**SELF-DRIVIN CAR USING RASBERRY PI AND ARDIUNO**

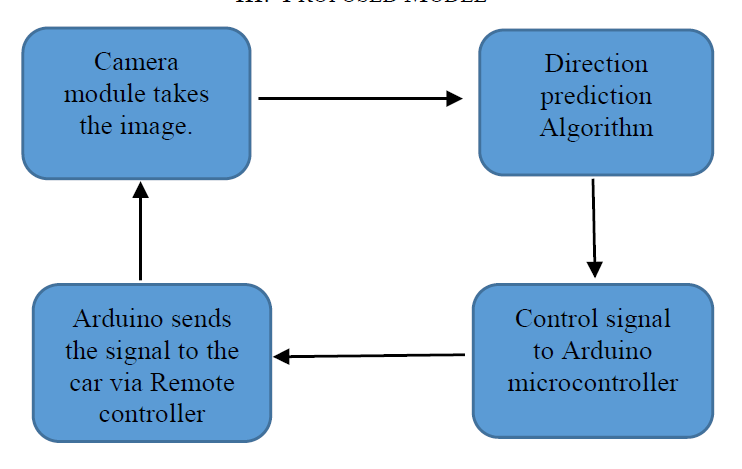
**PROPOSED SYSTEM:**

The evolution of Artificial Intelligence has served as the catalyst in the field of technology. We can now develop things which was once just an imagination. One of such creation is the birth of self-driving car. Days have come where one can do their work or even sleep in the car and without even touching the steering wheel, accelerator you will still be able to reach your target destination safely. This paper proposes a working model of self-driving car which is capable of driving from one location to the other or to say on different types of tracks such as curved tracks, straight tracks and straight followed by curved tracks. A camera module is mounted over the top of the car along with Raspberry Pi sends the images from real world to the Convolutional Neural Network which then predicts one of the following directions. i.e. right, left, forward or stop which is then followed by sending a signal from the Arduino to the controller of the h-bridge of the cheese

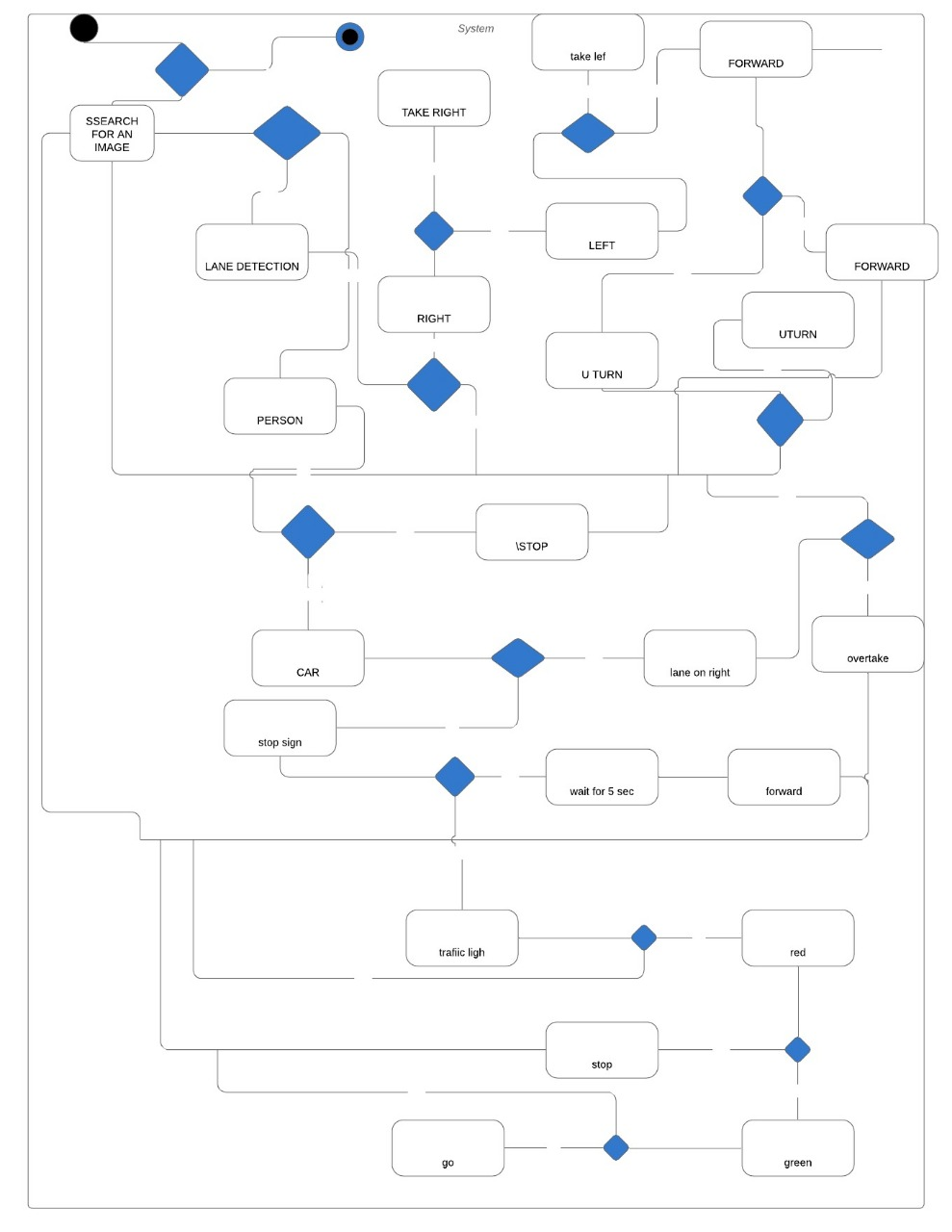
car and as a result of it the car moves in the desired direction without any human intervention.

**FLOW OF MODULES**

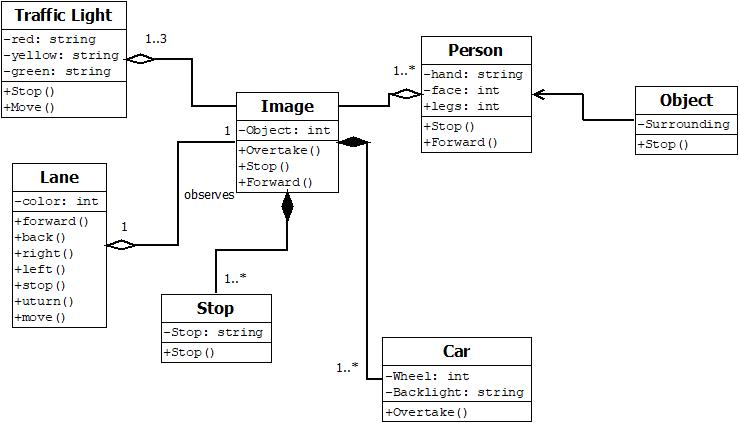
An external device, PICamera will be responsible for image capturing from the environment and passes to the RaspberryPI processor. Now, RaspberryPI has the captured image and will perform some analysis on it by using the image detection algorithm producing some function value and further passes it to the CNN model. In our case, the images are well- defined i.e Stop sign signal, Object detection (Car, human, etc), traffic lights (Red, Green and Yellow) and Lane detection. The corresponding functions are made in the raspberryPI algorithm. So, whenever an image passes through an image detection algorithm, the corresponding function is called and the value of that function is passed by raspberryPI to an Arduino through GPIO pins to RX-TX i.e Pin no.1 to 4. The value that’s inputted in the Arduino, the corresponding value’s decimal form is taken by calling a function. Based on that function, the Arduino gives command to a L298HBridge through 8bit PWM (Pulse With Modulation) Pins. Then, the bridge will take action on the motor.

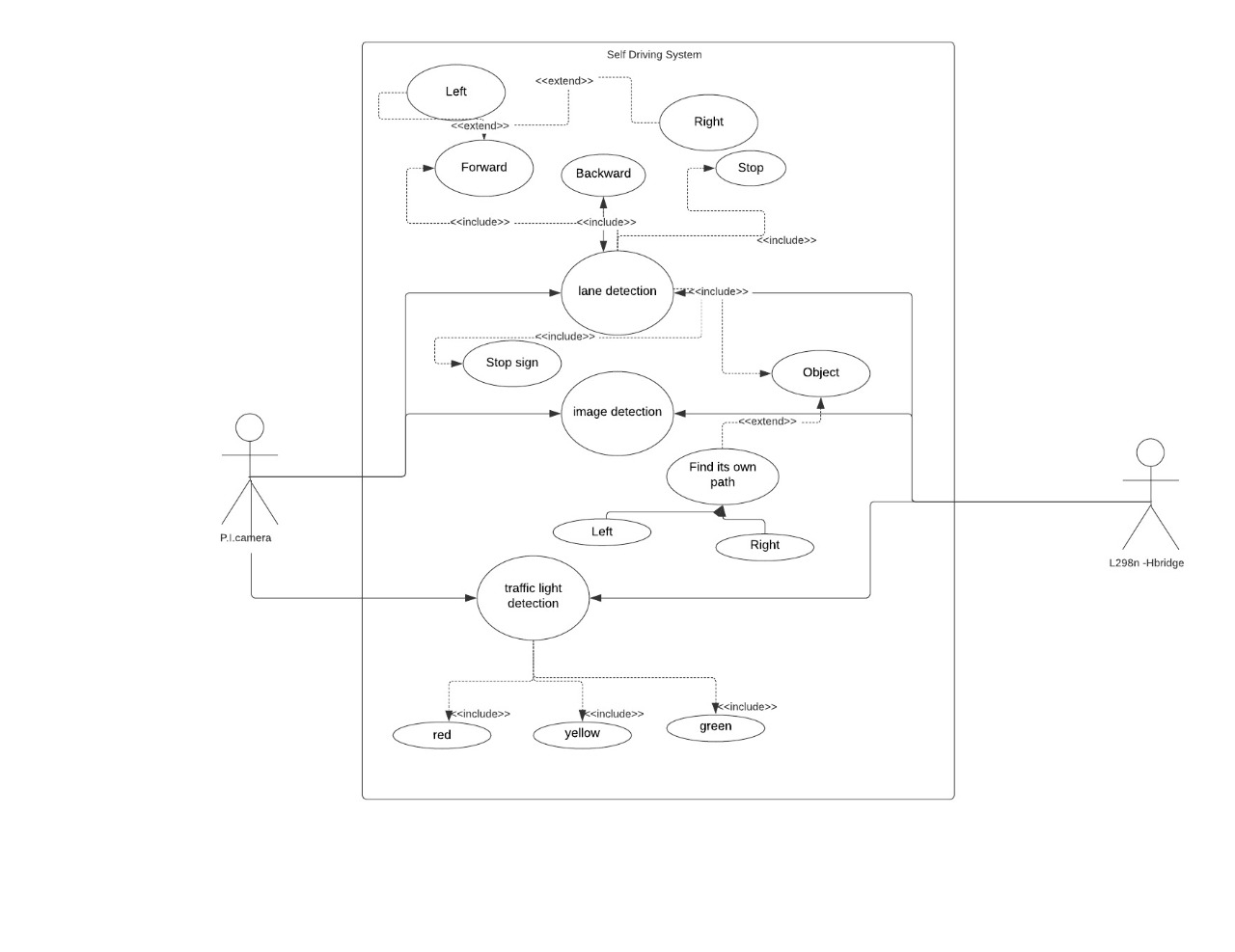


**ACTIVITY DIAGRAM:**

****

**CLASS DIAGRAM:**

****

****

**USE CASE DIAGRAM**

**DESCRIPTION OF USE CASE DIAGRAM:**

The above Use Case Diagram The main role is Pi Camera is used to detect which of the operation should be performed when any obstacle ,Lane direction, potholes or traffic signals occurred.

1. First Pi Camera take the front image and divide the image into pixels, then each pixels will be justify in CNN. After than decision will made.

2. If it is Traffic Signals, in that case it’s about Colour .Then there will be following rules:

i. If it is Red Colour, then the Car will be Stop.

ii. If it is Green Colour, then the Car will Start moving Forward direction.

iii .If it is Yellow Colour, then the Speed of Car will be decrease.

3.If the Image detected by Pi camera said it is Lane Direction picture, then there will be two ways to go:

i. If it is Right sign, Then the car will be moved in Right Direction

ii.If it is Left sign, Then the car will be moved in Left Direction

4.If there is no Path and it observed the U-Turn sign, then the cars is used to take U-turn.

Module 1 : PICamera

Pi camera is great gadget to capture time-lapse, slow motion with great video clarity. The dimensions of camera are 25mm to 24mm by 9mm, which connects to Raspberry Pi via a flexible elastic cord which supports serial interface. The camera image sensor has a resolution of five megapixels and has a focused lens. The camera provides a great support for security purpose. Various characteristics of the camera are it supports 5MP sensor, Wide image, capable of 2592x1944 stills, 1080p30 video on Camera module v1

Module 2 :Raspberry pi:

The Raspberry Pi is a small low cost single board computer having a processor speed ranging from 700 MHz to 1.2 GHz for the Pi 3.The on-board memory ranges from 256 MB to 1 GB RAM. The boards supports up to 4 USB ports along with HDMI port. Along from all this it has number of GPIO pins which support protocols like I²C.Moreover it also supports Wi-Fi and Bluetooth facility which makes device very compatible with other devices. It supports Scratch and Python programming languages [10]. It supports many operating systems like Ubuntu MATE, Snappy Ubuntu, Pidora, Linutop and many more out of which Raspbian is specifically designed to support Raspberry Pi’s hardware

Module 3 :Arduino uno:

This microcontroller is based on ATmega329P.There are 14 digital input/output pins available out of which 6 ca n be used a PWM outputs. It also supports 6 analog inputs. It has 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It has 32 kb of flash memory and 2 kb of SRAM and weighs

around 25g [13].Apart from all these features Arduino IDE is very user friendly and uses basic c as its programming language After attaching these hardware on the car and connecting Arduino with the controller. The setup looks something like shown in the figure. below

Module 4: *Arduino IDE*

Ardino IDE is the platform were the programs are written for Arduino board.It has compile button which helps in compiling the code along with the upload tab which helps to upload the code on the board.Programs written on Arduino IDE are often called Sketches and are saved as .ino extension.The editor has numerous other features like verify, save, upload , include library and serial monitor.Apart from this ,the developers have

made easy to use functions, which makes coding easy and fun.Moreover there are number of examples provided for each and every interface which helps the user learn more about functions and hardware as well.

Module 5:Open CV

OpenCV is an open source computer vision library which is capable of handling images/videos from fairly basic tasks to utter complex tasks like facial recognition. It supports C++, C, Python and Java programming languages and supports Windows, Linux, Mac OS, iOS and Android. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform [14]. In this project it is serving a major support, it helps to crop out the section of the video from the Raspberry-Pi cam interface as shown above and converts it to the grayscale, resize it and then passes it to the Convolutional Neural Network. Spyder is a powerful interactive development environment for the Python language which has advanced editing, interactive testing, debugging and introspection features and a numerical computing environment. It has a matplotlib as plotting library which helps to plot 2D/3D graphs

Module 6: *Raspberry Pi Cam Interface*

To remotely capture the live feed from the camera to the laptop we need to develop an interface which would serve this purpose.This is were the software Rpi-cam interface comes into the picture. It’s the program which helps you capture the live feed by just letting the ip address of the Raspberry pi.One can record and download video/image in various resolutions with different number of settings. Below is the view of the software under action.

Module 8: CNN

**Python:**

It is the programming language used for Machine Learning or Artificial Intelligence tasks.

**OpenCV:**

It is a powerful computer vision package. It can be trained to detect objects in images (or video).

**Tensorflow:**

It i**s** Google’s popular deep learning framework. Tensorflow is used to make smart decisions based on the neural network.

**Google Colab:**

Colab is a free cloud-based Jupyter Notebooks that let you write and train deep learning models in Python. The popular python libraries supported are TensorFlow, Keras, OpenCV, and Pandas

**OpenCV for Computer Vision:**

Perception Sensor of our PiCar is a USB DashCam. A DashCam gives us a live video, which is essentially a sequence of pictures. We will use OpenCV, a powerful open source computer vision library, to capture and transform these pictures

**Numpy** and **Matplotlib** are two very useful python modules that we will use in conjunction with OpenCV for image processing and rendering.

**Tensorflow For CPU:**

Raspberry Pi is not recommended to perform any deep learning (i.e. model training), as its CPU is vastly insufficient for backward propagation, a very slow operation required in the learning process. However, we can use the **Tensorflow CPU** to do model prediction based on a pre-trained model. Model Training which uses only forward propagation, a much faster computer operation.

**TensorFlow for Edge TPU Co-Processor:**

Inferences can only do so on a relatively shallow model (say 20–30 layers) in real time.  But for deeper models (100+ layers), we would need the Edge TPU. A live video screen coming up, and it will try to identify objects in the screen at around 7–8 Frames/sec**.**[**COCO (Common Object in COntext) object detection model**](http://cocodataset.org/#home) can detect about 100 common objects, like a person etc.

The object detection model used in this program is called [**ssd\_mobilenet\_coco\_v2**](https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md#coco-trained-models)

**Autonomous Lane Navigation via OpenCV:**

 Adaptive Cruise Control (ACC) and some forms of Lane Keep Assist System (LKAS). Adaptive cruise control uses radar to detect and keep a safe distance with the car in front of it. Lane Keep Assist System is a relatively new feature, which uses a windshield mount camera to detect lane lines, and steers so that the car is in the middle of the lane. Lane detection concept will turn a video of the road into the coordinates of the detected lane lines. This will be achieved via the computer vision package: OpenCV.

The color space used in the image is  RGB (Red/Green/Blue). The RGB will be converted into HSV (Hue/Saturation/Value) color space.

**Detecting Edges of Lane Lines**:

The [**Canny edge detection function**](https://docs.opencv.org/master/da/d22/tutorial_py_canny.html)is a powerful command that detects edges in an image. This function is available in OpenCV. It converts the detected lane lines into number of white pixels

**Detecting Line Segments:**

The extraction of the coordinates of these lane lines from these white pixels. OpenCV contains a magical function, called Hough Transform, which does exactly this. Hough Transform is a technique used in image processing to extract features like lines, circles, and ellipses. We will use it to find straight lines from a bunch of pixels that seem to form a line. The function HoughLinesP essentially tries to fit many lines through all the white pixels and return the most likely set of lines, subject to certain minimum threshold constraints. Internally, HoughLineP detects lines using Polar Coordinates. Polar Coordinates uses elevation angle( as camera is at elevated height) and distance from the origin (car).

**Steering**: A heading Line is generated using two detected lane lines. A steering angle is calculated using the heading line .

**Image Augmentation:**

Some of the common augmentation operations are zooming, panning, changing exposure values, blurring, and imaging flipping. By randomly applying any or all of these 5 operations on the original images, we can generate a lot more training data. Our final trained model becomes much more robust.

**The Nvidia Model:**

The inputs to the Nvidia model are video images from DashCams onboard the car, and outputs are the steering angle of the car. The model uses the video images, exacts information from them, and tries to predict the car’s steering angles. At the core of the NVidia model, there is a **Convolutional Neural Network.** CNNs are used prevalently in image recognition deep learning models. The intuition is that CNN is especially good at extracting visual features from images from its various layers. The CNN layers used in the Nvidia model is very similar as above, as it extracts lines and edges in its early layers and complex shapes in its later layers.

Our Deep Learning model will apply CNNs, Transfer Learning , RNNs too.

Transfer Learning is used to apply the correct decision and observations observed in previous situations to current situation being faced which is similar to previous observation observed.

Module 7:testing

The car was trained under different combinations of the track i.e. straight, curved, combination of straight and curved and etc. Total of 24 videos were recorded out of which images were extracted.10868 images were extracted and was categorically placed in different folders like left, right, straight and stop. Below is the sample image of each of the scenario in its gray scaled version These images were resized to 320 x 240 and on which the network was trained on. The Convolutional Neural Network had 128 input nodes, 2 hidden layers of 32 nodes each and finally the output layer consisting of 4 nodes for each of the 4 output. To avoid overlearning of the network dropout of 0.5 was considered. ‘Relu’ activation function was used between the input and hidden layers and ‘Softmax’ activation function was used in the output layer. The Batch size was set to 10 and

number of epochs was set to 3 and it took 5-6 hours to train on the GPU mode. This was all about the network configuration that was used to train the model. Let’s now see how well the car performed on each track. make the system reliable but at the same time it would make the overall design attractive and risk-free from accidents.