ANN	ARTIFICIAL NEURAL NETWORK
GRNN	GENERAL REGRESSION NEURAL NETWORK

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 DOMAIN INTRODUCTION

Artificial intelligence, deep learning, and neural networks represent incredibly exciting and powerful machine learning-based techniques used to solve many real-world problems. While human-like deductive reasoning, inference, and decision-making by a computer is still a long time away, there have been remarkable gains in the application of AI techniques and associated algorithms. The primary motivation and driving force for these areas of study, and for developing these techniques further, is that the solutions required to solve certain problems are incredibly complicated, not well understood, nor easy to determine manually.

1.2 PROJECT INTRODUCTION

Autonomous Driving has been said to be the next big disruptive innovation in the years to come. Considered as being predominantly technology driven, it is supposed to have massive societal impact in all kinds of fields. Having a closer look at the history of Autonomous Driving, as explained in the IEEE Spectrum it can be observed that the technological development and main milestones of the autonomous driving field started already a few decades ago. Leading to a vast analysis of some semi-autonomous features, development of present technologies and understanding on the future problematic while focusing the near future in the connected car. Full automated car is not accepted by many

countries and doesn’t show high accuracy in real life. It doesn’t mean humans are better drivers, human drivers face many situations where they lose their human ability e.g. drunk and drive, drowsiness etc. For improving the safety and a new method has been proposed. The method includes the combination of deep-learning object detection and human activity monitoring to make self -driving more reliable.

1.3 SCOPE OF THE PROJECT

Compared to past 10 years, today, the idea of self-driving vehicles has started to become less far-fetched. Major automakers plan to get their autonomous vehicles on the road by 2025 which is less than a decade away. The questions are many and the answers too little. In this report, we take an in-depth study to understand the crucial role the automobile industry, geospatial industry and the government together play in bringing evolution in the autonomous vehicles industry. The report has been produced as a knowledge initiative to answer the ‘mystery’ questions around the role of geospatial information in self-driving vehicles, the role of the autonomous vehicle ecosystem and the government,

The study, therefore, is largely directed at all the stakeholders of the driverless car ecosystem and the reader would gain insights into:

- The concept of self-driving vehicles
- Current market trends and benefits and costs associated with driverless cars
- Action points of the self-driving cars ecosystem
- Role of the geospatial industry

- The defining role of the government

The report, in summary, examines the core of the driverless car ecosystem i.e. the data and how the other defining factors of the self-driving cars ecosystem form an inter-linkage to transform the complete landscape of the transportation industry in the near future. In future it is very useful for disabled person, senior citizens and reduces the number of road accidents. Since we are using semi automation system, manufacturing cost will be reduced, and it can be used for commercial purpose.

CHAPTER 2

SYSTEM ANALYSIS

The system analysis is "the process of studying a procedure or business in order to identify its goals and purposes and create systems and procedures that will achieve them in an efficient way". Another view sees system analysis as a problem-solving technique that decomposes a system into its component pieces for the purpose of the studying how well those component parts work and interact to accomplish their purpose. Analysis and synthesis, as scientific methods, always go hand in hand; they complement one another. Synthesis builds upon the results of a preceding analysis, and every analysis requires a subsequent synthesis in order to verify and correct its results. It is also "an explicit formal inquiry carried out to help a decision maker identify a better course of action and make a better decision than she might otherwise have made". The terms analysis and synthesis stem from Greek, meaning "to take apart" and "to put together," respectively. These terms are used in many scientific disciplines, from mathematics and logic to economics and psychology, to denote similar investigative procedures. Analysis is defined as "the procedure by which we break down an intellectual or substantial whole into parts," while synthesis means "the procedure by which we combine separate elements or components in order to form a coherent whole". System analysis is used in every field where something is developed. Analysis can also be a series of components that perform organic functions together, such as system engineering. System engineering is an interdisciplinary field of engineering that focuses on how complex engineering projects should be designed and managed.

2.1 LITERATURE SURVEY

[1]. A Self-Driving Car that can Handle Adverse Weather Conditions

AUTHORS: Unghui Lee, Jiwon Jung, Seunghak Shin, Yongseop Jeong, Kibaek Park, David Hyunchul Shim

Overview:

Autonomous driving technology has made significant advances in recent years. In order for self-driving cars to become practical, they are required to operate safely and reliably even under adverse driving conditions. However, most current autonomous driving cars have only been shown to be operational under amiable weather conditions, i.e., on sunny days on dry roads. In order to enable autonomous cars to handle adverse driving conditions such as rain and wet roads, the algorithm must be able to detect roads within a tolerable margin of error using sensors such as cameras and laser scanners. we propose a sensor fusion algorithm that is able to operate under a variety of weather conditions, including rain. we present the competition results that were collected on the same course on both sunny and rainy days. Based on the comparison, we propose the future directions to improve the autonomous driving capability under adverse environmental conditions.

Advantage:

- Common method that crops a plane out of a 3D scanning point cloud is RANSAC-based plane fitting. To reduce computing time, we applied RANSAC-based plane fitting separately per vertical layer of the LiDAR.
- Another important role of 3D scanning LiDAR in our system is to detect the lane and convert the 3D scanning cloud in RANSAC and used for obstacle detection.

Disadvantage:

- The LIDAR should be placed at top of the car, because LIDAR need to receive the signal from satellite, In case of any damage in LIDAR the whole system get failure, in heavy rain, bad weather, heavy wind the LIDAR get malfunction or damage.

[2]. Self-Driving and Driver Relaxing Vehicle

AUTHORS: Qudsia, Muzamil Ahmed, Shahzeb Ali, Azam Rafique Memon

Overview:

The vehicles are focused to be automated to give human driver relaxed driving. In the field of automobile various aspects have been considered which makes a vehicle automated. In this paper we have focused on two applications of an automated car, one in which two vehicles have same destination and one knows the route, where other don't. The following vehicle will follow the target (i.e. Front) vehicle automatically. The other application is automated driving during the heavy traffic jam, hence relaxing driver from continuously pushing brake, accelerator or clutch. The idea described in this paper has been taken from the Google car, defining the one aspect here under consideration is making the destination dynamic. This can be done by a vehicle automatically following the destination of another vehicle. Since taking intelligent decisions in the traffic is also an issue for the automated vehicle so this aspect has been also under consideration in this paper.

Advantage

- In self-driving car a solution to relax the driver by making vehicle smart

enough to make decisions automatically and move by maintaining a specified distance from vehicles and obstacles around.

- When two vehicles have the same destination but one of the drivers doesn't know its route. The driver can make his vehicle follow the front vehicle if they are known and share their location to reach the same destination

Disadvantage

- The front vehicle signal or gps location can disappear when during heavy rain
- Location or direction should change, the problem should have created in direction which is the way? should we go.

[3]. Road Marking Recognition with Computer Vision System

AUTHORS: Konstantin V. Ignatiev, Elena V. Serekh, Anna V. Mironiuk

Overview:

In this paper control system of autonomous car-like mobile robot on road with markings is considered. Tasks of road markings recognition with usage of OpenCV library and robot motion control according to received data after image processing is realized. And then successful tests were taken with four-wheeler mobile robot on practice ground with road markings.

Advantage:

- Road marking recognition system uses images obtained from camera and processes them with functions from OpenCV library.
- Then car should obtain the road marking image from camera and then follow the lane marking system and stop where the lane marking area should over.

Disadvantage

- In this paper obstacle can't be detected in the road marking recognition

[4]. End-to-End Learning for Lane Keeping of Self-Driving Cars

AUTHORS: Zhilu Chen and Xinming Huang

Overview:

Lane keeping is an important feature for self-driving cars. This paper presents an end-to-end learning approach to obtain the proper steering angle to maintain the car in the lane. The convolutional neural network(CNN)model takes raw image frames as input and outputs the steering angles accordingly. which contains the front view image frames and the steering angle data captured when driving on the road. The end-to-end model can directly steer the vehicle from the front view camera data after training. It learns how to keep in lane from human driving data. Many sensors installed on autonomous cars such as radar, LiDAR, ultrasonic sensor and infrared cameras, the ordinary color cameras are still very important for their low cost and ability to obtain rich information.

[5]. A Control Strategy of Autonomous Vehicles based on Deep Reinforcement Learning

AUTHORS: Wei Xia1, Huiyun Li1, Baopu Li2,1

Overview:

Deep reinforcement learning has received considerable attention after the outstanding performance of AlphaGo. In this paper, we propose a new control strategy of self-driving vehicles using the deep reinforcement learning model, in which learning with an experience of

professional driver and a Q-learning algorithm with filtered experience replay are proposed. Experimental results demonstrate that the proposed model can reduce the time consumption of learning by 71.2%, and the stability increases by about 32%, compared with the existing neural fitted Q-iteration algorithm

Advantage:

- The classic neural fitted Q-iteration and deep learning technique to obtain a control strategy with less resource consumption.
- Deep reinforcement learning provides a promising direction, we propose a Q-learning algorithm for self-driving vehicles to obtain effective control strategies.

Disadvantage:

- Developing the control strategy for self-driving vehicles is difficult, due to enormous circumstances should be taken into account.

2.2 SYSTEM ANALYSIS

- In this section, we discuss about the two things, one is existing system of the project (which is implemented in a real time in different areas) and second is our project idea which we proposed, and which is going to implement.

2.2.1 Existing system

An automated driving system where perception, decision making and of the automobile are performed by electronics and machinery instead of a human driver, and as introduction of automation into road traffic. Now, the GOOGLE Inc was introduced the real time existing system of fully automated car in America for testing purpose, but unfortunately the

automated system was not considered by US government to ride in road for some safety recognized. Right now, self driving cars can only be tested in Washington., Nevada, Florida, Michigan and California, but research and testing are still going. Technology, google and the parent company invented the google robotic and LIDAR system. The range finder mounted on the top is a velodyne beam laser. The laser allows the vehicle to generate a detailed 3D map of its environment. The car then takes these generated maps and combines them with high resolution maps of the world, producing different type of data models that allow it to drive.

Drawbacks of Existing system:

It only supports fully automated systems, which is not accepted worldwide, and its production cost is very high.

2.2.2 Proposed system:

The proposed system developed to drive cars in an efficient and safer way. This system does not support for fully automated self-drive car. Instead, we are likely to present the semi-automatic car for some CRISES situation. The semi-automatic car can be easily interacted with human beings in emergency. [(Ex) In Drunk and Driving situation, car would have 85% of the control, the driver can still drive the car but within a limited speed. During turning or Heavy Traffic, the car will detect the traffic signal using pre-trained model and make decisions according to the situation].

Advantages of Proposed system:

i. Drunk and driving incidents should decrease, because there's no designated driver needed when the car drive itself.
ii. Automatic driving, like autonomous braking, self-parking, sensors that clue a driver in to a nearby obstacles.
iii. Over time, higher speed limits might be considered as an option if more people are using self-driving cars. The computer calculates operation of the vehicle safely, driving time could be reduced by faster speeds allowed on the road.
iv. Accident is become less in number.
v. There is a less of concern about taking the keys away from the grandma when she gets too old to drive carefully the car will take care of her.

2.3 SYSTEM REQUIREMENTS

2.3.1 Software Requirements

- Operating system : windows 10, 64 – Bit.
- Tool : MATLAB, OpenCV, TensorFlow, python IDE, Anaconda

2.3.2 Hardware requirements

- System processor : Intel CORE i5 2.30GHZ.
- Hard Disk : 1 TB.
- RAM : 4 GB (minimum).
- Monitor : 15.6” VGA Colour.
- Webcam : minimum 8Mp
- Sensors : Arduino, RC Car, Alcohol detector

CHAPTER 3

SYSTEM DESIGN

3.1 SYSTEM DESIGN:

3.1.1 Architecture diagram:

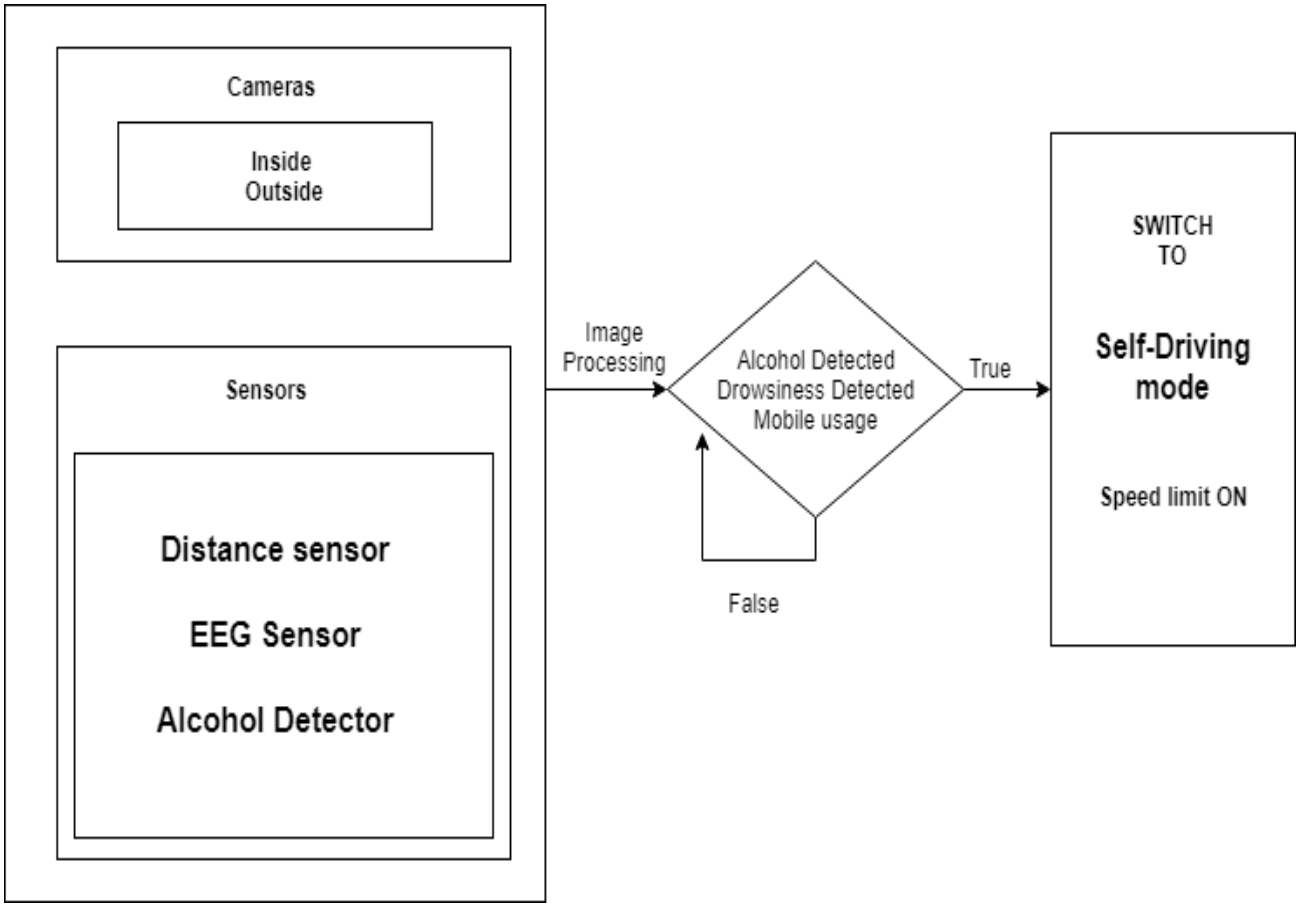


Fig.1: System Architecture of Automated Navigator

The above architecture diagram gives the full working module of the system. In that we can see four main blocks are the four major modules and each of the modules have their sub modules (processes) like object detection, image processing, etc.

3.2 UML DIAGRAMS

3.2.1 Usecase diagram:

Usecase diagram is a methodology used in system analysis to identify, clarify and organize system requirements. The use case is made up of a set of possible sequences of interactions system and users in a particular environment and related to a particular goal. It is represented using ellipse. Actor is any external entity that makes use of the system being modelled. It is represented using stick figure.

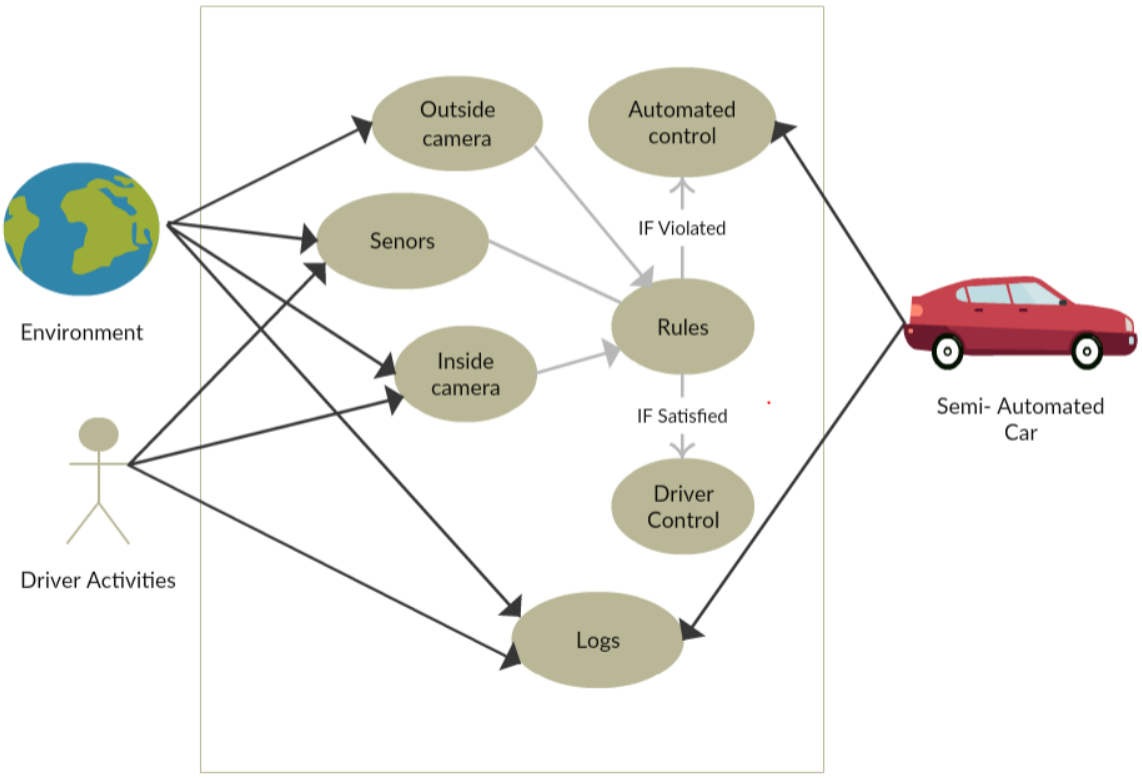


Fig.2: Usecase Diagram of Automated Navigation System

A usecase diagram can identify the different types of users of a system and different use cases. In the above diagram, the interaction between driver and the car in terms of various modules is described as use cases.

3.2.2 Class diagram:

A class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes and relationships between the classes. It is represented using a rectangle with three compartments the attributes and the bottom compartment with operations.

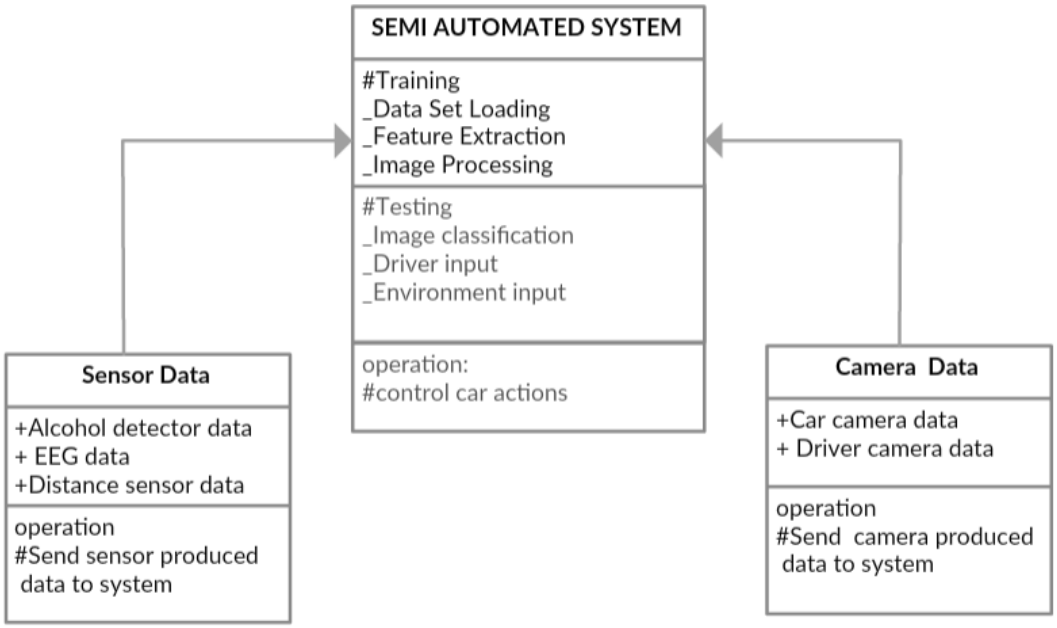


Fig.3: Class Diagram of Automated Navigation System

It is used both for general conceptual modelling of the systematic of the application, and for detailed modelling translating the models into programming code. In this class diagram, we have a Sensor data with

some attributes, Camera data with some attributes and the semi-automated system with the conversion.

3.2.3 Sequence diagram:

Sequence diagram is a Unified Modelling Language (UML) is a kind of an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a message sequence chart. There are two dimensions

1. Vertical dimension – represent time.
2. Horizontal dimension – represent different objects.

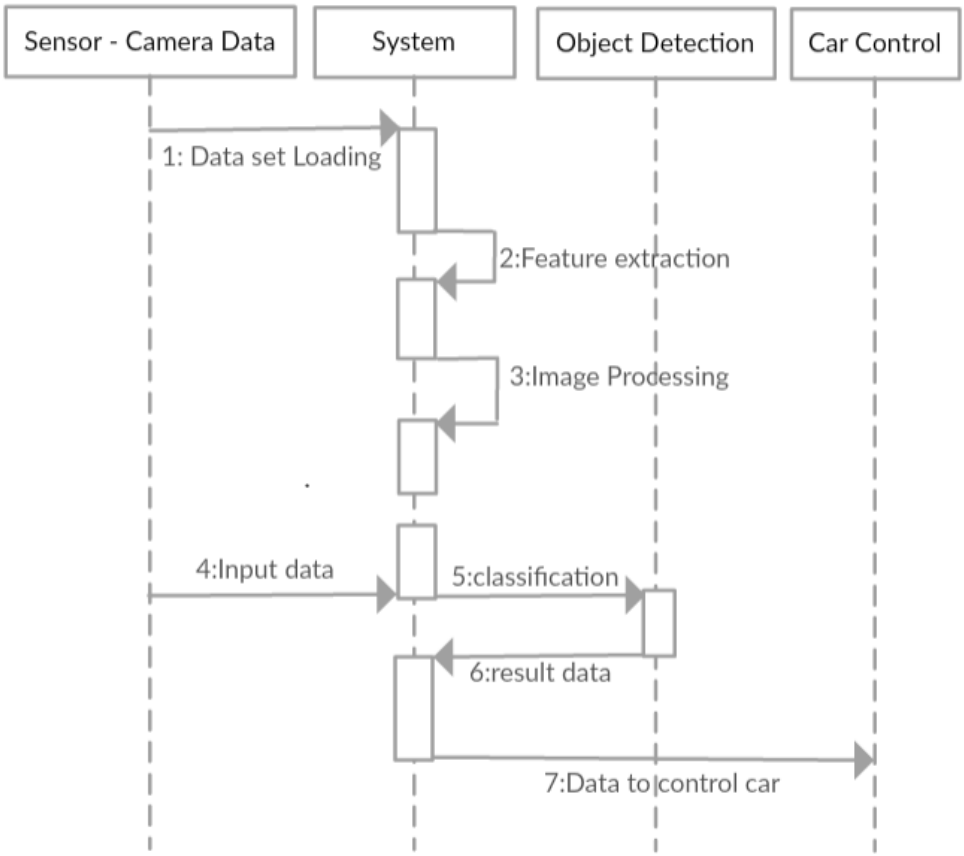


Fig.4: Sequence Diagram of Automated Navigation System

It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. It is typically associated with use case

realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.

3.2.4 Collaboration diagram:

Collaboration diagram also called as communication diagram or interaction diagram. A sophisticated modelling tool can easily convert a collaboration diagram into a sequence diagram and vice versa. A collaboration diagram resembles a flowchart that portrays the roles, functionality and behaviour of individual objects as well as the overall operations of the system in real time.

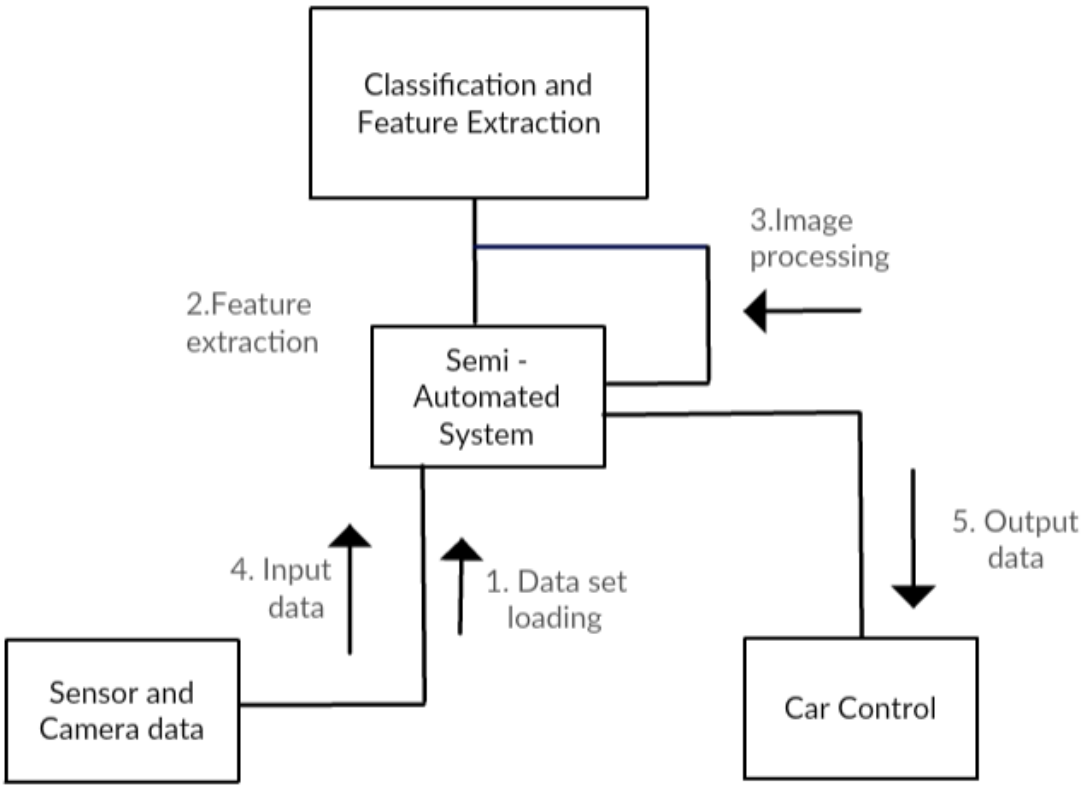


Fig.5: Collaboration Diagram of Automated Navigation System

In collaboration diagram, sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages. The method calls are similar to that of a sequence diagram.

3.2.5 State chart diagram:

The purpose of the state chart diagram is to understand the algorithm involved in performing a method. It is also called as state diagram. A state is represented as a round box, which may contain one or more components. An initial state is represented as small dot. A final state is represented as circles surrounding a small dot.

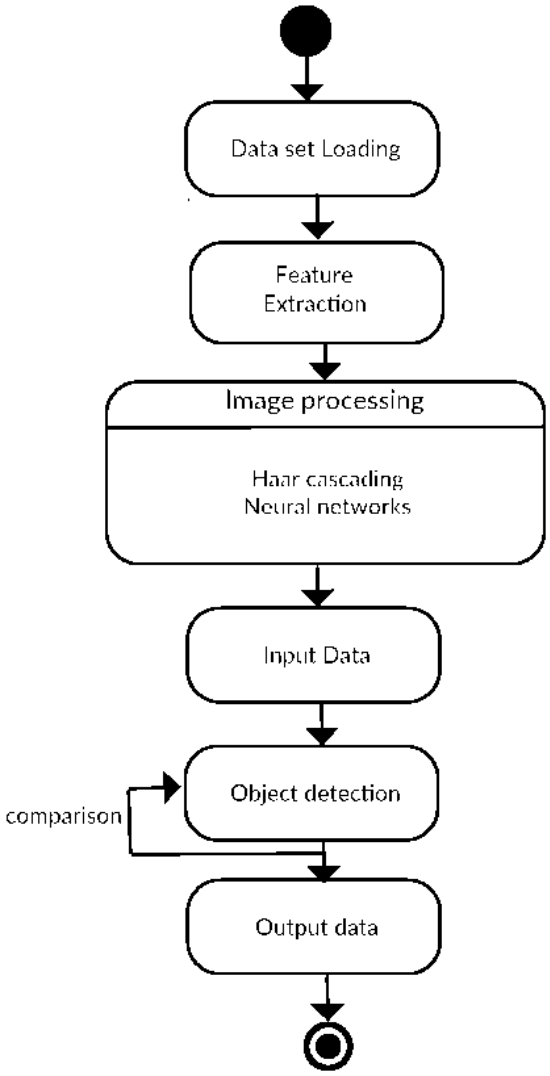


Fig.6: State Chart Diagram of Automated Navigation System

It describes the flow of control from one state to another state. States are defined as a condition in which an object exists, and it changes when some event is triggered. The most important of state chart is to model life time of an object from certain to termination.

3.2.6 Activity diagram:

Activity diagrams are graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagram can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control. An activity is shown as a rounded box containing the name of the operation.

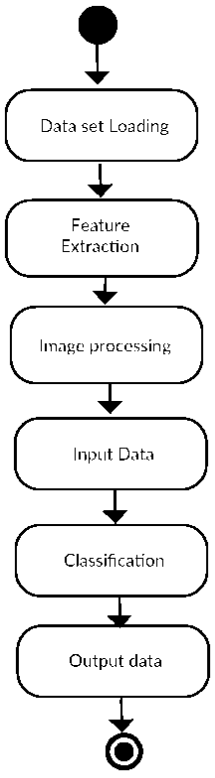


Fig.7: Activity Diagram of Automated Navigation System

It captures the dynamic behaviour of the system. Activity diagram are not only used for visualizing dynamic nature of a system.

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1. LIST OF MODULES:

Modules used in this system are listed below:

- **Data Phase:**
  - ✓ Camera data
  - ✓ Sensor data
- **Training Phase:**
  - ✓ Dataset Loading
  - ✓ Image processing
  - ✓ Data Correction
- **Testing and Automation Phase:**
  - ✓ Autonomous mode
  - ✓ Traffic sign Detection

4.2 MODULE DESCRIPTION:

We are using two models, one for self-driving and another one for detecting traffic signals. From the above list our modules are classified as three major phases such as “Data phase”, “Training phase” and “Testing phase”. In the training phase we train the dataset (set of Traffic signals) by image processing methods. And in the testing phase we are going to check the input images and trained system by using neural networks.

4.3 Data PHASE:

4.3.1. Sensor and camera data (Input data):

Model 1: Self driving car:

In the model we are using data produced by the steering wheel and the camera. Two types of files are produced jpg image files from three cameras and csv files to storing steering angle and speed of the car.

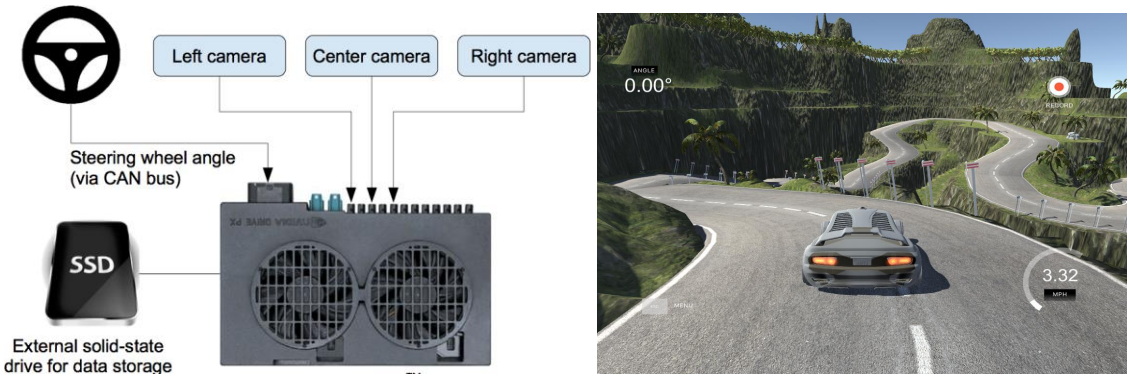


Fig.8: Data from Steering Wheel and Cameras

Model 2: Traffic sign detection using OpenCV

We collected around 1000 images of traffic signals and traffic sign boards, which is then feed into Haar cascade for generating xml files.

Fig.9: Data for Haar Cascading

4.4 TRAINING PHASE:

4.4.1. Dataset Loading:

Model 1: Self driving car:

Images are fed into a CNN that then computes a proposed steering command. The proposed command is compared to the desired command for that image, and the weights of the CNN are adjusted to bring the CNN output closer to the desired output. The weight adjustment is accomplished using back propagation.

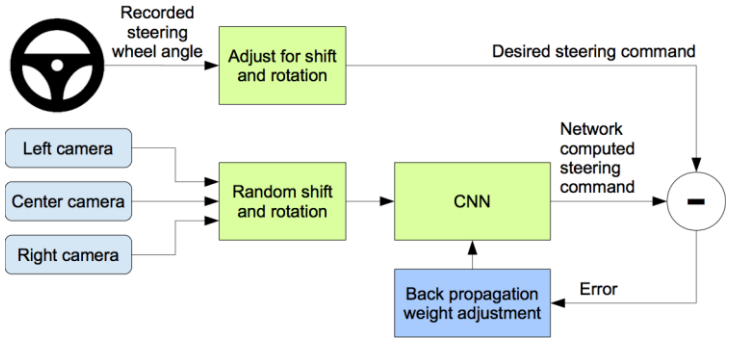


Fig.10: Process of CNN

4.4.2 Object detection:

Model 2: Traffic sign detection:

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. Here we will work with face detection. Initially, the algorithm needs a lot of positive images and negative images to train the classifier. Then we need to extract features

from it. For this, hear features shown in below image are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under white rectangle from sum of pixels under black rectangle.

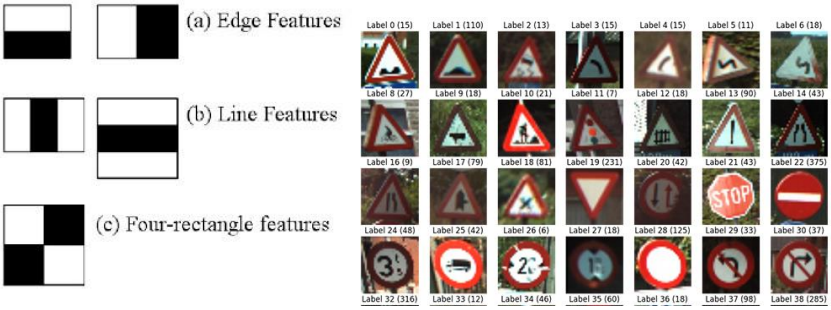


Fig.11: Process of Haar Cascading

4.5 TESTING PHASE:

The testing phase results the prediction and accuracy of the model that has been built.

4.5.1 Autonomous mode:

Model 1: Self driving car:

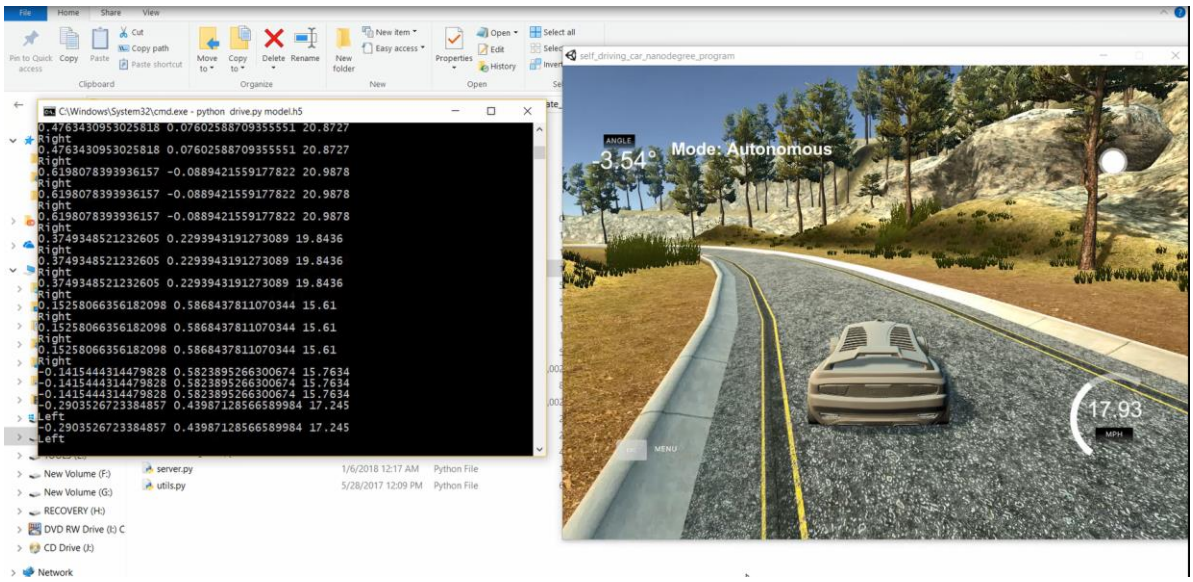


Fig.12: Autonomous Mode

In training phase, a predefined model (model.h5 file) using that the car drives autonomously. The process is shown below fig:

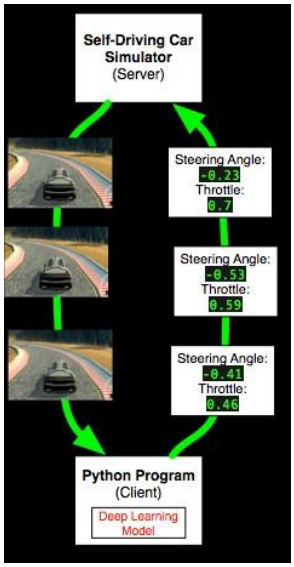


Fig.13. Process of Autonomous Mode

4.5.1 Object detection:

Model 2: Traffic sign detection:

Model 2 was build using a RC car, which is controlled by Arduino connected to a computer. This process contains 4 steps:

(i) Input data:

A camera is connected to the RC car which send the video (series of images) to server i.e. Laptop.

(ii) Image processing:

For image processing we are using the predefined model produced during training phase. Each image is converted into greyscale and resolution is change.

(iii) Object detection:

Using Haar cascading and predefined model object in video is detected.

(iv) Actions:

If an object is detected actions are performed. Action are the selecting according to predefined rules.

Fig.14. Process of Object Detection

4.6 **CONVOLUTIONAL NEURAL NETWORKS:**

Neural networks are predictive models loosely based on the action of biological neurons. The selection of the name “neural network” was one of the great PR successes of the Twentieth Century. It certainly sounds more exciting than a technical description such as “A network of weighted, additive values with nonlinear transfer functions”. However, despite the name, neural networks are far from “thinking machines” or “artificial brains”. A typical artificial neural network might have a hundred neurons. In comparison, the human nervous system is believed to have about $3x10^{10}$ neurons. We are still light years from “Data”.

4.6.1 Types of Neural Networks:

1) Feedforward Neural Network
2) Radial basis function Neural Network
3) Kohonen Self Organizing Neural Network
4) Convolutional Neural Network
5) Modular Neural Network
6) Recurrent Neural Network

Biological Neural Networks Overview:

The human brain is exceptionally complex and quite literally the most powerful computing machine known. The inner-workings of the human brain are often modeled around the concept of neurons and the networks of neurons known as biological neural networks. According to Wikipedia, it's estimated that the human brain contains roughly 100 billion neurons, which are connected along pathways throughout these networks. At a very high level, neurons interact and communicate with one another through an interface consisting of axon terminals that are connected to dendrites across a gap (*synapse*) as shown here.

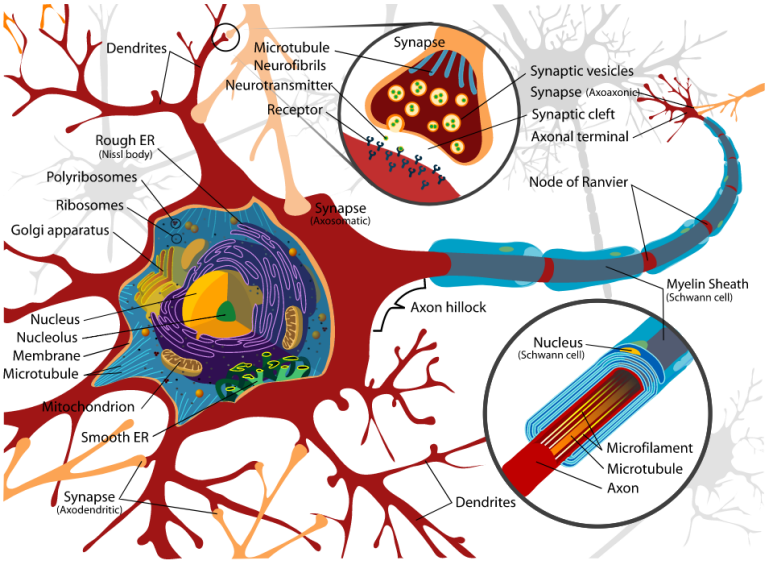


Fig.15: Biological Neurons

In general English, a single neuron will pass a message to another neuron across this interface if the sum of *weighted input signals* from one or more neurons (summation) into it is great enough (exceeds a threshold) to cause the message transmission. This is called activation when the threshold is exceeded, and the message is passed along to the next neuron.

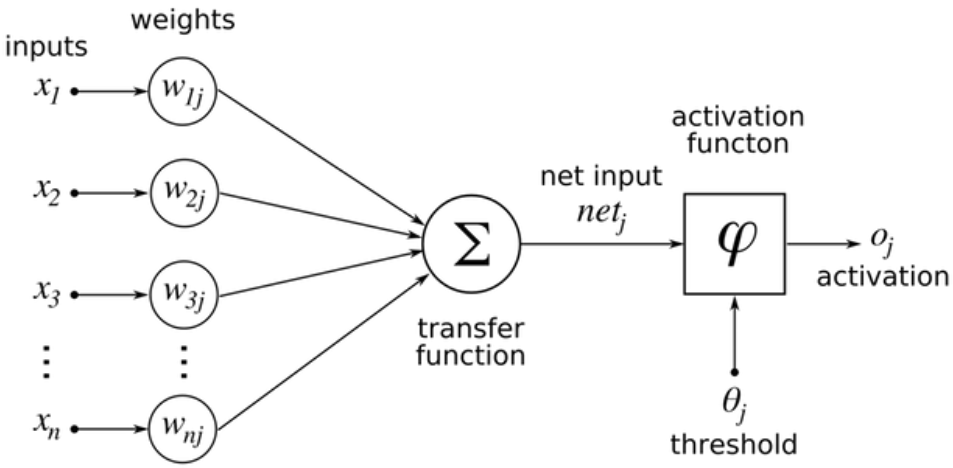


Fig.16: Summation Process of Multiple Neurons

The summation process can be mathematically complex. Each neuron's input signal is actually a weighted combination of potentially many input signals, and the weighting of each input means that that input can have a different influence on any subsequent calculations, and ultimately on the final output of the entire network. In addition, each neuron applies a function or transformation to the weighted inputs, which means that the combined weighted input signal is transformed mathematically prior to evaluating if the activation threshold has been exceeded. This combination of weighted input signals and the functions applied are typically either linear or nonlinear.

These input signals can originate in many ways, with our senses being some of the most important, as well as ingestion of gases (breathing), liquids (drinking), and solids (eating) for example. A single neuron may receive hundreds of thousands of inputs signals at once that undergo the summation process to determine if the message gets passed along, and ultimately causes the brain to instruct actions, memory recollection, and so on. The 'thinking' or processing that our brain carries out, and the subsequent instructions given to our muscles, organs, and body are the result of these neural networks in action. In addition, the

brain's neural networks continuously change and update themselves in many ways, including modifications to the amount of weighting applied between neurons. This happens as a direct result of learning and experience. Given this, it's a natural assumption that for a computing machine to replicate the brain's functionality and capabilities, including being 'intelligent', it must successfully implement a computer-based or artificial version of this network of neurons. This is the genesis of the advanced statistical technique and term known as artificial neural networks.

Artificial Neural Networks Overview:

Artificial neural networks (ANNs) are statistical models directly inspired by, and partially modeled on biological neural networks. They are capable of modeling and processing nonlinear relationships between inputs and outputs in parallel. The related algorithms are part of the broader field of machine learning and can be used in many applications as discussed. Artificial neural networks are characterized by containing adaptive weights along paths between neurons that can be tuned by a learning algorithm that learns from observed data in order to improve the model. In addition to the learning algorithm itself, one must choose an appropriate cost function. The cost function is what's used to learn the optimal solution to the problem being solved. This involves determining the best values for all of the tunable model parameters, with neuron path adaptive weights being the primary target, along with algorithm tuning parameters such as the learning rate. It's usually done through optimization techniques such as gradient descent or stochastic gradient descent. These optimization techniques basically try to make the ANN solution be as close as possible to the optimal solution, which when

successful means that the ANN is able to solve the intended problem with high performance.

Architecturally, an artificial neural network is modeled using layers of artificial neurons, or computational units able to receive input and apply an activation function along with a threshold to determine if messages are passed along. In a simple model, the first layer is the input layer, followed by one hidden layer, and lastly by an output layer. Each layer can contain one or more neurons. Models can become increasingly complex, and with increased abstraction and problem-solving capabilities by increasing the number of hidden layers, the number of neurons in any given layer, and/or the number of paths between neurons. Note that an increased chance of overfitting can also occur with increased model complexity.

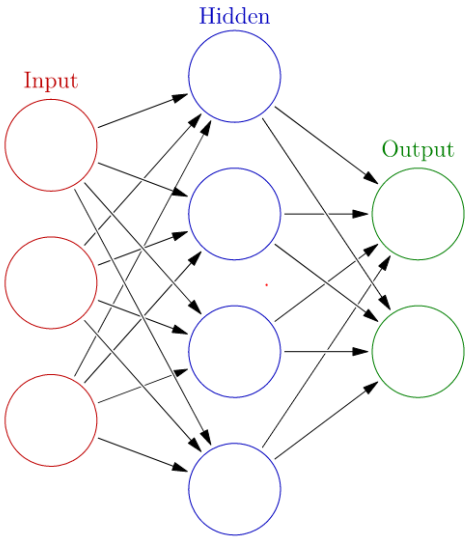


Fig.17: Artificial Neural Networks

Model architecture and tuning are therefore major components of ANN techniques, in addition to the actual learning algorithms themselves. All of these characteristics of an ANN can have significant impact on the performance of the model. Additionally, models are characterized and tunable by the activation function used to convert a neuron's weighted

input to its output activation. The abstraction of the output as a result of the transformations of input data through neurons and layers is a form of distributed representation, as contrasted with local representation. The meaning represented by a single artificial neuron for example is a form of local representation. The meaning of the entire network however, is a form of distributed representation due to the many transformations across neurons and layers. One thing worth noting is that while ANNs are extremely powerful, they can also be very complex and are considered black box algorithms, which means that their inner-workings are very difficult to understand and explain. Choosing whether to employ ANNs to solve problems should therefore be chosen with that in mind.

4.6.2 Convolutional Neural Networks:

Convolutional Neural Networks are very similar to ordinary Neural Networks from the previous chapter: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. Regular Neural Nets don’t scale well to full images. In CIFAR-10, images are only of size 32x32x3 (32 wide, 32 high, 3 color channels), so a single fully-connected neuron in a first hidden layer of a regular Neural Network would have 32*32*3 = 3072 weights. This amount still seems manageable, but clearly this fully-connected structure does not scale to larger images. For example, an image of more respectable size, e.g. 200x200x3, would lead to neurons that have 200*200*3 = 120,000 weights. Moreover, we would almost certainly want to have several such neurons, so the parameters would add up quickly! Clearly, this full connectivity is wasteful, and the huge number of parameters would

quickly lead to overfitting. 3D volumes of neurons. Convolutional Neural Networks take advantage of the fact that the input consists of images and they constrain the architecture in a more sensible way. In particular, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions: width, height, depth. (Note that the word depth here refers to the third dimension of an activation volume, not to the depth of a full Neural Network, which can refer to the total number of layers in a network.) For example, the input images in CIFAR-10 are an input volume of activations, and the volume has dimensions 32x32x3 (width, height, depth respectively). As we will soon see, the neurons in a layer will only be connected to a small region of the layer before it, instead of all of the neurons in a fully-connected manner. Moreover, the final output layer would for CIFAR-10 have dimensions 1x1x10, because by the end of the ConvNet architecture we will reduce the full image into a single vector of class scores, arranged along the depth dimension. Here is a visualization:

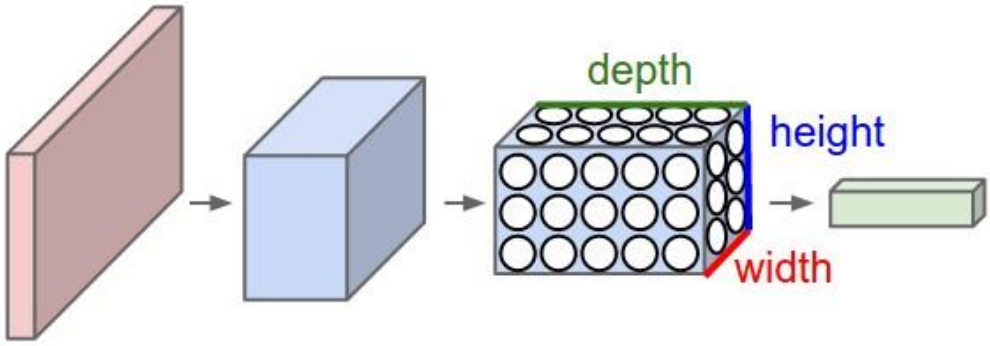


Fig.18: Convolutional Neural Networks

A simple ConvNet is a sequence of layers, and every layer of a ConvNet transforms one volume of activations to another through a differentiable function. We use three main types of layers to build ConvNet architectures: Convolutional Layer, Pooling Layer, and Fully-Connected

Layer (exactly as seen in regular Neural Networks). We will stack these layers to form a full ConvNet architecture.

Example Architecture: Overview. We will go into more details below, but a simple ConvNet for CIFAR-10 classification could have the architecture [INPUT - CONV - RELU - POOL - FC]. In more detail:

- INPUT [32x32x3] will hold the raw pixel values of the image, in this case an image of width 32, height 32, and with three color channels R,G,B.
- CONV layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. This may result in volume such as [32x32x12] if we decided to use 12 filters.
- RELU layer will apply an elementwise activation function, such as the max(0,x)max(0,x) thresholding at zero. This leaves the size of the volume unchanged ([32x32x12]).
- POOL layer will perform a down sampling operation along the spatial dimensions (width, height), resulting in volume such as [16x16x12].
- FC (i.e. fully-connected) layer will compute the class scores, resulting in volume of size [1x1x10], where each of the 10 numbers correspond to a class score, such as among the 10 categories of CIFAR-10. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

In this way, ConvNets transform the original image layer by layer from the original pixel values to the final class scores. Note that some layers contain parameters and other don’t. In particular, the CONV/FC layers perform transformations that are a function of not only the activations in the input volume, but also of the parameters (the weights and biases of the neurons). On the other hand, the RELU/POOL layers

will implement a fixed function. The parameters in the CONV/FC layers will be trained with gradient descent so that the class scores that the ConvNet computes are consistent with the labels in the training set for each image.

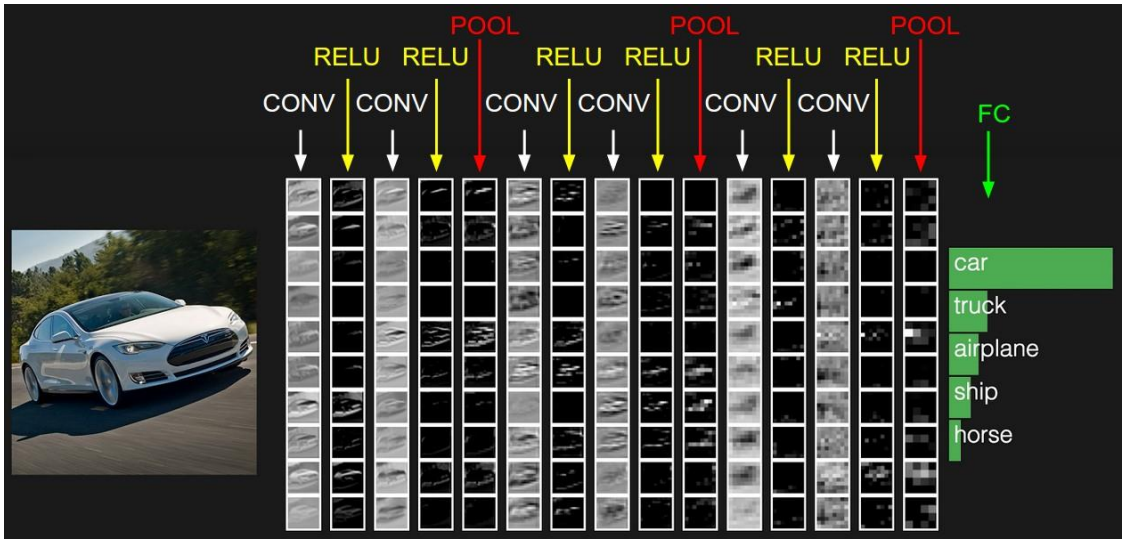


Fig.19: CNN Layers

Advantages and disadvantages of CNN networks:

1) High accuracy in image recognition problems compared to other neural networks.
2) CNN networks often have High computational cost.
3) CNN networks are relatively need high end GPU.
4) CNN networks usually need a lot of training data.

CHAPTER 5

TESTING

5.1 TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

5.2 DEVELOPING METHODOLOGIES

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. We are using two types of testing, Unit testing and Integration testing.

5.3 UNIT TESTING:

Unit testing involves the design of the test cases that validates that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic test at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a

business process performs accurately to the documented specifications and contains clearly defined inputs and expected results. Since we are building multiple modules in python, each module is tested separately.

Advantages of Unit testing:

- One of the main benefits of unit testing is that it makes the coding process more Agile.
- It identifies every defect that may have come up before code is sent further for integration testing.
- Unit testing allows the programmer to refactor code or upgrade system libraries at a later date and make sure the module still works correctly.

5.4 INTEGRATION TESTING:

Software integration is the incremental integration testing of two or more integrating software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

User acceptance testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Advantages of Integration testing:

- Integration testing need not wait until all the modules of a system are coded and unit tested. Instead, it can begin as soon as the relevant modules are available.
- Integration testing or incremental testing is necessary to verify whether the software modules work in unity

CHAPTER 6

CONCLUSION AND FUTURE ELABORATION

6.1 CONCLUSION

Deep learning is a new breed of Machine Intelligence technique, which is gaining much popularity and wide use in various computer science fields, such as object recognition, speech recognition, signal processing, robotics, AI gaming, and so forth. The semi-automated designed system uses low power consumption and only used in crises situations. The proposed system is mainly useful in reducing the highway accidents. In future it is very useful for disabled person, senior citizens and reduces the number of road accidents. Since semi automation system is used, manufacturing cost will be reduced, and it can be used for commercial purpose.

Advantages of proposed system:

a. Drunk and driving incidents should decrease, because there's no designated driver needed when the car drive itself.
b. Automatic driving, like autonomous braking, self-parking, sensors that clue a driver in to a nearby obstacles.
c. Over time, higher speed limits might be considered as an option if more people are using self-driving cars. The computer calculates operation of the vehicle safely via speed and time.

6.2 FUTURE EXTENSION

In future it is very useful for disabled person, senior citizens and also reduces the number of road accidents. Since we are using semi automation system, manufacturing cost will be reduced, and it can be used for commercial purpose.

APPENDIX-I

SAMPLE CODE

Arduino code:

```
int right_pin = 6;
int left_pin = 7;
int forward_pin = 10;
int reverse_pin = 9;

String readString;

void setup() {
  Serial.begin(115200);
  pinMode(right_pin, OUTPUT);
  pinMode(left_pin, OUTPUT);
  pinMode(forward_pin, OUTPUT);
  pinMode(reverse_pin, OUTPUT);// Sets the echoPin as an Input
}


void loop() {
  while(Serial.available()){
    delay(50);
    char c=Serial.read();
    readString+=c;
  }
  if(readString.length()>0){
    Serial.println(readString);

    if (readString =="FORWARD"){
      digitalWrite(forward_pin, LOW);
      delay(500);
      Serial.println("forwding");
    }
    if (readString =="BACKWARD"){
      Serial.println("forwding");
      digitalWrite(reverse_pin, LOW);
      delay(500);
      Serial.println("backward");
    }
    if (readString =="LEFT"){
      Serial.println("forwding");
      digitalWrite(left_pin, LOW);
```

```
    delay(500);
    Serial.println("left");
   }
   if (readString =="RIGHT"){
    Serial.println("forwding");
    digitalWrite(right_pin, LOW);
    delay(500);
    Serial.println("right ");
   }
   if (readString =="STOP"){
    Serial.println("forwding");
    digitalWrite(right_pin, HIGH);
    digitalWrite(left_pin, HIGH);
    digitalWrite(forward_pin, HIGH);
    digitalWrite(reverse_pin, HIGH);
    delay(500);
    Serial.println("releasing");
   }

   readString="";
  }
}
```

Autonomous drive code :

```
#parsing command line arguments
import argparse
#decoding camera images
import base64
#for frametimestamp saving
from datetime import datetime
#reading and writing files
import os
#high level file operations
import shutil
#matrix math
import numpy as np
#real-time server
import socketio
#concurrent networking
import eventlet
#web server gateway interface
import eventlet.wsgi
#image manipulation
from PIL import Image
#web framework
from flask import Flask
#input output
from io import BytesIO
```

```
#load our saved model
from keras.models import load_model

#helper class
import utils

#initialize our server
sio = socketio.Server()
#our flask (web) app
app = Flask(__name__)
#init our model and image array as empty
model = None
prev_image_array = None

#set min/max speed for our autonomous car
MAX_SPEED = 25
MIN_SPEED = 10

#and a speed limit
speed_limit = MAX_SPEED

#registering event handler for the server
@sio.on('telemetry')
def telemetry(sid, data):
    if data:
        # The current steering angle of the car
        steering_angle = float(data["steering_angle"])
        # The current throttle of the car, how hard to push peddle
        throttle = float(data["throttle"])
        # The current speed of the car
        speed = float(data["speed"])
        # The current image from the center camera of the car
        image = Image.open(BytesIO(base64.b64decode(data["image"])))
        try:
            image = np.asarray(image)       # from PIL image to numpy array
            image = utils.preprocess(image) # apply the preprocessing
            image = np.array([image])       # the model expects 4D array

            # predict the steering angle for the image
            steering_angle = float(model.predict(image, batch_size=1))
            # lower the throttle as the speed increases
            # if the speed is above the current speed limit, we are on a downhill.
            # make sure we slow down first and then go back to the original max speed.
            global speed_limit
            if speed > speed_limit:
                speed_limit = MIN_SPEED  # slow down
            else:
                speed_limit = MAX_SPEED
            throttle = 1.0 - steering_angle**2 - (speed/speed_limit)**2
            ############################################################
```

```
            print('{} {} {}'.format(steering_angle, throttle, speed))
            send_control(steering_angle, throttle)
            if steering_angle < -0.15 and steering_angle < 0  :
                print('Left')
            elif steering_angle > 0.15 :
                print('Right')
            #elif steering_angle < -0.36 and steering_angle < 0 :
                #print('Left')
        except Exception as e:
            print(e)

        # save frame
        if args.image_folder != '':
            timestamp = datetime.utcnow().strftime('%Y_%m_%d_%H_%M_%S_%f')[:-3]
            image_filename = os.path.join(args.image_folder, timestamp)
            image.save('{}.jpg'.format(image_filename))
    else:

        sio.emit('manual', data={}, skip_sid=True)


@sio.on('connect')
def connect(sid, environ):
    print("connect ", sid)
    send_control(0, 0)


def send_control(steering_angle, throttle):
    sio.emit(
        "steer",
        data={
            'steering_angle': steering_angle.__str__(),
            'throttle': throttle.__str__()
        },
        skip_sid=True)


if __name__ == '__main__':
    parser = argparse.ArgumentParser(description='Remote Driving')
    parser.add_argument(
        'model',
        type=str,
        help='Path to model h5 file. Model should be on the same path.'
    )
    parser.add_argument(
        'image_folder',
        type=str,
        nargs='?',
        default='',
        help='Path to image folder. This is where the images from the run will be saved.'
```

```
    )
    args = parser.parse_args()

    #load model
    model = load_model(args.model)

    if args.image_folder != '':
        print("Creating image folder at {}".format(args.image_folder))
        if not os.path.exists(args.image_folder):
            os.makedirs(args.image_folder)
        else:
            shutil.rmtree(args.image_folder)
            os.makedirs(args.image_folder)
        print("RECORDING THIS RUN ...")
    else:
        print("NOT RECORDING THIS RUN ...")

    # wrap Flask application with engineio's middleware
    app = socketio.Middleware(sio, app)

    # deploy as an eventlet WSGI server
    eventlet.wsgi.server(eventlet.listen(('', 4567)), app)
```

Object Detection code:

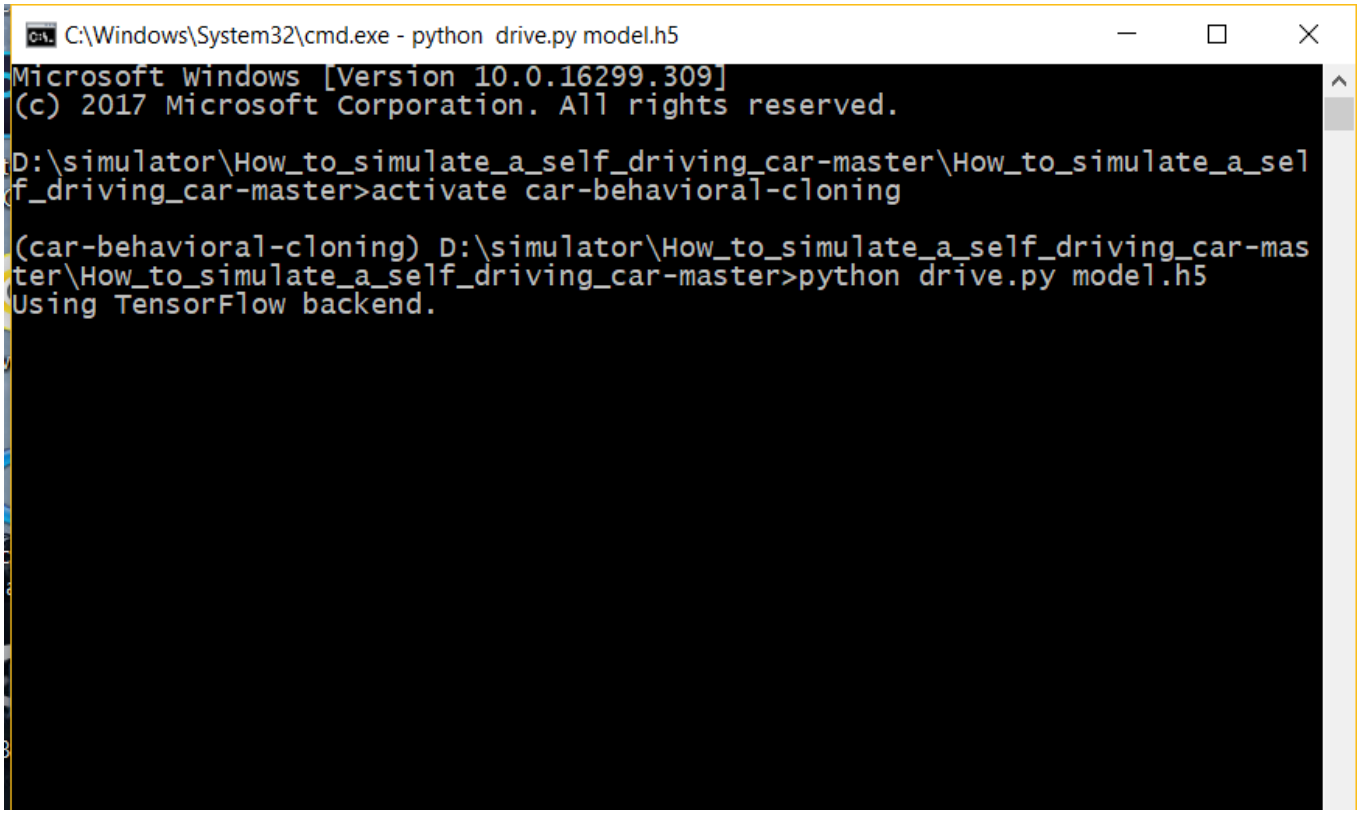
```
import numpy as np
import cv2
import serial
import pygame
from pygame.locals import *
from pygame import *


import urllib.request


url = 'http://192.168.0.100:8080/shot.jpg'
cascade = cv2.CascadeClassifier('traffic_light.xml')


class RCTest(object):

    def __init__(self):


        pygame.init()
        screen = display.set_mode((640,480))
        display.set_caption('the input..')
        self.ser = serial.Serial('COM35', 115200, timeout=1)
        self.send_inst = True
```

```
        self.steer()

    def stop(self):
        print("stop")
        self.ser.write(chr(0).encode())


    def forward(self):
        print("Safe go forward...")
        self.ser.write(chr(1).encode())


    def steer(self):

        while self.send_inst:
            imgResp=urllib.request.urlopen(url)
            imgNp=np.array(bytearray(imgResp.read()),dtype=np.uint8)
            img=cv2.imdecode(imgNp,-1)
            cv2.imshow('test',img)

            gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
            frame = cascade.detectMultiScale(gray, scaleFactor=1.1,
                    minNeighbors=5,
                    minSize=(30, 30),
                    flags=cv2.CASCADE_SCALE_IMAGE )
            if not any(map(len, frame)):
                print("leist emyg")
                self.forward()


            for (x,y,w,h) in frame:
                cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,255),2)
                roi_gray = gray[y:y+h, x:x+w]
                roi_color = img[y:y+h, x:x+w]
                self.stop()
                #self.ser.write(chr(0).encode())


            cv2.imshow('test',img)
            if ord('q')==cv2.waitKey(10):
                exit(0)

if __name__ == '__main__':
    RCTest()
```

APPENDIX-II

SCREEN SHOTS

S1-CMD commands:



Screen shoot description:

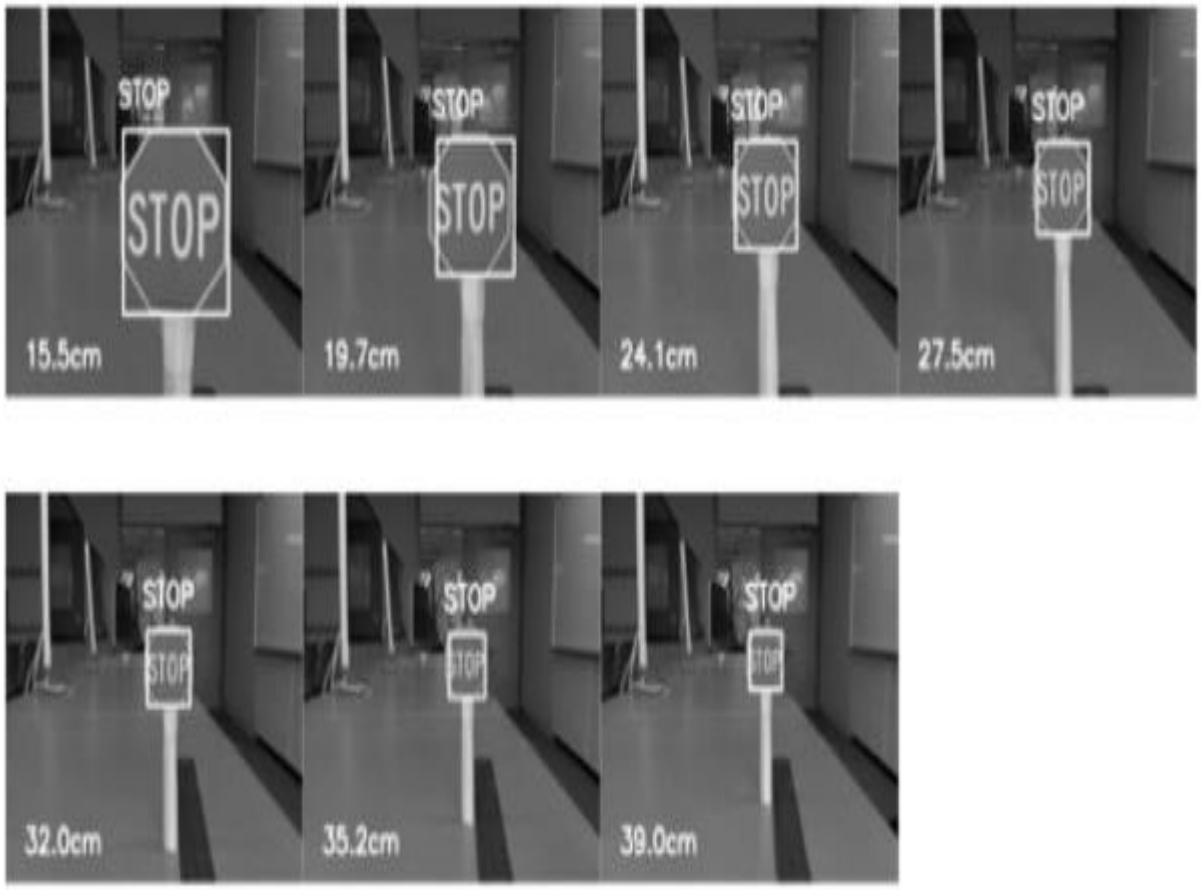
A conda environment is a directory that contains a specific collection of conda packages that you have installed. Using that we can easily transfer one python program from one environment to another.

S2-RC Objection Setup:

Screen shoot description:

Rc car remote is connect via a ardunio. Then the ardunio is connect to Laptop. Using python coding we can easily control the car from the laptop. And we also attached a camera to the car so that it can observe the surrounding environment.

S3- TESTING FOR REAL TIME OBJECT DETECTION:



Screen shoot description:

Testing the object detection, and distance between the camera and the object. Which is then send to laptop then navigation function is activated.

S4 - TESTING OF AUTONOMUS MODE:

Screen shoot description:

Rc car is tested in a virtual environment, first the car is trained for few laps. And the autonomous mode is tested.

S5 - TESTING OF REALTIME COLOR DETECTION:

Screen shoot description:

The Navigator is tested in real life environment and performance is noted. Using that data Navigator performance is increased.

APPENDIX-III

LIST OF PUBLICATION

International Journal Publication:

- ✓ Gokul S , Murugan V and Dr.R Indra Gandhi, "Artificial Neural Network Based Automated Escalating Tool for Crisis Navigation ", International Conference on Modern Global Research in Engineering and Technology, 10th March 2018

Conferences:

- ✓ Gokul S , Murugan V and Dr.R Indra Gandhi," Artificial Neural Network Based Automated Escalating Tool for Crisis Navigation ", International Journal of Trend in Scientific Research and Development (IJTSRD), Volume -2 (issue -3), Mar – Apr 2018

International Journal of Trend in Scientific Research and Development (IJTSRD)
International Open Access Journal
ISSN No: 2456 - 6470 | www.ijtsrd.com | Volume - 2 | Issue – 3

Artificial Neural Network Based Automated Escalating Tools for Crises Navigation

Murugan Venkatesan, S. Gokul, Dr. R. Indra Gandhi
Department of Computer Science and Engineering, G.K.M. College of Engineering and Technology
Perungalathur, Chennai, Tamil Nadu, India

ABSTRACT

Autonomous driving technology has made significant advances in recent years. In order to make self-driving cars more practical, they are required to operate safely and reliably even under adverse driving condition. The object detection based on deep learning is an important application in deep learning technology, which is characterized by its strong capability of features learning and feature representation compared with the traditional object detection method. After analyzing the characteristics of videos shot by the camera, we choose to use deep learning to train a vehicle detection model to detect targets in video. In the end, we use trained data set to control the speed and navigate the vehicle in crises situations. Conversely, not much research is going on of usage such networks for elaborating of real time data. The goal of this work is exploring, experimenting and providing new approaches of classification non-stationery data using neural network.

Keywords: *Autonomous, Object Detection, Deep Learning, Vehicle Detection, crises, Neural Network*

I. INTRODUCTION

Autonomous Driving has been said to be the next big disruptive innovation in the years to come. Considered as being predominantly technology driven, it is supposed to have massive societal impact in all kinds of fields. Having a closer look at the history of Autonomous Driving, as explained in the IEEE Spectrum it can be observed that the technological development and main milestones of the autonomous driving field started already a few decades ago. Leading to a vast analysis of some semi-autonomous features, development of present technologies and understanding on the future problematic while focusing the near future in the connected car. Full automated car is not accepted by many countries and doesn't show high accuracy in real life. It doesn't mean humans are better drivers, human drivers face many situations where they lose their human ability e.g. drunk and drive, drowsiness etc. For improving the safety and a new method has been proposed. The method includes the combination of deep-learning object detection and human activity monitoring to make self -driving more reliable.

II. RELATED WORKS

The application of reinforcement learning on control and decision-making has been investigated in several works. Pyeatt and Howe [10] applied reinforcement learning to learning racing behaviours in Robot Auto Racing Simulator, precursor of the TORCS platform, both are open source racing simulators. Daniele et al. [11] used the tabular Q learning model to learn the overtaking strategies on TORCS. Riedmiller [3] proposed a neural reinforcement learning method, namely neural fitted Q-iteration (NFQ), to generate control strategy for the pole balancing and mountain car task with least interactions. In above work, both the state and action spaces are low dimensional. NFQ used the MultiLayer Perceptron (MLP) model as the value function for Q iteration in which the traditional three-layer neural network is employed under normal circumstances. It may fail to find a feasible solution with the increased status and action spaces. In recent years, the deep reinforcement learning has seen an

exciting development. One of the representative works is published in Nature 2015 by the researchers of Google DeepMind, in which the authors developed a novel artificial agent of a deep Q-network, based on convolutional neural network (CNN) and Q-learning. The artificial agent learned policies directly from high-dimensional sensor inputs and achieved human-level control on the challenging domain of classic Atari 2600 games [8]. Another exciting work by Google DeepMind is more convincing. AlphaGo defeated the world chess champion Lee Se-dol by 4:1 [12]. The technology behind it was a combination reinforcement learning, and Monte Carlo tree search technique, supplemented with extensive training set.

III. Proposed System

The proposed system developed to drive cars in an efficient and safer way. This system does not support for fully automated self-drive car. Instead, we are likely to present the semi-automatic car for some CRISES situation. The semi-automatic car can be easily interacted with human beings in emergency. [(Ex) In Drunk and Driving situation, car would have 85% of the control, the driver can still drive the car but within a limited speed. During turning or Heavy Traffic, the car will detect the traffic signal using pre-trained model and make decisions according to the situation]. The four phases involve mainly a sensor phase, an image processing phase, an object detection and an automated phase.

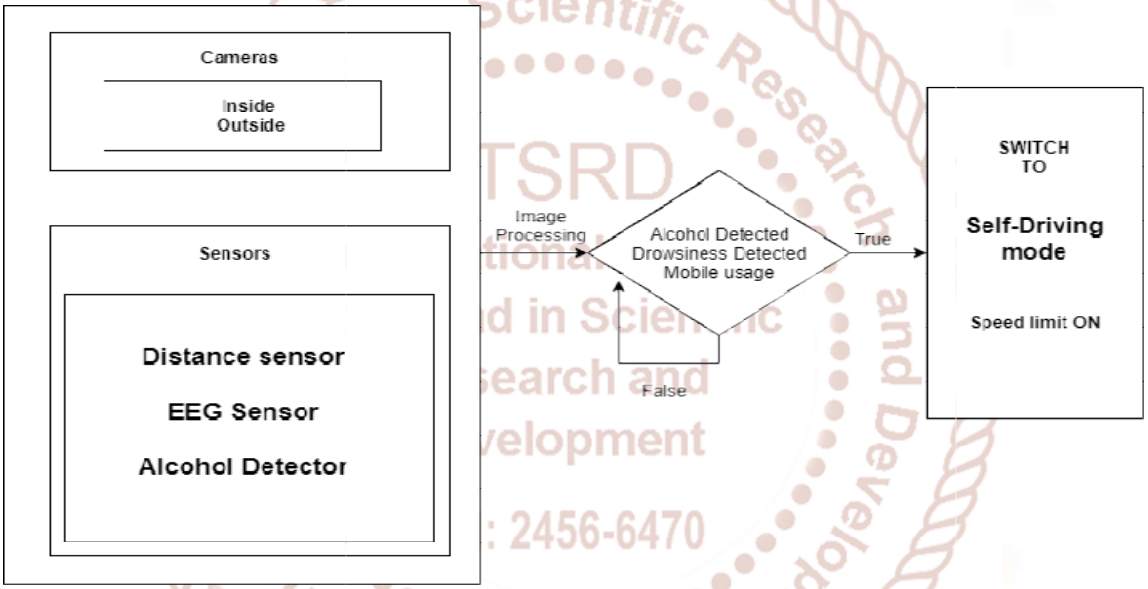


Fig. 1 Architecture of semi-automated system

A. Sensor phase

Multiple sensors and cameras are attached with the car. This provides input data to the system. Data are divided in two types of data, Camera data and sensor data.

i. Camera data:

Data from camera is used for image processing. Data from the camera is high in resolution, so it is reduced to low resolution. Four cameras are used, three cameras are inserted outside the cars which used control the steering angle, and another camera is used to collect images inside the car to detect drowsiness and mobile usage.

ii. Sensor data:

Three types of sensors are used Distance sensor used to calculate distance, EEG Sensors is used to detect drowsiness and Alcohol detector (MQ-3) used detect the level of alcohol in the driver's breath.

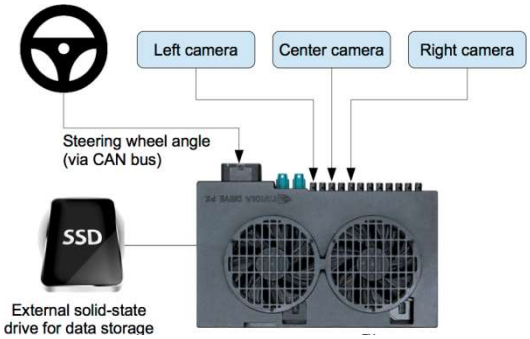


Fig. 2 Camera data

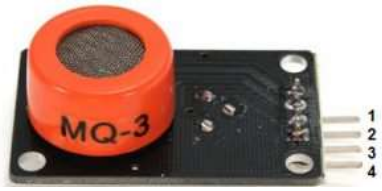


Fig. 3 MQ-3 Alcohol detector

B. *Image Processing Phase*

For image processing neural networks is used. One advantage of using neural network is that once the network is trained, it only needs to load trained parameters afterwards, thus prediction can be very fast. An end-to-end learning approach is used for self-driving. The end-to-end learning takes the raw image as input and outputs the control signal automatically. The model is self-optimized based on the training data and there is no manually defined rules. These become the two major advantages of end-to-end learning: better performance and less manual effort. Because the model is self-optimized based on the data to give maximum overall performance, the intermediate parameters are self-adjusted to be optimal.

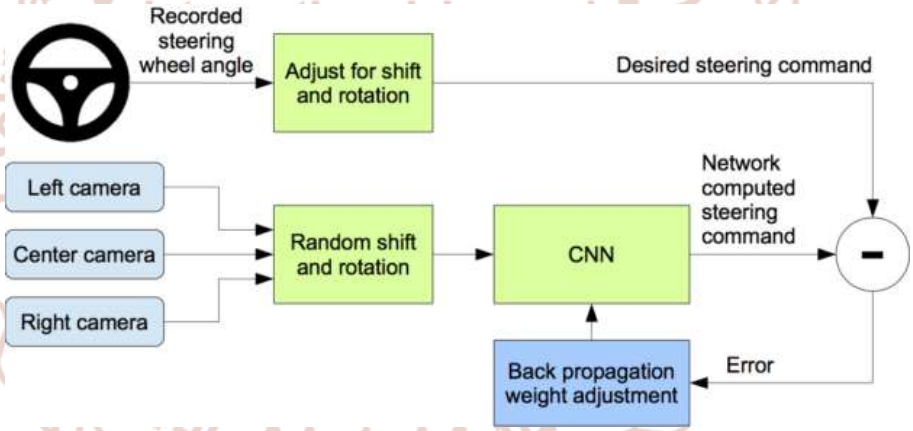


Fig. 4 End-to-End Deep Learning

C. Object Detection Phase

For object detection TensorFlow object detection API is used. In order to use this API, a Images are collected in TFRecords for the training and testing data, two options are available. Pre-trained model is used, and then transfer learning to learn a new object, or learn new objects entirely from scratch. The benefit of transfer learning is that training can be much quicker, and the required data that you might need is much less. For this reason, we're going to be doing transfer learning here. Pre trained ssd_mobilenet_v1_coco model was used to train the Traffic signals dataset, which produced high accuracy and faster detection.

Fig. 5 Traffic sign detection

D. Automated Phase

In this phase data from sensors used for controlling the car i.e., each sensor is set to a threshold value, a driver should satisfy this in order to control the car. Distance sensor is used to calculate the distance between the object in front of the car, for an example minimum distance can be set to 20 meters which helps the car to avoid collision. EEG sensor is used to detect drowsiness and the alcohol detector detects the alcohol level of the driver if the level is high, the car is switched to automatic mode and speed of car is reduced. Since the human driver is not is his control so he/she lost is human ability, so the automatic car can perform better than the human driver.

E. Conclusions

Deep learning is a new breed of Machine Intelligence technique, which is gaining much popularity and wide use in various computer science fields, such as object recognition, speech recognition, signal processing, robotics, AI gaming, and so forth. The semi-automated designed system uses low power consumption and only used in crises situations. The proposed system is mainly useful in reducing the highway accidents. In future it is very useful for disabled person, senior citizens and reduces the number of road accidents. Since semi automation system is used, manufacturing cost will be reduced, and it can be used for commercial purpose.

REFERENCES

1) Deep Neural Network for Structural Prediction and Lane Detection in Traffic Scene Jun Li, Xue Mei, Senior Member, IEEE, Danil Prokhorov, Senior Member, IEEE, and Dacheng Tao, Fellow, IEEE

2) Lee, Unghui, et al, "Local path planning in a complex environment for self-driving car," Cyber Technology in Automation, Control, and Intelligent Systems, 2014 IEEE 4th Annual International Conference on. IEEE, 2014, pp. 445-450.

3) Shim Inwook, et al, "An Autonomous Driving System for Unknown Environments using a Unified Map," IEEE Transactions on Intelligent Transportation Systems, December 2014, to be published.

4) Shin, Seunghak, Inwook Shim, and In So Kweon. "Combinatorial approach for lane detection using image and LIDAR reflectance." Ubiquitous Robots and Ambient Intelligence (URAI), 2015 12th International Conference on. IEEE, 2015, pp. 485-487

5) A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 427-436.

6) Y. Qian, Y. Fan, W. Hu, and F. K. Soong, "On the training aspects of Deep Neural Network (DNN) for parametric TTS synthesis," in 2014 IEEE

International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 3829-3833

7) E. Santana and G. Hotz, "Learning a driving simulator," CoRR, vol. abs/1608.01230, 2016. [Online]. Available: http://arxiv.org/abs/1608.01230

8) Ma B., Lakshmanan S., Hero A.O. Simultaneous Detection of Lane and Pavement Boundaries Using Model-Based Multisensor Fusion // IEEE Transcations on Intelligent Transportation Systems, 2000. Vol. 1, no. 3. pp. 135–147.

9) Lane Recognition Using On-vehicle LIDAR / T. Ogawa, K. Takagi // Intelligent Vehicles Symposium, 2006. pp. 540–545

10) P. WawrzyĔSki and A. K. Tanwani, "Autonomous reinforcement learning with experience replay," Neural Networks, vol. 41, pp. 156167, 2013.

International Journal of Trend in Scientific Research and Development

Online Peer Reviewed Open Access Journal ISSN: 2456-6470

Certificate of Publication

This Certificate is Proudly Presented for Honorable Achievement to

Murugan Venkatesan

Department of Computer Science and Engineering,
G.K.M. College of Engineering and Technology, Perungalathur, Chennai, Tamil Nadu, India

For Research Paper with title **"Artificial Neural Network Based Automated Escalating Tools for Crises Navigation"**

Published in International Journal of Trend in Scientific Research and Development (IJTSRD), Vol - 2 | Issue - 3 (Mar-Apr-2018). With Impact Factor 4.101 for the year of 2018 by I2OR

IJTSRD10900
Unique Paper ID

www.ijtsrd.com
Available Online

M M Patel

Dr. M. M. Patel
Chief Editor

International Journal of Trend in Scientific Research and Development

Online Peer Reviewed Open Access Journal ISSN: 2456-6470

Certificate of Publication

This Certificate is Proudly Presented for Honorable Achievement to

S. Gokul

Department of Computer Science and Engineering,
G.K.M. College of Engineering and Technology, Perungalathur, Chennai, Tamil Nadu, India

For Research Paper with title **"Artificial Neural Network Based Automated Escalating Tools for Crises Navigation"**

Published in International Journal of Trend in Scientific Research and Development (IJTSRD), Vol - 2 | Issue - 3 (Mar-Apr-2018). With Impact Factor 4.101 for the year of 2018 by I2OR

IJTSRD10900
Unique Paper ID

www.ijtsrd.com
Available Online

M M Patel

Dr. M. M. Patel
Chief Editor

SRI RAMAKRISHNA COLLEGE OF ENGINEERING
love knowledge strength
SRI RAMAKRISHNA
GROUP OF INSTITUTIONS

ECHOWN

CERTIFICATE

International Conference on Modern Global Research in Engineering and Technology

Perambalur 10th March 2018

This is to certify that S.Gokul of GKM College of Engineering and Technology has done his/her excellence is presenting research paper titled

" Artificial Neural Network Based Automated Escalating Tools For Crises Navigation "

at Perambalur 10th March 2018

SECRETARY

SRI RAMAKRISHNA COLLEGE OF ENGINEERING * PERAMBALUR * SIRUVACHUR *

PROGRAM CHAIR

SRI RAMAKRISHNA COLLEGE OF ENGINEERING
love knowledge strength
SRI RAMAKRISHNA
GROUP OF INSTITUTIONS

ECHOWN

CERTIFICATE

International Conference on Modern Global Research in Engineering and Technology

Perambalur 10th March 2018

This is to certify that V.Murugan of GKM College of Engineering and Technology has done his/her excellence is presenting research paper titled

" Artificial Neural Network Based Automated Escalating Tools For Crises Navigation "

at Perambalur 10th March 2018

SECRETARY

SRI RAMAKRISHNA COLLEGE OF ENGINEERING * SRI RAMAKRISHNA COLLEGE * PERAMBALUR

PROGRAM CHAIR

APPENDIX-IV

REFERENCES

1) Deep Neural Network for Structural Prediction and Lane Detection in Traffic Scene Jun Li, Xue Mei, Senior Member, IEEE, Danil Prokhorov, Senior Member, IEEE, and Dacheng Tao, Fellow, IEEE
2) Lee, Unghui, et al, "Local path planning in a complex environment for self-driving car," Cyber Technology in Automation, Control, and Intelligent Systems, 2014 IEEE 4th Annual International Conference on. IEEE, 2014, pp. 445-450.
3) Shim Inwook, et al, “An Autonomous Driving System for Unknown Environments using a Unified Map,” IEEE Transactions on Intelligent Transportation Systems, December 2014, to be published.
4) Shin, Seunghak, Inwook Shim, and In So Kweon. "Combinatorial approach for lane detection using image and LIDAR reflectance." Ubiquitous Robots and Ambient Intelligence (URAI), 2015 12th International Conference on. IEEE, 2015, pp. 485-487
5) A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 427-436.
6) Y. Qian, Y. Fan, W. Hu, and F. K. Soong, "On the training aspects of Deep Neural Network (DNN) for parametric TTS synthesis," in 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, pp. 3829-3833

7) E. Santana and G. Hotz, “Learning a driving simulator,” CoRR, vol. abs/1608.01230,2016.[Online].Available: http://arxiv.org/abs/1608.01230
8) Ma B., Lakshmanan S., Hero A.O. Simultaneous Detection of Lane and Pavement Boundaries Using Model-Based Multisensor Fusion // IEEE Transcations on Intelligent Transportation Systems, 2000. Vol. 1, no. 3. pp. 135–147.
9) Lane Recognition Using On-vehicle LIDAR / T. Ogawa, K. Takagi // Intelligent Vehicles Symposium, 2006. pp. 540–545