iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 8 **Issue:** VI **Month of publication:** June 2020

DOI: http://doi.org/10.22214/ijraset.2020.6316

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VI June 2020- Available at www.ijraset.com

Voice Assisted Smart E-Cane for the Visually Challenged using Machine Learning

Adam Filbert Ashwal[1], Kimberly Claudia Dsouza[2], Meghana S[3], Muhammad Hashim Iqbal Husain[4], Manju Devi[5]
[1, 2, 3, 4]*Dept of Electronics and Communication Engineering, The Oxford College of Engineering, Bangalore, India*
[5]*Head of Department, ECE, The Oxford College of Engineering, Bangalore, India*

Abstract: ***This prototype device focuses on aiding the visually challenged in terms of movement. The device includes a Smart E-Cane implemented using Raspberry Pi and sensors for obstacle, water detection and vibration, buzzer as feedback respectively. The device also includes a Smart Band implemented using Node MCU which provides safety and security features such as sending the location to the user's relatives at home and locating the Smart E-Cane within the user's vicinity. The prototype is trained to detect staircase in real-time using YOLOv3, deep learning concept and report back to the user via speech playback. This Smart E-Cane promises independency along with security in terms of walking without assistance.***
Keywords: Blind Stick, Staircase Detection, Raspberry Pi, Node MCU, IoT, YOLOv3, Deep Learning.

I. INTRODUCTION

There are many people who suffer from visual impairment and need to go through expensive treatments or mental and physical challenges to cope with their disability. This was the ideology behind our project. The prototype consists of a Smart E – Cane and a Smart Band. The Smart E – Cane is to assist the visually challenged while the Smart Band provides safety and security features to the user.

The Smart E – Cane is implemented using Raspberry Pi microprocessor and incorporates with various sensors and feedback mechanisms that will aid the user for navigation and alerting purposes. While the most important is the detection of raised surface level such as a staircase using Raspberry Pi and Pi Camera.

The prototype also focuses on providing accurate location information when needed to the user's contacts. It includes a wearable Smart Band for emergencies. The Smart Band is implemented using NodeMCU. The framework detects an ascent in surface level such as a staircase as the individual is walking, using deep learning and reports back what sort of raised surface is ahead. This Smart E – Cane not only provides independency but security too in terms of walking without any sort of assistance.

II. LITERATURE SURVEY

After referring many papers, we found umpteen devices that aid the visually challenged and blind to go about their daily life. Few of the implementations that stood out were:

"Real Time Object Detection Using YOLOv3" by Omkar Masurekar, Omkar Jadhav, Prateek Kulkarni, Shubham Patil. They have created a model to distinguish only three objects which can be scaled additionally to detect manifold quantity of objects. In this, they had used the Bounding box method for localization of the objects to overcome the drawbacks of the sliding window method. [1]

"Smart Stick for Blind using Machine Learning" by Mohamedarif Regade, S Bibi Ayesha Khazi, Sushmita Sunkad, Mrityunjay C.K, Sharda K.S. A wall-following platform is overlaid so the user will walk straight in a passage in an inhouse environment. Head level obstacle recognition can also be carried out.

Programmable wheels to be inured to steer the stick off from the obstructions. [3] "Electronic Smart Canes for Visually Impaired People" by Dimitra P. Marini in her B.Sc Thesis book, mentioned that this project includes capturing the background via a camera and estimating the background details to form required path to be taken for the blind person. This path is then provided to the user through auditory response. [4] "Smart Walking Stick for Blind integrated with SOS Navigation System" by Saurav Mohapatra, Subham Rout, Varun Tripathi, Tanish Saxena, Yepuganti Karuna. It is a device that integrated a blind stick with Smart navigation system for emergency situations. It assists in identifying obstacles through live streaming system and directs the users next move based on the obstacle in front of them. [5]

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 8 Issue VI June 2020- Available at www.ijraset.com

III.TOOLS AND TECHNOLOGIES

A. Hardware Tools Used in Smart E-Cane

The different hardware tools that are used in the Smart E-Cane are as follows:

1) *Raspberry Pi:* The Raspberry Pi microprocessor is like a credit card size computer capable of controlling the various I/O devices connected to it. The main processor which is used to control the various hardware components used in the Smart E – Cane is the Raspberry Pi 3 model B microprocessor. The Raspberry Pi 3 is interfaced with ultrasonic sensor (HC-SR04), moisture sensor and moisture meter (YL-69 & YL-38), Raspberry Pi camera (rev 1.3, 5MP), motor driver board (L293D-R1), 2.4 GHz RF receiver, vibrator motor and magnetic buzzer.
2) *Ultrasonic Sensor:* An ultrasonic sensor is a device which is used in measuring distance or sensing objects. When triggered, the ultrasonic transmitter transmits 8 ultrasonic pulses which travel in air and the echo pin is kept at a high logic state. When the transmitted waves are reflected by an object, they are reflected back towards the sensor and are received by the ultrasonic receiver. At this instant the echo pin is brought back to the low logic state. The amount of time during which the echo pin stays high is measured by the microprocessor as it gives the time taken for the ultrasonic waves to return back to the sensor. Using this value, the distance of the obstacle can be measured using the simple speed-distance formula. We have used HC-SR04 ultrasonic sensor to detect obstacle on the path.
3) *Moisture Sensor and Moisture Meter:* The presence of wet surface is detected by using a moisture sensor along with a moisture meter. We have used a combination of YL-69 moisture sensor and YL-38 moisture meter to detect wet surfaces such as water puddles. The moisture sensor acts as a variable resistor whose resistance varies inversely according to the water content in water puddle. The YL-38 moisture meter which is attached to the moisture sensor produces a voltage (analog) which is equivalent to the resistance value. The LM393 comparator chip which is incorporated in the moisture meter is used to digitize the analog voltage signal.
4) *Raspberry Pi Camera:* The Raspberry Pi camera is a portable light weight camera module which is compatible with the different Raspberry Pi microprocessor models. It communicates with the Raspberry Pi processor using the MIPI camera serial interface protocol. We have used Raspberry Pi rev 1.3 5MP camera as the input device to record the images of the environment so that these recorded images can be processed to detect the presence of staircase in the environment.

B. Hardware Tools Used in Smart Band

The different hardware tools that are used in the Smart Band are as follows:

1) *Node MCU:* Node MCU is the pairing of firmware and hardware based around the ESP8266-12E module which is a low cost Wifi enabled microchip with a full TCP/IP stack and microcontroller capabilities. Node MCU is the main microcontroller which is used to control the various hardware components used in the Smart Band. Node MCU is interfaced with GPS module (SKG13BL), 2.4GHZ RF transmitter and the two 4-pin push buttons.
2) *GPS Receiver Module:* GPS (Global Positioning System) is a satellite-based system that sends a signal to the GPS receiver. This signal contains information such as exact time of signal transmission, orbital position of the satellite, etc. The GPS receiver uses the received information signals to calculate its distance from the GPS satellites. This is done by measuring the time required for the signals to travel from GPS satellites to the GPS receiver and then multiplying the measured time with speed of light. The GPS receiver requires signals from at least 4 GPS satellites to calculate its location accurately. We have incorporated GPS receiver module (SKG13BL) in our Smart Band to help the visually impaired user to determine and send the location details to user's relative's mobile in the case of emergency.
3) *RF Transmitter and Receiver Modules:* RF modules (transmitter and receiver) are small size electronic devices that are used to aid in wireless communication between an embedded system and another device. This communication is accomplished by using radio frequency signals. Communication over radio frequency is advantageous as it doesn't require a line of sight connection between the transmitter and receiver and the range of RF communication is very high when compared to IR communication. We have used 2.4 GHz RF transmitter and receiver modules in order to establish wireless communication between the Smart E – Cane and the Smart Band.

C. Software Tools

The major software tools which were used to program the different sensors, output devices, microprocessor and microcontroller used in our entire project are as follows:

1) *Anaconda:* Anaconda is an open source distribution of the Python and R programming languages for scientific computing that aims to simplify package management and deployment. It includes a desktop graphical user interface called Anaconda Navigator. Package management in Anaconda is done by using conda which is an open source, cross-platform package manager and environment management system that installs, runs, and updates various packages and their dependencies. We have used conda package manager to install various libraries and their dependencies which were needed in implementing staircase detection.
2) *Blynk App:* Blynk is an IoT (Internet of Things) platform which enables users to control hardware components such as different sensors remotely, display sensor data, store the sensor data and visualize it using android devices. We have used this app to display the location details of the user in the user's relative's mobile.
3) *VOTT:* VOTT (Visual Object Tagging Tool) aids user to label different objects in different images. These labelled images can be then used to train object detector models. We have used VOTT to annotate staircases in different images (250 images).
4) *VNC Viewer:* VNC (Virtual Network Computing) is a graphical desktop sharing system that uses the RFB (Remote Frame Buffer) protocol to remotely control another computer by transmitting the keyboard and mouse events from one computer to another and relaying the graphical screen updates back in the other direction over a common network. We need a VNC server application for the computer which we want to control and a VNC viewer application for the computer we want to control from. We have used VNC viewer to control Raspberry Pi 3 Model B microprocessor remotely.

D. Software Libraries

1) *Keras:* Keras is an open source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML libraries. Keras library contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions and optimizers. It also contains a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code [22].
2) *OpenCV:* OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library [23]. This library includes a set of both classic and state-of-the-art computer vision and machine learning algorithms which are used in real time image and video processing.
3) *TensorFlow:* TensorFlow is an end-to-end open source platform for machine learning [24]. This library uses data flow graphs to build models and also allows developers to create large-scale neural networks with many layers. TensorFlow library is mainly used for classification, perception, understanding, discovering, prediction and creation. It provides a variety of toolkits that allow users to construct models at their preferred level of abstraction. The library is written using python programming language and is built to run on multiple CPUs or GPUs.
4) *YOLOv3:* You Only Look Once (YOLO) is a state-of-the-art, real-time object detection system. It applies a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities [16]. YOLOv3 is an updated version of YOLO that includes additional features such as multi-scale predictions and a better backbone classifier. YOLOv3 predicts an object likelihood score for each bounding box using logistic regression. Each box predicts the classes the bounding box may contain using multi label classification [25]. One of the salient features of YOLOv3 is that it makes detections at three different scales [27]. YOLOv3 is a very strong detector that excels at producing decent boxes for objects [25].
5) *gTTS API:* gTTS (Google Text-to-Speech), a Python library and CLI tool to interface with Google Translator's text-to-speech API [26]. There are several APIs available to convert text to speech in python. One of such APIs is the Google Text to Speech API commonly known as the gTTS API. gTTS is a very easy to use tool which converts the text entered, into audio which can be saved as a mp3 file. It features flexible pre-processing and tokenizing. The speech can be delivered in any one of the two available audio speeds, fast or slow.

IV. DESIGN AND IMPLEMENTATION

A. Design

The prototype model consists of a Smart E-Cane and a Smart Band as in figure 1. The Smart E-Cane's main processor is the Raspberry Pi and consists of Ultrasonic sensor, Moisture sensor, Raspberry Pi Camera, Vibrator Motor, Magnetic Buzzer, RF Receiver module and a Motor Driver. The ultrasonic sensor and moisture sensor implement the obstacle detection and wetness detection respectively.

The feedback for obstacle detection is provided via a vibrator motor and feedback for wetness detection is provided via a magnetic buzzer. The Smart Band main controller is the Node MCU and consists of a GPS module, two push buttons and a RF Transmitter module. The Smart Band provides safety and security features to the user. The two push buttons are utilised to provide these features. Finally, the most important feature of our prototype model is the detection of staircase using Deep Learning. Here we use YOLOv3 for staircase detection while the core libraries include Keras, TensorFlow and for image processing, OpenCV is used.

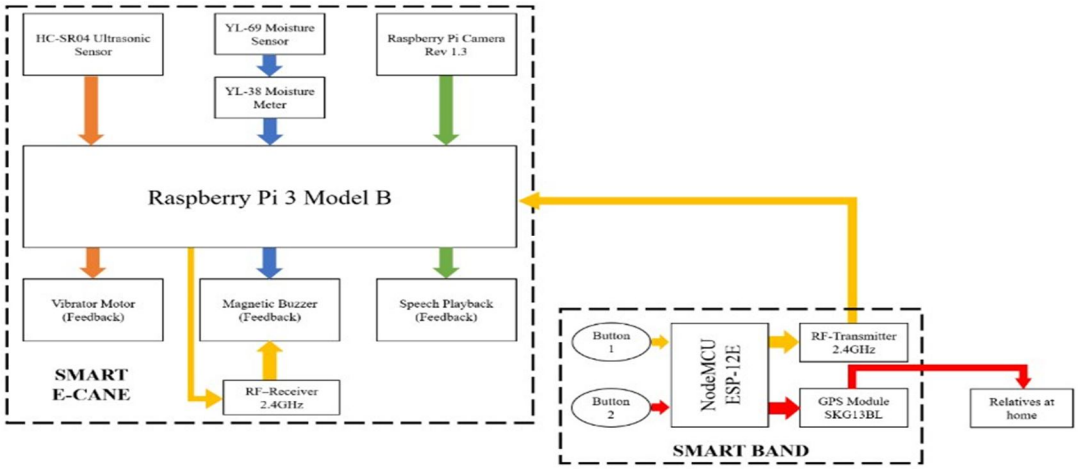


Fig. 1: Generalized Block diagram of Voice Assisted Smart E – Cane for the Visually Challenged Using Machine Learning

B. Implementation

1) Hardware: The Smart E – Cane which contains the ultrasonic sensor, moisture sensor, RF receiver module, the Pi camera and the Raspberry Pi microprocessor (shown in figure 2) is the main processing unit of this project. When the device is connected to the power source, the sensors are all active and provide constant feedback.

TABLE I. Hardware Components List

Raspberry Pi 3 Model B Microprocessor	Node MCU ESP-12E
HC-SR04 Ultrasonic Sensor	YL-69 Moisture Sensor
YL-38 Moisture Meter	Magnetic Buzzer
Vibrator Motor	4 Pin Push Button
Raspberry Pi Camera Rev 1.3	GPS Module SKG13BL
9V Power Supply	Radio Frequency TX/RX 2.4GHz
L293D R1 Motor Driver	

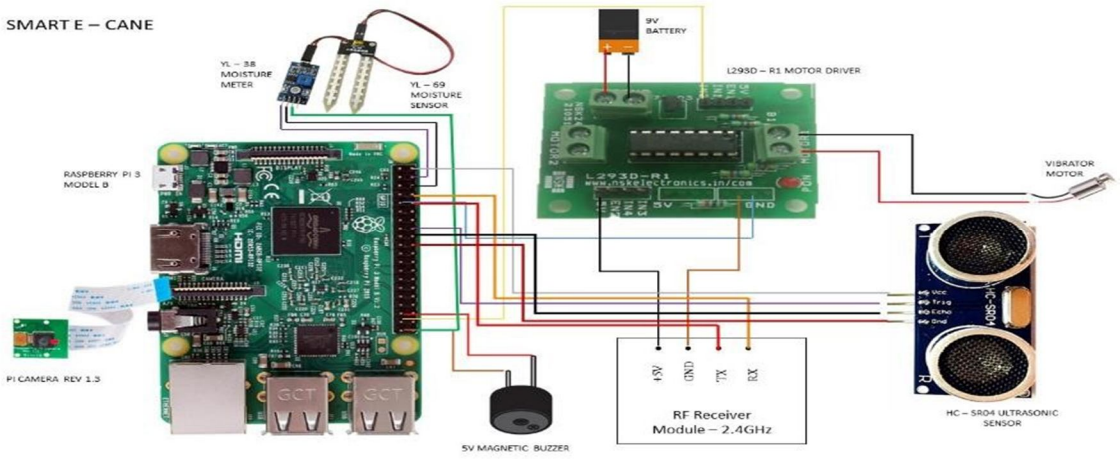


Fig. 2: Smart E – Cane pin diagram

The ultrasonic sensor uninterruptedly measures the distance to sense if there are any obstacles present in front of the user. To measure the distance the sound has travelled in air we use the formula: Distance = (Time x Speed of Sound in Air) / 2. The "2" represents the sound travelling back and forth. To read the distance as centimetres we use the formula: Centimetres = ((Microseconds / 2) / 29). Here, we take into consideration the temperature while calculating the speed of sound in air. On detecting an object within the range of 100cms, the vibrator motor is turned on and alerts the user. If the user approaches the object closer than 50cms, the intensity of vibrations increases. The moisture sensor also continuously provides feedback if in contact with water. The feedback is given via the buzzer to differentiate between obstacles.

The reason we use these two different feedback mechanisms by two different sensors is because, the probability of obstacles being detected is more compared to wet areas, hence to avoid causing disturbances to surrounding people the feedback from ultrasonic sensor (obstacle detection) is provided by vibrations by the vibrator motor as it is sound-free (sound of vibrations compared to that of the magnetic buzzer is assumed to be negligible). Moreover, the possibility of confusion is much greater when the sensors are connected to the same feedback device.

The raised surface detection is implemented using the Raspberry Pi microprocessor on the Smart E – Cane. The Raspberry Pi microprocessor receives data from the image captured by the Pi camera. Deep learning is a subset of machine learning that uses multi-layered neural network to progressively extract higher level features from the raw input data. Since this is a prototype model, only staircase detection has been trained into the machine by data sets and the deep learning mechanism is used on various staircase. The flowchart for the staircase detection is explained in the figure 3. A threshold value of 0.25 is set, so that images detected as staircases below 25% confidence value are discarded. The feedback from this process is provided via speech playback using a speaker or earphones.

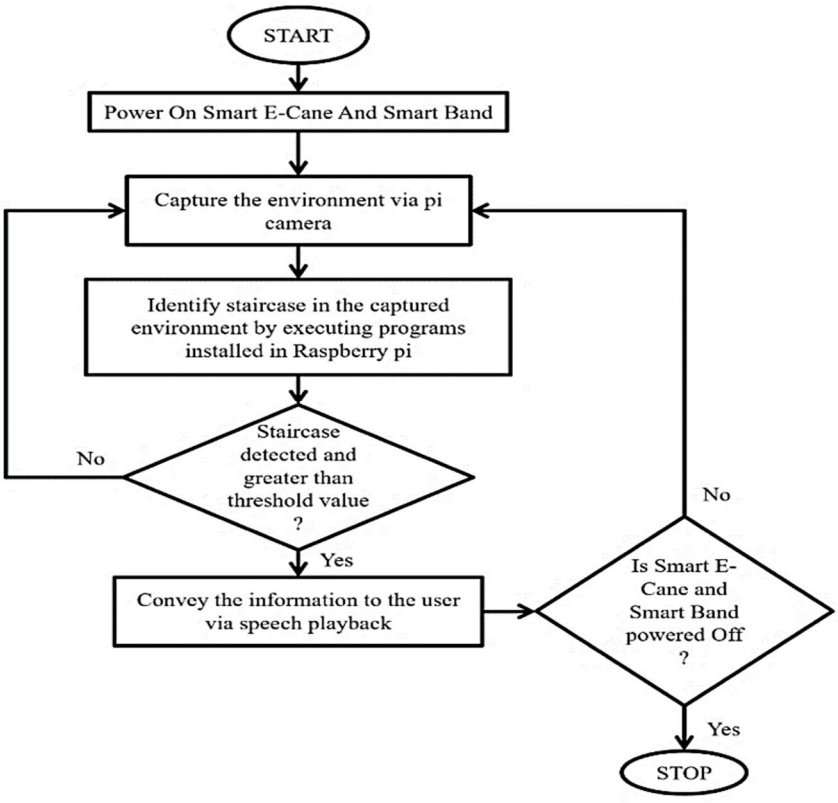


Fig. 3: Staircase Detection Using Deep Learning Flowchart

Once the code is dumped onto the processor and a power supply is provided, the Smart Band which contains the NodeMCU, the GPS module, RF transmitter module and the push buttons (as shown in figure 4) is turned on. The code contains the BlynkSimpleEsp8266 header file which was included to provide access to the Blynk app. Another library called TinyGPS is included for parsing NMEA data streams provided by GPS modules. This library provides compact and easy-to-use methods for extracting position, date, time, altitude, speed, and course from consumer GPS devices.

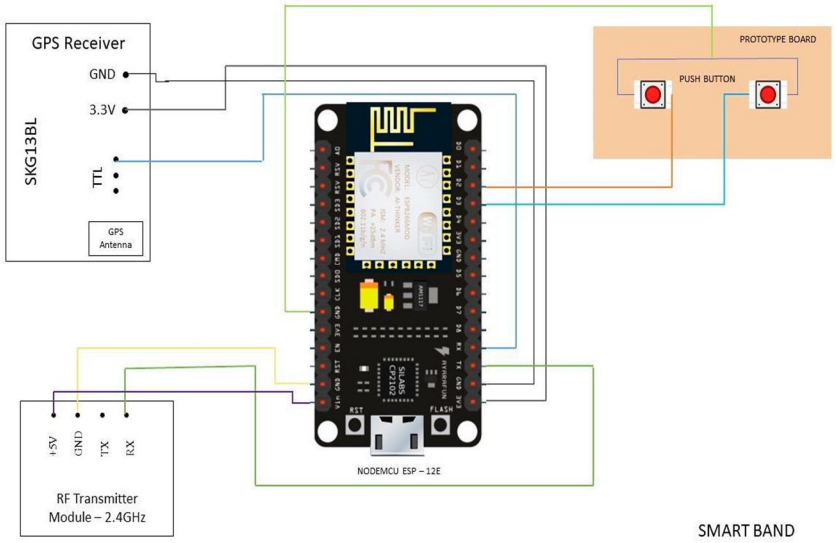


Fig. 4: Smart Band pin diagram

As seen, there are two push buttons. One works as an SOS button and the other is used to locate the Smart E – Cane. When the user presses Button 2 (the SOS button), the GPS module collects the location information and send it to Blynk (latitude first, then longitude), along with a "HELP!!!" message indicating the user is in some kind of situation that needs the attention of his contacts. The emergency contacts will be able to view his location on a map. When Button 1 is LOW, the RF transmitter sends a signal to the RF receiver on the Smart E – Cane, which activates the buzzer.

2) *Software:* The software part of our project is based on image processing using deep learning and machine learning algorithms. We use yolov3 which is a quick, real time, image processing tool.

TABLE II
Software Components List

Python 3	OpenCV
TensorFlow	YOLOv3
Blynk	Vott
Keras	Arduino IDE
VNC Viewer	Anaconda
Raspberry Pi OS	gTTS API

a) *Step 1:* Image Annotation

For detecting staircases in pictures, it needs to be fed with labelled training data. For accurate results, we labelled about 250 objects. Labelling is done using Microsoft's Visual Object Tagging Tool (VoTT). Once the images are classified and the coordinates are established, and we export it as csv file shown in figure 5.

```
"image", "xmin", "ymin", "xmax", "ymax", "label"
"9k_%20(1).jpg", 79.57432922407541, 3.0179032258064513, 215.0891950688905, 143.72129032258064, "Staircase"
"9k_%20(2).jpg", 59.43710609243698, 6.045714285714285, 174.3266619147659, 126.4342857142857, "Staircase"
"2Q__%20(1).jpg", 26.52356780275562, 15.133688842482101, 106.929659173314, 87.18022822195704, "Staircase"
"2Q__%20(2).jpg", 86.63551136363635, 2.3005714285714283, 196.61732954545454, 124.23085714285713, "Staircase"
"2Q__.jpg", 3.4578606592465753, 0.8868571428571429, 106.12652504280823, 103.31885714285714, "Staircase"
"9k_%20(4).jpg", 95.65192168237853, 29.751827267303103, 150.57868020304568, 79.45349791169451, "Staircase"
"9k_%20(5).jpg", 135.09789702683102, 4.32421875, 265.69253081943435, 130.57421875, "Staircase"
"9k_%20(3).jpg", 65.82857142857142, 48.857142857142854, 177.17142857142855, 152.74285714285713, "Staircase"
"9k_%20(6).jpg", 145.25868486352357, 54.57142857142857, 212.95099255583125, 129.22514285714286, "Staircase"
"9k_%20(7).jpg", 80.65844815083393, 39.45290322580645, 179.530094271211, 137.41790322580647, "Staircase"
"9k_%20(7).jpg", 148.09064539521393, 0, 248.48005801305294, 100.78306451612903, "Staircase"
"9k_.jpg", 25.509090909090908, 0, 185.95369318181815, 161.66742857142853, "Staircase"
"Z%20(2).jpg", 187.30964467005077, 0, 268.67295141406817, 69.77331606217615, "Staircase"
"Z%20(3).jpg", 13.483670295489889, 0, 247.25349922239502, 170.41142857142862, "Staircase"
"Z%20(4).jpg", 79.95935828877006, 58.28, 184.91764705882355, 171.61599999999999, "Staircase"
"Z%20(1).jpg", 46.416052756654, 3.7645714285714282, 142.6137713878327, 100.17942857142856, "Staircase"
"Z.jpg", 117.18662827557058, 22.057142857142857, 235.000660397295, 127.49028571428573, "Staircase"
"Z.jpg", 5.550110946745562, 84.25828571428572, 103.50784551986476, 179.54514285714288, "Staircase"
"images%20(1).jpg", 8.831926323867997, 9.252571428571429, 194.93323100537222, 168.64914285714286, "Staircase"
"images%20(10).jpg", 55.808, 23.296, 118.52799999999999, 82.176, "Staircase"
```

Fig. 5: Annotations_export.csv

In the final step, we convert the VoTT csv format to the YOLOv3 format by, running the conversion script within the Image_Annotation folder:

>>> python Convert_to_YOLO_format.py

```
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(1).jpg 80,3,215,144,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(2).jpg 59,6,174,126,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/2Q__%20(1).jpg 27,15,107,87,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/2Q__%20(2).jpg 87,2,197,124,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/2Q__.jpg 3,1,106,103,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(4).jpg 96,30,151,79,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(5).jpg 135,4,266,131,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(3).jpg 66,49,177,153,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(6).jpg 145,55,213,129,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_%20(7).jpg 81,39,180,137,0 148,0,248,101,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/9k_.jpg 26,0,186,162,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/Z%20(2).jpg 187,0,269,70,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/Z%20(3).jpg 13,0,247,170,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/Z%20(4).jpg 80,58,185,172,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/Z%20(1).jpg 46,4,143,100,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/Z.jpg 117,22,235,127,0 6,84,104,180,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/images%20(1).jpg 9,9,195,169,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/images%20(10).jpg 56,23,119,82,0
/home/ubuntu/TrainYourOwnYOLO/Data/Source_Images/Training_Images/vott-csv-export/images%20(100).jpg 109,86,154,124,0
```

Fig. 6: data_train.txt

This script generates two output txt files: data_train.txt and data_classes.txt. The data_train file is shown in figure 6 and the data_classes.txt contains the class of the recognizable images which, in our project, is the staircase.

b) Step 2: Training

By means of the training images and the annotation file data_train.txt which we have created in the earlier step we now train our YOLOv3 detector. Before this we download the pre-trained YOLOv3 weights and convert them to keras format. These weights are pre-trained under ImageNet 1000 dataset and thus work quite well for object detection that are very similar to the images and objects in the ImageNet 1000 dataset. The final weights are saved in Model_weights.

c) Step 3: Inference

In this step, we test our detector on staircase images located in Test_Images. The outputs are saved and stored into the Test_Image_Detection_Results. The outputs include the original images with bounding boxes and confidence scores as well as a file called Detection_Results.csv containing the image file paths and the bounding box coordinates.

C. Sequence Flow

When the device is ON, the Raspberry Pi runs the scheduled processes. Once the Ultrasonic sensor detects the object that is in its range, it activates the vibrator through the Raspberry Pi. A similar process takes place when the moisture sensor comes in contact with water; it activates the buzzer through the processor. When the Pi camera captures an image, it stores the image. Based on the parameters and trained model sets, the image is processed by yolov3 and if the staircase is detected, it sends the feedback to the processor which then activates a voice message stating the obstacle that is detected.

When the Button 1 on the Smart Band is pressed, it activates the buzzer through Radio frequency transmission. When the Button 2 on the Smart Band is pressed, it initiates the GPS module to send location via IoT back to the NodeMCU which then sends that location with an alert message to the application of the contacts having the Blynk application.

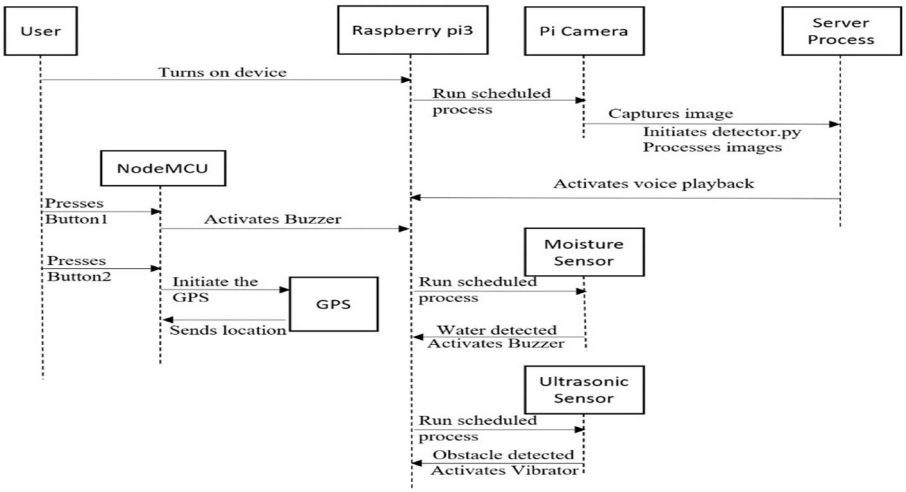


Fig. 7: Sequence Flow of Raspberry Pi and NodeMCU

V. RESULTS

The Smart E – Cane and Smart Band devices are powered on when a power source of 9V or more is provided to them. In order to initiate the working of Smart E – Cane and to view the output from various sensors along with the status of output devices they are connected to, we execute the run command in the terminal window of Raspberry Pi.

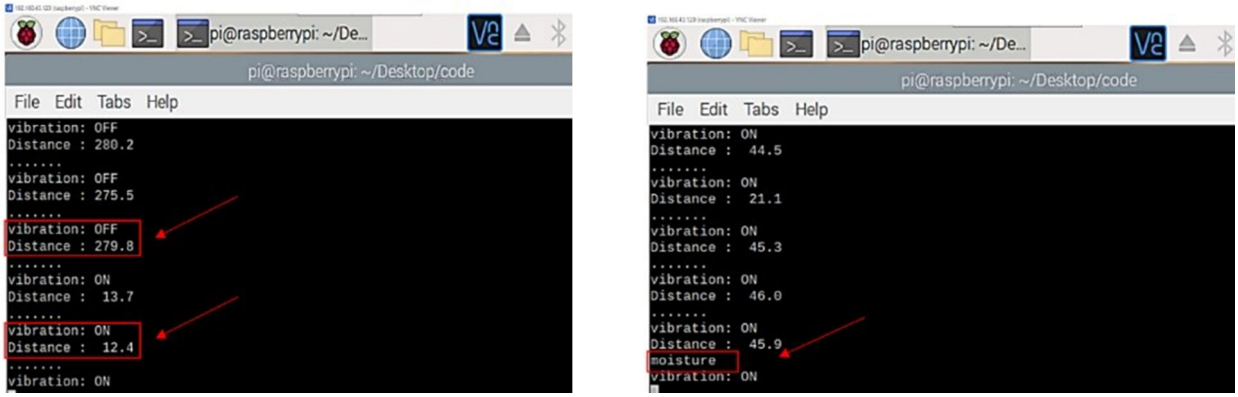


Fig. 8: (Left) Output of Ultrasonic sensor showing distance from object with vibrator status. (Right) Output on command prompt showing water detected by moisture sensor

When Button 1 on the Smart Band is pressed to locate the Smart E – Cane, the receiver on the Smart E – Cane receives the signal from the transmitter on the Smart Band and activates the buzzer on the Smart E – Cane. This is depicted in figure 9.

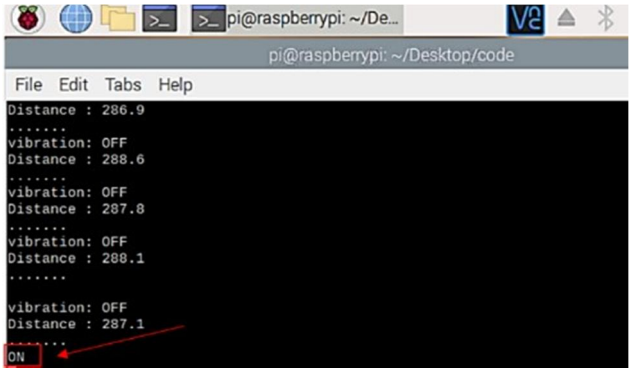


Fig. 9: Button 1 on Smart Band pressed, activating buzzer on Smart E – Cane

When the Button 2 on the Smart Band is pressed, the GPS module obtains and sends the location to the NodeMCU which then sends it to the Blynk app, which alerts the respective contact of the user. When the button isn't pressed, the status is left blank as shown in figure 10.

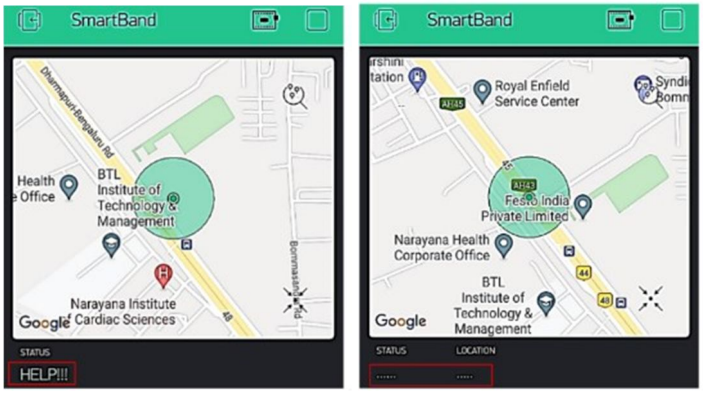


Fig. 10: (Left) Location of user with alert notification on Blynk app of the emergency contact. (Right) Location of the user with blank status on Blynk app

When the user comes in front of an obstacle such as the staircase, the processor detects it and produces an output as shown in figure 11. A voice message alerts the user if the staircase is detected. We have used YOLOv3 object detection algorithm in our project to detect staircase. YOLOv3 produces a confidence score beside the object class when detecting an image. The score is a number between 0 and 1 that indicates confidence that the object was genuinely detected [24]. The closer the number is to 1, the more confident the model is. In our staircase detector model, we have defined cut-off threshold level as 0.25. In that case, we would ignore the remaining objects because those confidence scores are below 0.25.

Fig. 11: Staircase detection with a confidence score of 0.80 or 80%.

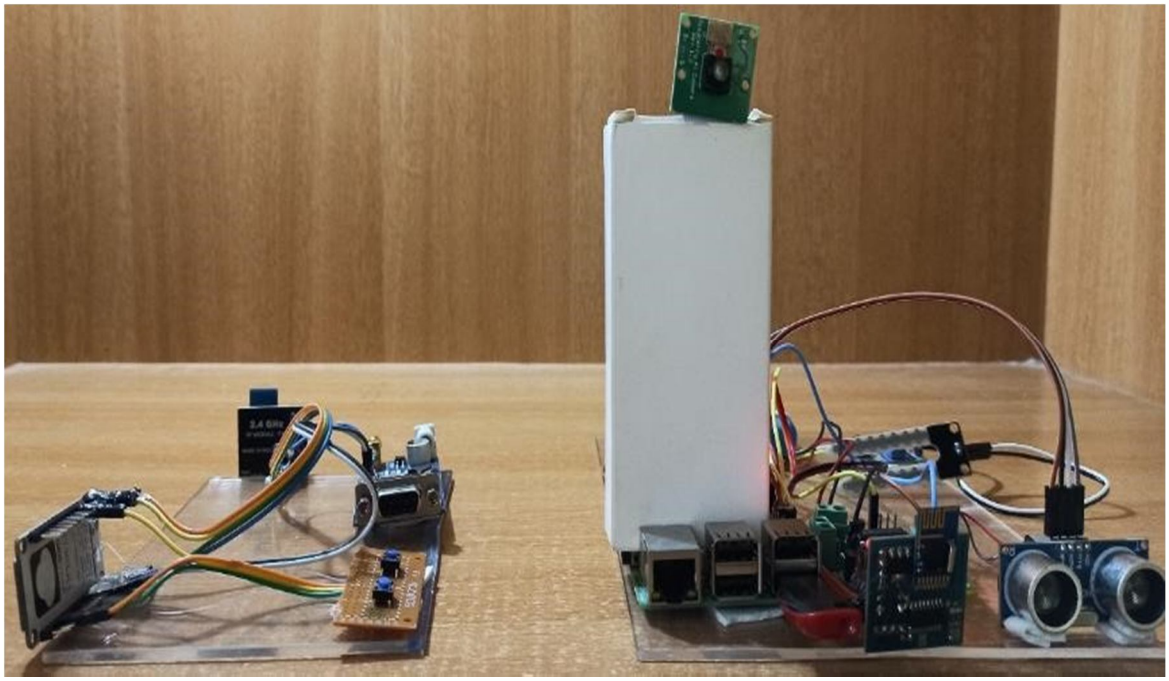


Fig. 12: Prototype model of Voice Assisted Smart E-Cane for the Visually Challenged Using Machine Learning

VI. CONCLUSIONS

Voice Assisted Smart E-Cane for the Visually Challenged Using Machine Learning, this prototype model is a great electronic aid for the visually challenged and the blind. It not only alerts the user if the obstacle is detected but also provides security for the user. The Smart E-Cane with Raspberry Pi microprocessor detects the obstacles using ultrasonic sensors and alerts the user by vibrator motor. Intensity of the vibrations depends on how far the obstacle is located, if the obstacle is far the intensity is low otherwise the intensity is high.

The Smart E-Cane detects the water in the surrounding using moisture sensor and alerts the user by buzzer sound. Most important feature is the detection of staircase and it is done using Deep Learning and Neural Network concepts, where it uses YOLOv3 the latest, popular object detection algorithm. If the staircase is detected the voice playback saying "STAIRCASE FOUND" is played.

VII. FUTURE SCOPE

Our prototype model can further be powered by solar cells instead of Li-ion battery as they have to be replaced often. Can establish wireless communication between the components to achieve the compactness of the system. As the prototype model only detects staircase, further it can be trained to detect other elevated surfaces like ladder, an inclined plane, ramp etc. To increase the processing speed and to reduce the delay occurred, one can go with GPU (Graphical Processing Unit) or high-speed processors. Moreover, earphones can be made use for the better clarity of the voice playback and to speak to the concerned person at any location.

REFERENCES

[1] Omkar Masurekar, Omkar Jadhav, Prateek Kulkarni, Shubham Patil, *"Real Time Object Detection Using YOLOv3"*, 2020, International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056. Volume: 07 Issue: 03, Mar 2020.

[2] Habib, A., Islam, M.M., Kabir, M.N., Mredul, M.B., Hasan, M. (2019). *"Staircase detection to guide visually impaired people: A hybrid approach"*, Revue d'Intelligence Artificielle, Vol. 33, No. 5, pp. 327-334. Accessible at: https://doi.org/10.18280/ria.33050.

[3] Mohamedarif Regade, S Bibi Ayesha Khazi, Sushmita Sunkad, Mrityunjay C.K, Sharda K.S, *"Smart Stick for Blind using Machine Learning"*, 2019, International Journal of Innovative Science and Research Technology ISSN No: 2456-2165, Volume 4, Issue 5. May 2019.

[4] Dimitra P. Marini, BSc THESIS: *"Electronic Smart Canes for Visually Impaired People"*, 2018, National and kapodistrian university of athens school of science department of informatics and telecommunication. Athens.

[5] Saurav Mohapatra, Subham Rout, Varun Tripathi, Tanish Saxena, Yepuganti Karuna, *"Smart Walking Stick for Blind integrated with SOS Navigation System"*, 2018, Proceedings of the 2nd International Conference on Trends in Electronics and Informatics (ICOEI 2018) IEEE Conference Record: # 42666, IEEE Xplore ISBN:978-1-5386-3570-4.

[6] Kunja Bihari Swain, Rakesh Kumar Patnaik, Suchandra Pal, Raja Rajeswari, Aparna Mishra and Charusmita Dash, *"Arduino based Automatic Stick Guide for a Visually Impaired Person"*, 2017, IEEE International conference on smart technologies and management for computing, communication, controls, energy and materials (ICSTM) 4 August 2017, Veltech Dr. RR & Dr. SR university, Chennai, TN, India.

[7] Akhilesh Krishnan, Deepakraj G, Nishanth N, Dr.K.M.Anandkumar, *"Autonomous Walking Stick For The Blind Using Echolocation And Image Processing "*, 2016, 2nd International Conference on Contemporary Computing and Informatics.

[8] S. Murali, R. Shrivatsan, V. Sreenivas, S. Vijjappu, S. J. Gladwin and R. Rajavel, *"Smart walking cane for the visually challenged"*, 2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Agra, 2016, pp. 1-4, doi: 10.1109/R10-HTC.2016.7906791.

[9] Alejandro R. Garcia Ramirez, Renato Fonseca Livramento da Silva, Milton Jose Cinelli, Alejandro Durán Carrillo de Albornoz, *"Evaluation of Electronic Haptic Device for Blind and Visually Impaired People: A Case Study"*, 2015, Journal of Medical and Biological Engineering 2015, 32(6): 423-428.

[10] Mohammad Hazzaz Mahmud, Rana Saha, Sayemul Islam, *"Smart walking stick - an electronic approach to assist visually disabled persons"*, 2013, International Journal of Scientific & Engineering Research, Volume 4, Issue 10, ISSN 2229-5518, October 2013.

[11] S.Koley and R. Mishra, *"Voice Operated Outdoor Navigation System for Visually Impaired Persons"*, 2012, International journal of engineering trends and technology, Vol.3 Issue 2.

[12] Farnel, Newark, element14, *"Raspberry Pi 3 Model B"*, URL: https://us04web.zoom.us/j/71160940605?pwd=ZEZUeTFNcG1 naHA3ZjRvWmlEV3dGdG09, Retrieved on 04/05/2020.

[13] Components 101, *"Pi Camera Module – 5MP"*, URL: https://components101.com/misc/Pi-camera-module, Published on 09/12/2018, Retrieved on 04/05/2020.

[14] Proto Supplies, *"ESP8266 NodeMCU V1.0 ESP-12E Wi-Fi Module"*, URL: https://protosupplies.com/product/esp8266-nodemcu-v1-0-esp-12e-wifi-module/, Retrieved on 06/05/2020.

[15] Amy Unruh, *"What is the TensorFlow machine intelligence platform?"*, OpenSource, 09 Nov 2017. Available at: https://opensource.com/article/17/11/intro-tensorflow. Accessed on 06/05/2020.

[16] Redmon, Joseph and Farhadi, Ali, *"YOLOv3: An Incremental Improvement"*, yolov3, arXiv, 2018. URL: https://pjreddie.com/darknet/yolo/. Accessed on 06/05/2020.

[17] Lucy, *"How does VNC technology work?"*, RealVNC, August 23, 2019. Available at: https://help.realvnc.com/hc/en-us/articles/360002320638-How-does-VNC-technology-work-, Accessed on 07/05/2020.

[18] Tegan, *"Understanding VNC Server Modes"*, RealVNC, August 15,2019. Available at: https://help.realvnc.com/hc/en-us/articles/360002253238-Understanding-VNC-Server-Modes, Accessed on 12/05/2020.

[19] diycode, *"VoTT"*, diycode, Available at: https://www.diycode.cc/projects/Microsoft/VoTT, Accessed on 20/05/2020.

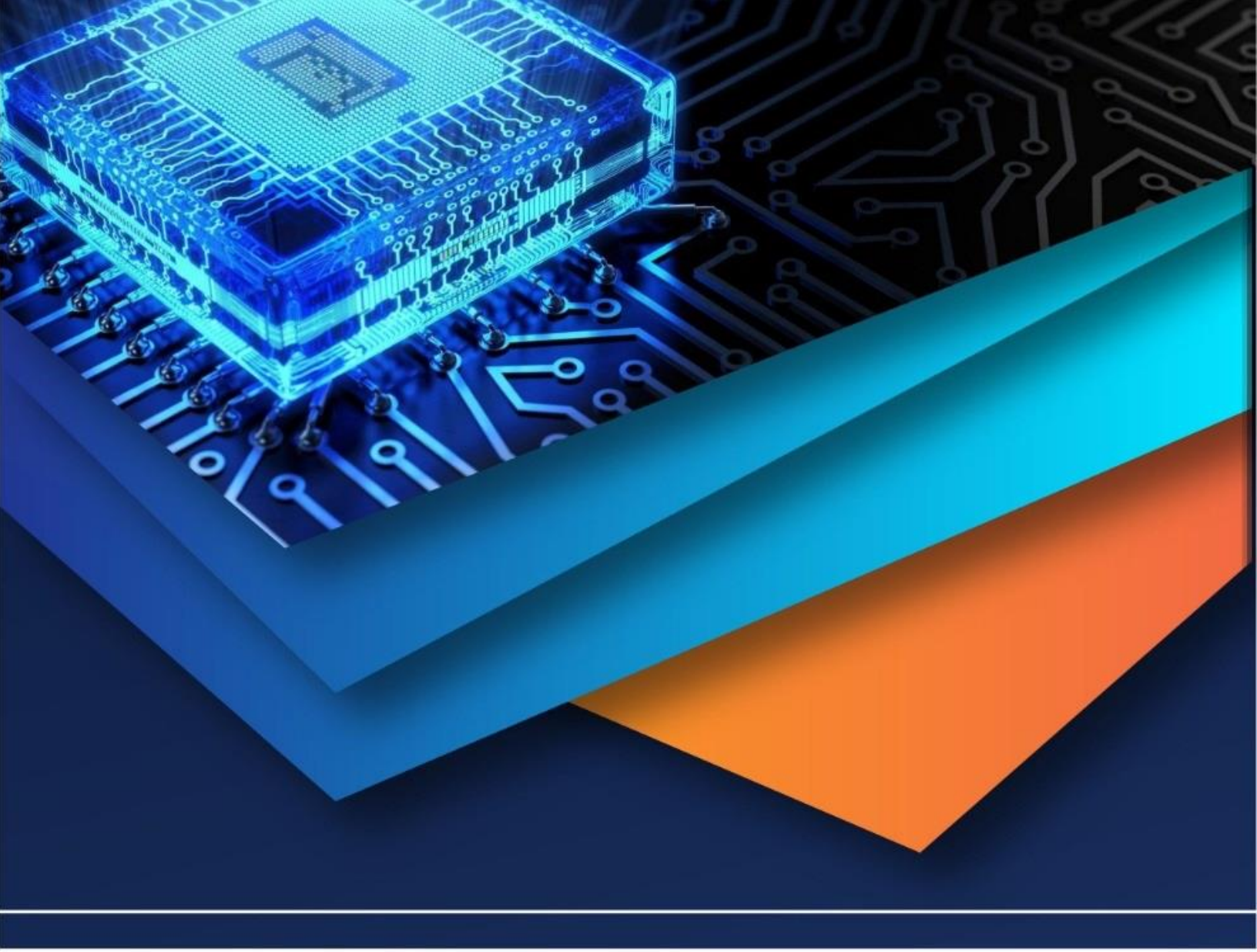
[20] tutorialspoint, *"Python 3 Tutorial"*, tutorialspoint, Available at: https://www.tutorialspoint.com/python3/index.htm, Accessed on: 22/05/2020.

[21] Raspberry Pi foundation, *"Raspberry Pi OS"*, Raspberry Pi organisation, Available at: https://www.raspberrypi.org/documentation/raspbian/, Accessed on 01/06/2020.

[22] Wikipedia, "Keras", Wikipedia, the free encyclopedia, Accessible by URL: https://en.wikipedia.org/wiki/Keras#:~:text=Keras%20is%20an%20open%2Dsource,friendly%2C%20modular%2C%20and%20extensible, Accessed on 08/06/2020.

[23] OpenCV Team, *"About OpenCV"*, OpenCV, Accessible at URL: https://opencv.org/about/, Accessed at 08/06/2020.

[24] TensorFlow, *"Object Detection"*, For Mobile and IoT, TensorFlow, 04/06/2020. Available at: https://www.tensorflow.org/lite/models/object_detection/overview, Accessed on: 08/06/2020.
[25] Joseph Redmon, Ali Farhadi, *"YOLOv3: An Incremental Improvement"*, University of Washington, Accessible at: https://pjreddie.com/media/files/papers/YOLOv3.pdf
[26] Pierre Nicolas Durette, *"gTTS"*, Built with Sphinx using a theme provided by Read the Docs, Accessible at: https://gtts.readthedocs.io/en/latest/. Accessed on 22/06/2020.
[27] Ayoosh Kathuria, *"What's new in YOLO v3?"*, Towards Data Science, A Medium publication sharing concepts, ideas, and codes. Available at: https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b, Accessed on 22/06/2020.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY