The Deep Learning based Smart Navigational Stick for Blind People

Article · March 2023

DOI: 10.32350/umtair.22.05

4 authors, including:

Muhammad Sulaman
Balochistan Think Tank Network

UMT Artificial Intelligence Review (UMT-AIR)
Volume 2 Issue 2, Fall 2022
ISSN(P): 2791-1276 ISSN(E): 2791-1268
Homepage: https://journals.umt.edu.pk/index.php/UMT-AIR

Article QR

Title: **Deep Learning based Smart Navigational Stick for Blind People**

Author (s): Muhammad Sulaman[1], Sibghat ullah Bazai[2], Muhammad Akram[3], Muhammad Akram khan[2]

Affiliation (s): [1]IT Administrator Balochistan Think Tank Network (BTTN) Quetta, Pakistan
[2]Department of Computer Engineering BUITEMS Quetta, Pakistan
[3]Department of Software Engineering BUITEMS Quetta, Pakistan

DOI: https://doi.org/10.32350.umt-air.22.05

History: Received: September 17, 2022, Revised: November 10, 2022, Accepted: December 12, 2022

Citation: Muhammad Sulaman, Sibghat Ullah Bazai, Muhammad Akram, Muhammad Akram khan, "Deep learning based smart navigational stick for blind people," *UMT Artif. Intell. Rev.*, vol. 2, no. 2, pp. 00–00, 2022, doi: https://doi.org/10.32350.umt-air.22.05

Conflict of Interest: Author(s) declared no conflict of interest

A publication of
Department of Information System, Dr. Hasan Murad School of Management
University of Management and Technology, Lahore, Pakistan

Deep Learning-Based Smart Navigational Stick for Blind and Visually Impaired People

M. Sulaman[1*], S.U.Bazai[2] , M. Akram[3] , and M.A.Khan [2]

[1]Balochistan Think Tank Network,Quetta, Balochistan, Pakistan;
[2]Department of Computer Engineering, BUITEMS, Quetta, Balochistan, Pakistan
[3]Department of Software Engineering BUITEMS Quetta, Pakistan

***Abstract*-Blind and visually impaired people find difficulty in detecting obstacles and recognizing people in their way, which makes it dangerous for them to walk, to work, or to go in a crowded area/place. They have to be cautious all the time to move, while avoiding any solid obstacles in their way. Typically, they use different aid devices to reach their destination or to accomplish their daily task. The normal stick is useless for blind and visually impaired people since it cannot detect barriers or people's faces. Visually impaired individuals are unable to distinguish between different types of objects in front of them. They are unable to gauge the size of an object or its distance from them. Several works have been done by public individuals and scientific investigators but their work is dearth in technological aspect. This technological aspect need to be addressed by adding artificial intelligence (AI). This prototype aims to help blind and visually impaired individuals in several aspects to simply obtain/perform everyday tasks.and help these individuals to live with the same confidence as sighted people live.Therefore, this study inclined deep learning Mobile-Net Single Shot MultiBox detection (SSD) algorithm for object recognition and Dlib library for face recognition. Subsequently, the proposed solution is using an Open CV and Python. Additionally, Ultrasonic sensors are used for distance measurement, which can be a great help for visually impaired people. These components are grouped together to work effectively and efficiently for the development of visually impaired people. The recognition procedure was revealed through headphones, which notifies the visually impaired when face or any object get recognized. Inclusively, the innovative solution would be a great aid for the blind and visually impaired individuals. As a result, to test and validate the accuracy of the smart navigational stick, several experiments have been conducted on a range of objects and faces. Hence, this study's modified navigational system was adequate and valid for visually impaired people.**

[*] Corresponding Author: muhammadsulaman996@gmail.com

***Index Terms*- artificial intelligence, deep learning, dlib library, face recognition, mobile-net ssd, open CV, object recognition, python**

I. Introduction

Vision (visual perception) is a valuable blessing and the most important belongings that anybody would ever like to lose. The eye with vision is just like a window through which an individual can see all the excellent things of this world. This vision enables us to distinguish and perceive between different items, to perform daily routine tasks and jobs. There is a large number of individuals, referred to as visually impaired who have totally or mostly lost their vision.

The World Health Organization (WHO) estimated that 2.2 billion people worldwide suffer from a distant or near visual impairment [1]. About 1.12 million people are blind and 1.09 million people suffer from near or far vision impairment in Pakistan [2]. The estimated population of Pakistan is 220 million [3]. It is extremely troublesome for visually impaired people to identify and perceive any obstacle in their way. They could maintain a strategic distance from it to get out of damage. Numerous individuals come up with discrete sticks for blind and visually impaired people, which consists of several features but technological aspects are not addressed properly [4].

Multi-Functional Blind Stick with several functions, is demonstrated in [4]. This strategy uses the Internet of Things (IoT) concept of this research is to remove barriers between blind individuals and their environment. This stick identify anomalies like stairs and damp terrain, a number of sensors were employed in this study. The smart blind stick prototype is easy-to-use, high-tech device that has the internet of things sensors and modules. Additionally, this system offers a means of informing concerned parties about its location via text messages or phone calls. In addition to the foregoing, a software programme is made by which friends and family members can configure the stick for the user's convenience.

A smart blind stick in paper [5] deals with the problems of blind or visually impaired people being unable to navigate without bumping into other people or objects. This smart stick allows blind individuals to navigate safely and independently by sending audio alerts through an earpiece to their phone when obstacles like water, walls, stairs or muddy ground are encountered. This device acts as a companion for the blind when they are walking. Similar to a white cane, this method helps the blind monitor their environment by monitoring landmarks or obstacles. This device is equipped with an ultrasonic sensor and water detection sensors that determine if there is a puddle or obstruction in their way.

The intelligent smart blind stick that enables blind or visually impaired people to walk independently and completely relieved the cause of any mishap. The device's main goal is to enable blind or visually impaired person to navigate their surroundings without getting any help from others. The blind stick is an arduino-based, bluetooth-enabled hardware device that helps people with low vision navigate their surroundings. It consists of three ultrasonic sensors, a panic button, a navigation switch, and a soil moisture sensor. When the user approaches a floor surface that is too slippery or wet for them to walk on, the smart blind stick's bottom sensor detects this fact and automatically alerts them to a potential hazard [6].

An affordable and reliable blind-accessible stick that helps blind individuals navigate their daily lives. The device has an ultrasonic sensor, infrared sensor, vibration motor, and buzzer for alarm. It also detects impediments in front of the blind user. One of the biggest problems for blind or visually impaired people when going up or down stairs is not knowing when one is present. By incorporating a feature that alerts the user when a staircase is present. This device contains a built-in GPS module and a GSM module that enable position tracking and display on a smartphone app. The device was equipped with ultrasonic and infrared sensors that could detect objects up to 150 cm away from the user. However, the smart blind stick offers a number of benefits, such as affordability, the capacity to detect impediments above knee height, identification of stairways, location monitoring through smartphone app, and others. More experiments would need to be performed in order to ascertain the accuracy and dependability of the system in practical situations [7].

The proposed smart stick [8] can detect obstacles and water using ultrasonic and water sensors. When the stick detects obstacles it vibrates to alert the user using RF Module and GPS-GSM module. However, previous researchers have covered different aspects but they did not use Artificial Intelligence to propose any idea. The iWalk stick which uses an ultrasonic sensor to find impediments and a water sensor to find water before activating a buzzer. iWalk is made up of a wireless RF remote control that makes noises when a button is pressed. This paper used different equipments to build a perfect prototype, however, due to the lack of technological advancements, there remains a gap for future researchers to explore [9].

An intelligent blind stick [10] uses an ultrasonic sensor and a water sensor. A buzzer would be activated if an obstacle gets detected near to the stick. This paper is limited by its lack of integration of face and object recognition, which may impede the practical application of the proposed solution and hinder its potential impact in real-world scenarios. Another prototype named blind stick [11] made a smart vest that vibrates to alert blind individuals from obstacles by taking the help of different ultrasonic sensors. The authors of this paper ignored integrating face and object recognition. A stick proposed by Jismi Johnson [12] consisted of a GPS and a GSM module, which is used to send an SMS to the user in case he losses his stick. This paradigm also consists of an ultrasonic sensor for obstacle detection. The boundaries are not to integrate face and object recognition.

In contrast to the challenges faced by visually impaired or blind individuals in their everyday lives, this study proposes a prototype of an intelligent smart stick that integrates both face and object recognition technologies. The smart stick provides a sense of security to the user by identifying potential obstacles that may pose a threat to their safety. As individuals with visual impairments often face difficulties in exploring the outside world and understanding complex situations, the smart stick can assist them in navigating unfamiliar environments and becoming more familiar with their surroundings. By enhancing their ability to perceive the world around them, the smart stick can ultimately improve their overall quality of life.

A. Problem Statement

Blind and visually imparired individuals find difficulties in recognizing faces and obstacles in their way. Taking only a local stick in their hand is difficult for them to walk to travel with the same confidence as sighted people. They always depend on others, while walking, travelling, and working.

They feel insecure whether they are in a crowd or traffic areas. Without vision, blind or visually impaired people may find it difficult to move, whether they are in a room or in a corridor without stumbling into things. Even with a tool like a walking stick, avoiding obstacles may be difficult, awkward, and even incorrect to avoid things. The disadvantage of local cane is its failure to recognize obstacles and faces. The difficulties of blindness and visual impairment are considerable. Blind and visually challenged persons are unable to recognize people and things in their path, which means that any obstacle, anything even a piece of furniture, or a brick wall may suddenly crash into them and cause severe injury. They have no sense of distance, relying instead on others to guide them toward their destination.

B. Research Motivation

A substantial portion of the blind population in our society finds it difficult to carry out their routine tasks. When crossing roadways, seeing obstructions, and others, these folks especially require assistance from others. These occurrences have compelled the researcher to create and explore a smart blind stick that would be extremely helpful for blind or visually impaired people. This study aims to provide support to the visually impaired and blind by providing them with a tool that would allow them to participate fully in society as functioning individuals.

C. Research Contributions

This study aims to extend the limited research for making and proposing various types of gadgets for blind and visually impaired people. This limited research is not completing all the requirements of blind and visually impaired individuals. This study is among the first to consider face recognition, object recognition, and measuring the distance from the object. No previous study to the best of the author's knowledge and through search in peer-reviewed papers has empirically explored this idea before. Previous researches is a defect in the technological aspect.

II. Review of Existing Devices

With the progression of innovation, numerous individuals have stepped up created and developed different types of prototypes for visually impaired and blind people. The highlights and advantages of these items depend on different kinds of sensors and other hardware components with which they are equipped. The ultrasonic sensor, which is used to identify barriers and gauge the distance to an item, is the most often used in intelligent smart sticks.

Still suggested works are not sufficient to figure out the blind or visually impaired’s problems. Many folks used Arduino, Raspberry Pi, and Google APIs but the problem was with internet connectivity, face recognition, object recognition, and accuracy in a single prototype. Such proposed solutions do not fit well to fulfil the needs of the blind or visually impaired individuals.

This paper proposes the integration of an ultrasonic sensor as an associated supersonic device to detect obstacles. The utilization of multiple sensors in this device allows for the detection of obstacles in the environment. When an object is detected, the device alerts the blind individual through the use of vibratory motors. The presence of warmth (above 70 deg. Celsius) is measured using a victimization LM35 temperature sensor. The the limitations of this study is its unsatisfactory accuracy in identifying individuals and objects, which hinders its ability to effectively address relevant problems. [13]

The main component of the suggested smart stick is an embedded system. In which a pair of ultrasonic sensors are utilised to locate obstructions in front of the blind or visually impaired up to 400 cm in front, from ground level to head level. Upward and downward steps are identified using an infrared sensor. The microcontroller receives the data that these sensors have collected. It analyses the information and starts the motor vibrating through an earphone, it summons the appropriate spoken warning message. The spread of water is detected using a water sensor. The circuits are powered by a rechargeable battery. The study used several sensors at once, which may affect the suggested system's accuracy [14].

This proposed system uses ultrasonic sensors, a buzzer, and a vibrating motor to identify an obstacle and inform the blind person when an impediment is identified. The researchers found that any obstruction to the right or left indicates a mistake. The time delay of the buzzer was also observed, while turning it on and off. However, this proposed system does not give a complete solution to the blind individual [15].

A prototype of wearable smart glasses was developed to help blind or visually impaired individuals to navigate their environment. This device consists of an intelligent smart stick, which is attached to the person's finger or wrist by an adhesive bandage and detects obstacles using an ultrasonic sensor. If the blind individual gets lost somewhere or becomes injured, then the smart stick sends a message to their relatives [16].

The stick is combined with ultrasonic, water, and light sensors. Ultrasonic sensors are used for detecting obstacles when the obstacles get detected, then data is passed to a microcontroller. It processes data and calculates the distance from a obstacle. Buzzer activates if the object near the stick gets detected by a sensor. This stick also allows the user to detect lightness or darkness in a room. The RF-based remote has been used to find stick; thus, detects obstacles, measures distance, and helps blind or visually impaired individuals to find misplaced sticks if misplaced. This stick provides no accurate path or position of any obstacle [17].

The suggested walking stick replaces the conventional walking stick. This system made use of an Arduino Nano, an LCD, a voltage regulator, an IR sensor, a speech playback module, and an ultrasonic sensor. Arduino nano is a microcontroller, which controls all the components and does calculations with high accuracy. The ultrasonic sensor is used for detecting obstacles. IR Sensor is used for motion detection. The voice playback module shall support the blind individual to reach the destination via the command or microphone. Limitations are not to provide the accurate path and position of the obstacles [18]. This proposed system incorporates multiple ultrasonic sensors. This system used a buzzer, which notifies the user when an obstacle gets detected. The concept of this paper is very basic and has not used any advanced techniques [19].

This proposed solution is the implementation of a smart stick for blind or visually impaired people by incorporating and taking the assistance of an ultrasonic sensor. They utilized an ultrasonic sensor to detect the obstacles in front of blind individuals. The smart stick vibrates when an ultrasonic sensor detects an obstacle near or in front of itself. They also used GPS and GSM modules for sending the user's location to relatives in case the blind individual is lost. This paper aims to detect obstacles and share the user's location with relatives. The limitations of the system are not being able to recognize faces and obstacles in front of blind or visually impaired individuals [20].

A. Mobile-Net SDD Algorithm

Single Shot MultiBox Detector (SSD) is a well-liked approach for detecting objects. It generally performs faster than Faster RCNN but it requires more training data. SSD is a convolutional neural network that uses a single convolutional network to anticipate bounding box positions and categorize these places in a single run. It can be trained from beginning to end. The MobileNet basic design is followed by a number of convolution layers in the SSD network.

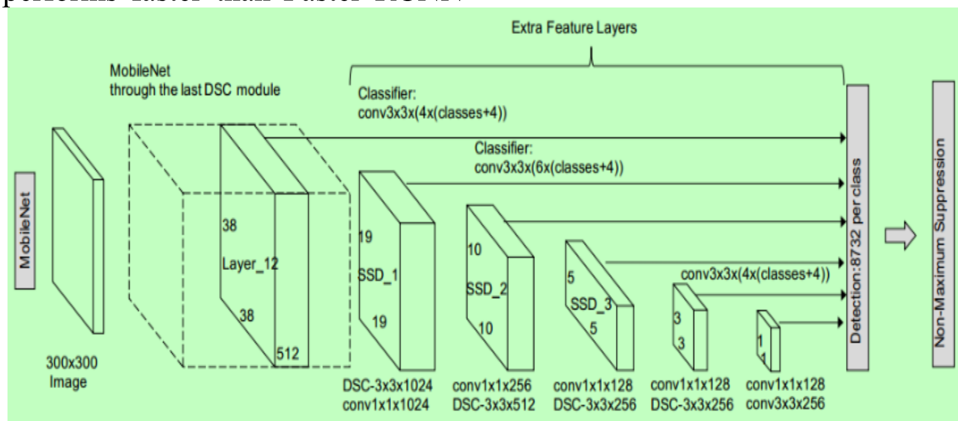


Fig. 1. Mobilenet SSD layered architecture [22]

The single-shot detection (SSD) approach compares favourably the two-shot region proposal network (RPN) methods like the R-CNN series. The SSD system requires only one shot to identify objects in an image, unlike RPN methods, which require two shots. As a result, the SSD method is considerably quicker than the RPN method [21].

For this purpose, the autors selected the MobileNet SSD algorithm because of its speed and performance. Single-shot detection was the ideal intersection of performance and resources. The MobileNet SSD algorithm also offers quicker inference than a two-shot detector and trains more quickly [22]. Therefore, the authors followed a paper by A. Younis [23] who used the MobileNet SSD Algorithm for object recognition.

B. CNN-Based Face Detector Using DLIB

Dlib is a Python library for creating practical C++ applications for conducting the data analysis and machine learning. The library was initially developed in C++ but it features strong, user-friendly Python bindings. This detector was based on linear Support Vector Machines (SVM) and a histogram of oriented gradients (HOG) [24].

The HOG-based face detector in dlib was able to recognize faces to a considerable extent even when they are not frontal. This is excellent for applications that require face detection for a large number of people [25].

We are unsure of how many of us were aware of the CNN (Convolutional Neural Network) based face detector that is present in dlib, even though the HOG+SVM-based face detector has been around for a while and has amassed a sizable user base. The researcher would like to know if this is the case because the researcher found it by accident when looking through the dlib repository on GitHub.Therefore, the face detector usage would be demonstrated to provide a part of dlib's CNN-based face detector. This would enable us to accurately identify and distinguish faces. This researcher followed a paper by S. Reddy Boyapally [26] who recognized faces with Dlib.

Dlib was selected because it was a flexible and widely used facial recognition tool kit that may strike the perfect balance between resource consumption, accuracy, and latency. The library was becoming increasingly popular in computer vision and facial recognition projects because of its flexibility in handling various challenges across different platforms.

C. Dataset

Dataset, which recognizes 91 various objects from its dataset was used in the study. The dataset is called COCO 2017, which stands for Common Objects in Context and is one of the most popular open-source object detection, segmentation, and captioning datasets. This dataset consists of 123K images. Images in COCO 2017 dataset were taken from everyday used equipments. There were 91 stuff categories, which include objects and materials with no clear boundaries like the sky, grass, street, trees, and others. Including the other 80 object categories that can be easily labelled as a person, table, tv, bottle [27]. SSD detected all the objects in a single shot.

III. Methodology

A. Experimental Setup

To minimize the initial problem of vision of visually impaired people, this study initially set up the Raspberry Pi 3 Model B, with a Raspbian operating system, in order to introduce a navigational stick as a modified approach for the blind/viusually impaired people. Before setting up Raspbian, the Raspberry Pi was connected to a laptop monitor via a 100Mbps Ethernet connection. The desktop GUI of the Raspberry Pi was connected to the laptop via VNC server software. A Raspberry Pi and laptop may connect via various tools such as VNC server software.

The desktop of Raspberry Pi can be viewed remotely by using the mouse and keyboard just as to take a live front view of the device by installing a VNC server on the desktop Pi. Furthermore, it indicated that Pi can be placed at any place in the house and still can control it. With the Pi connected to the personal laptop's WiFi through Ethernet, the person can also browse the internet.

These are the following stages for the experiment conducted in this project:

Firstly, the Raspberry Pi was configured by following the instructions in [28]. The procedure in [29] was used to configure the VNC Server to Link Raspberry Pi to a Laptop display. Moreover, TensorFlow was installed in Raspberry pi by using the method used in [30]. Now, after that, Jypyter Notebook was installed in Raspberry pi using the method used in [31]. In the same way, Open CV was installed in Raspberry Pi the method used in [32] and in this study the researcher used [33] to convert text to speech in Raspberry Pi.

B. Proposed Prototype

The proposed prototype consists of hardware equipment, software, a Mobile-Net SSD algorithm discussed earlier for object recognition as shown in Fig 2 and a dlib library for face recognition as shown in Fig 3. The combination of various hardware, software, and algorithms is combined effectively and efficiently. The hardware is programmed in Python. The Raspberry Pi 3 Model B is the primary component of the system. It is a computer that is roughly the size of a credit card and runs on an operating system (OS) that is either Linux or Windows-based. However, there is a unique structured Linux-based OS for Raspberry Pi named Raspbian. The rest of the segments are controlled by it.

The camera module was installed into Raspbian, which is used to capture images by pressing the integrated button on the prototype. Furthermore, an ultrasonic sensor was used for measuring the distance between the smart stick and the obstacle, which is shown in Fig 4. Mobile-Net Single Shot Multi-Box detection (SSD) algorithm [34], [35], [23] was used for object recognition, which recognizes 91 various objects from its dataset.

Both algorithms used the included camera. The picture caught by the camera module was sent to the Raspberry Pi, Pi processes the images by using SSD for object recognition and dlib for face recognition. Two ultrasonic sensors were used for measuring distances and detecting obstacles. A library of python was used for converting text to speech transformation named Python Text to Speech (Pyttsx).

Fig. 2 shows the working flow of object recognition, which usually gets started when blind or visually impaired individuals capture an image through the intended camera by pressing the mounted button. The object recognition model can recognize various objects. If an object gets recognized it returns an object name or else returns no object detected.

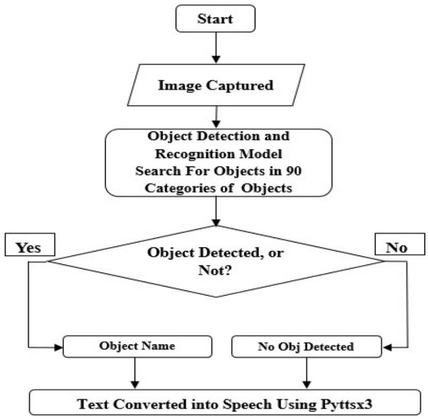


Fig. 2. Working flow of object recognition

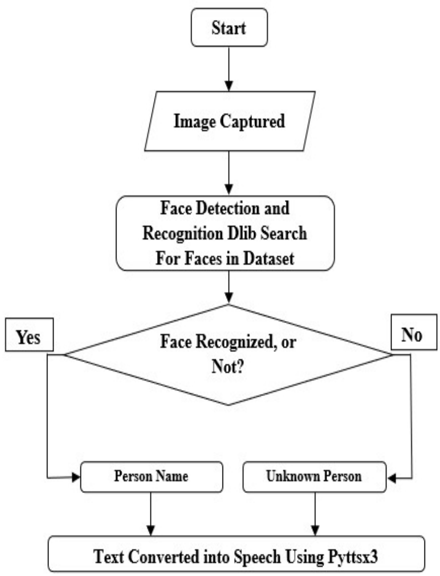


Fig. 3. Working flow of face recognition

Fig. 3 expresses the working flow of face recognition, which usually gets started when blind or visually impaired individuals capture an image through the intended camera by pressing the mounted button. Face recognition engine searches for the images in the dataset. If a face gets recognized, it returns the face name or else it returns with an indication message of unknown person.

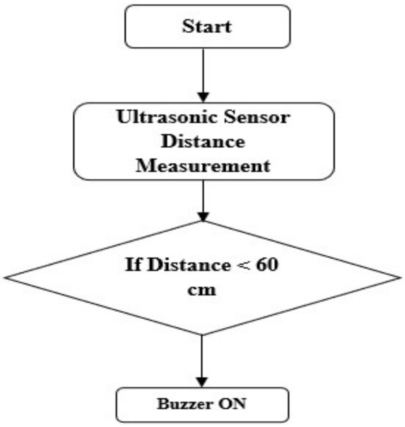


Fig. 4. Working flow of distance measurement

In Fig.4 when any impediment comes in front of the ultrasonic sensor the sound waves would replicate in the shape of an echo and generate an electric pulse. The purpose of the HC-SR04 ultrasonic sensor is used to calculate the range between the item and the ultrasonic sensor. It consists of crystal control, which transmits 40 000 Hz that travels in the air and bounces back in case an object is found. The distance is calculated with the travel time and speed of the sound. It gives splendid range detection with excessive accuracy. It calculates the distance between 2cm to 400cm. Therefore, a distance limit of less than 60 cm which is done by programming in Raspberry Pi was conducted in the current study. The buzzer activated automatically if it detects obstacles having distance of less than 60cm [36].

IV. Results and Discussion

Smart stick grants better results when evaluated for different object recognition, which is shown in Figures 5-7 and evaluated various face recognition which is shown in Figures 8-10. First, face recognition and object recognition models were evaluated individually. Then, each of their results and performances were analysed individually. Next, both models were combined and tested simultaneously and embedded into the intelligent smart blind stick with distance measurement. This smart stick was convenient and easy to use. Objects and faces can be automatically recognized in this navigational stick.The researcher tried to reduce the cost and complexity by using Raspberry Pi. The stick measured the distance with high accuracy.

Hence, this intelligent stick was recommended for blind and visually impaired people.

Fig. 5. Object recognition result 1

In this paper, 80 different categories of daily routine equipment model were used which is called mobile-net SSD. The proposed model was evaluated and compared by the object recognition results. After setting up the mobile-net SSD Object recognition model different objects were evaluated, which is shown in Fig 5. The model was tested on one of the objects (Bus) from 80 different categories of objects. The model successfully guessed 14 times out of 15 tests.

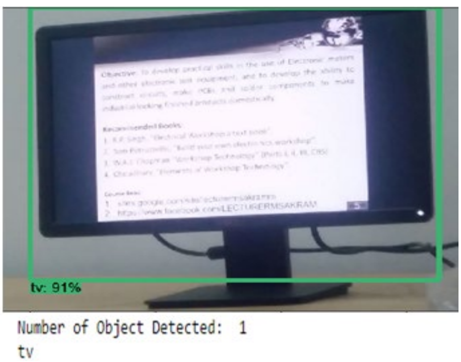


Fig. 6. Object recognition result 2

The experimental result which is shown in Fig 6. is of the TV category of objects, which is successfully guessed 13 times out of 15 tests. The response was then directed to the headphones as a speech output.

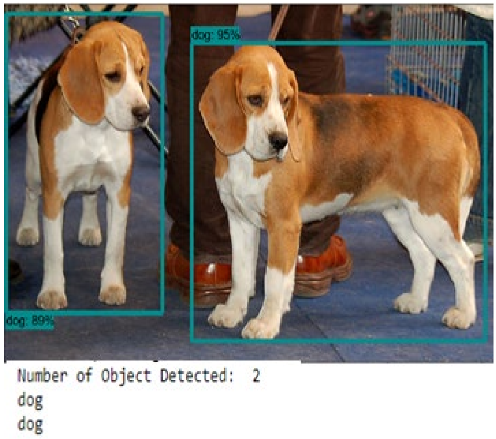


Fig. 7. Object recognition result 3

Another experimental result which is shown in Fig 7 are of dog category objects, which were tested on different instances of dog category in a single image. Finally, the model successfully guessed 14 times out of 15 tests.

Table I
Results of Object Recognition

Object	Test	Accuracy	Percision	Recall
Bus	15	93.75 %	96.77 %	93.75 %
TV	15	88.23 %	93.75 %	88.23 %
Dog	15	93.75 %	96.77 %	93.75 %
Laptop	15	88.23 %	93.75 %	88.23 %
Chair	15	93.75 %	96.77 %	93.75 %
Cat	15	88.23 %	93.75 %	88.23 %
Knife	15	93.75 %	96.77 %	93.75 %
Apple	15	93.75 %	96.77 %	93.75 %

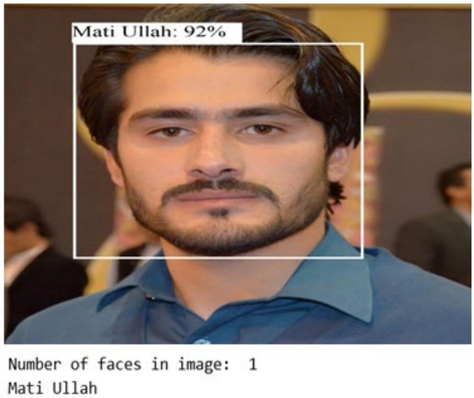


Fig. 8. Face recognition result 1

After this dlib was applied as the facial recognition engine on the image which is shown in Fig.8, which is successfully recognized 18 times out of 20 tests.

Fig. 9. Face recognition result 2

To further evaluate dlib as the facial recognition engine, it was tested on Fig 9, which gave us a better result. Furthermore, it was tested for 20 times and the results were positive 19 times.

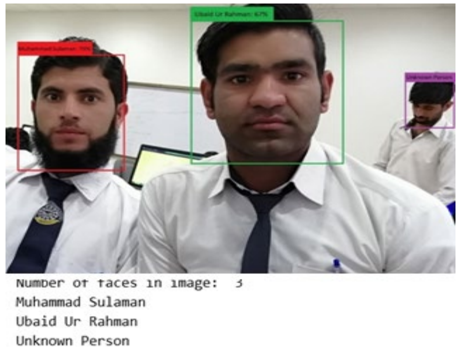


Fig. 10. Face recognition result 3

While, evaluating dlib as a facial recognition engine, better results were obtained with an image having a single face. Then, recognition engine was tested on an image having multiple faces, which gave better results as shown in Fig.10

Table II
Results of Face Recognition

Face	Test	Accuracy	Percision	Recall
Matiullah	20	92.68 %	94.73 %	90 %
Ishaq	20	95.12 %	95%	95 %
Sulaman	20	90.24 %	94.44 %	85 %
Imran	20	92.68 %	94.73 %	90 %
Zubair	20	95.12 %	95 %	95 %
Ubvaid	20	90.24 %	94.44 %	85 %

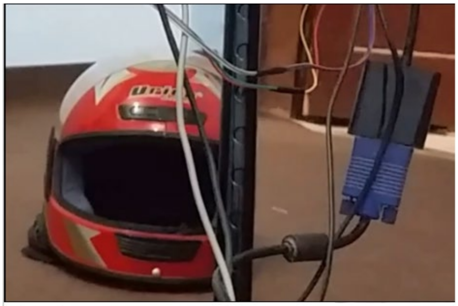
In machine learning, precision, accuracy, and recall are commonly used metrics to assess any model performance. In this study, Tables 1 and 2 presented results obtained using these performance metrics.

Precision: Precision measured the proportion of true positives among all predicted positives. In other words, it measured the model's ability to correctly identify the positive samples. The formula for precision is: Precision = True Positives / (True Positives + False Positives)

Accuracy: Accuracy measured the proportion of correctly classified samples (both true positives and true negatives) among all samples. The formula for accuracy is: Accuracy = (True Positives + True Negatives) / (True Positives + False Positives + True Negatives + False Negatives)

Recall: Recall measured the proportion of true positives among all actual positives. In other words, it measured the model's ability to correctly identify all positive samples. The formula for recall is: Recall = True Positives / (True Positives + False Negatives)

It's important to note that the choice of metric to focus, would depend on the problem being solved. For example, in a medical diagnosis task, a recall may be more important than precision, as it's more important to correctly identify all positive cases, even if some false positives are included. Conversely, in a fraud detection task, precision may be more important than recall, as it's more important to avoid false positives and correctly identify all negative cases.



Helmet
('Distance: ' 53.0 'cm'

Fig. 11. Distance measurement result 1

The image above displayed in Figure 11. is a helmet captured by a camera, along with the corresponding output from an ultrasonic sensor. This output is then transmitted to the headphones as a speech output.

Laptop
('Distance: ' 53.0 'cm'

Fig. 12. Distance measurement result 2

The image above displayed in Figure 12 is a laptop captured by a camera, along with the corresponding output from an ultrasonic sensor. This output is then transmitted to the headphones as a speech output.

V. Conclusion

This study’s goal was to reduce the anxiety experienced by blind or visually impaired individuals when they are in a potentially unsafe environment ecounter any objects in a crowdy environment. Therefore, this study proposed an innovative navigational stick that would offer assistance to the blind or visually disabled community. The smart stick prototype incorporates many intelligent features that would make it the best choice forvisually impaired people.

The designed smart navigational stick is a precision-made intelligent walking stick, which is designed to enable blind individuals to navigate from one location to another without anyone’s assistance. With this intelligent stick, they would be able to walk into an environment that would give them directions to the placeswhich they require. It's also a useful mobility aid that helps and guides the users by detecting and recognizing humans and objects at once. This navigational device can provide information to blind and visually impaired people when they are alerted by any fast-moving object, which would get detected at or less than 60 centimetres. The tool is effective and unique in its capacity to identify and recognise people and items that blind people may come into contact with. It is easy to use, making it accessible to a wide range of users.

A. Future Implications

In future, the proposed prototype in this research work might require certain modification in the methodology by adding the new version of Raspberry Pi 4, which is currently available in the market. Secondly, in the future, Raspberry Pi can be replaced by using the NVIDIA Jeston Nano developer kit. It is a compact, powerful computer that enables the parallel operation of many neural networks, including those for speech, face and object recognition, and picture classification. In future, the researcher is intended to add air quality monitoring and reporting features to facilitate the blind person using several approaches processed in [37], [38]. Security can also be added to the smart stack by using the method discussed in [39].

References

[1] World Health Organization, “Blindness and vision impairment.” World Health Organization. https://www.who.int/news-

room/fact-sheets/detail/blindness-and-visual-impairment (accessed Dec. 12, 2022).

[2] B. Hassan, R. Ahmed, B. Li, A. Noor, and Z. ul Hassan, "A comprehensive study capturing vision loss burden in Pakistan (1990-2025): Findings from the Global Burden of Disease (GBD) 2017 study," *PLOS ONE*, vol. 14, no. 5, pp. 1–19, May 2019, doi: https://doi.org/10.1371/journal.pone.0216492

[3] The World Bank, "Population, total – Pakistan Data", World Bank. Available: https://data.worldbank.org/indicator/SP.POP.TOTL?locations=PK\&name-desc=true (accessed Dec. 12, 2022).

[4] V. Kunta, C. Tuniki, and U. Sairam, "Multi-Functional blind stick for visually impaired people," in *5th Int. Conf. Commun. Electron. Syst.*, June 10–12, 2020, pp. 895–899, doi : https://doi.org/10.1109/ICCES48766.2020.9137870

[5] A. Elsonbaty, "Smart blind stick design and implementation," *Int. J. Eng. Adv. Technol.*, vol. 10, pp. 17–20, June 2021, doi: https://doi.org/10.35940/ijeat.D2535.0610521

[6] S. Grover, A. Hassan, K. Yashaswi, and N. Shinde, "Smart blind stick," *Int. J. Electro. Commun. Eng.*, vol. 7, pp. 19–23, May 2020, doi: httpa://doi.org/10.14445/23488549/IJECE-V7I5P104

[7] H. Q. Nguyen, A. H. L. Duong, M. D. Vu, T. Q. Dinh, and H. T. Ngo, "Smart blind stick for visually impaired people," in *8th Int. Conf. Develop. Biomed. Eng. Vietnam*, Cham, 2022, pp. 145–165. doi: https://doi.org/10.1007/978-3-030-75506-5_12

[8] M. P. Agrawal and A. R. Gupta, "Smart Stick for the blind and Visually Impaired People", in *2nd Int. Conf. Inven. Commun. Comput. Technol.*, pp. 542–545, 2018.

[9] R. F. Olanrewaju, M. L. A. M. Radzi, and M. Rehab, "iWalk: Intelligent walking stick for visually impaired subjects," in *IEEE 4th Int. Conf. Smart Instru. Measur. Appli.*, pp. 1–4, 2017.

[10] K. Manikanta, T. Phani, and A. Pravin, "Implementation and design of smart blind stick for obstacle detection and navigation system", *Int. J. Eng. Sci. Comput.*, vol. 8, no. 8, pp. 18785–18790, 2018.

[11] S. Halim, F. Handafiah, R. Aprilliyani, and A. Udhiarto, “Electronic white cane with GPS radar-based concept as blind mobility enhancement without distance limitation,” *AIP Conf. Proc.*, vol. 1933, pp. 040024-1–040024-7, Feb. 2018, doi: https://doi.org/10.1063/1.5023994

[12] N. R. P. J. Johnson, "Smart walking stick for blind," *Int. J. Eng. Sci. Inv. Res. Develop.*, vol. 3, no. 4, 2017.

[13] M. Bansal, S. Malik, M. Kumar, and N. Meena, “Arduino based smart walking cane for visually impaired people,” in *4th Int. Conf. Inv. Syst. Cont.*, Jan. 2020, pp. 462–465. doi: https://doi.org/10.1109/ICISC47916.2020.9171209

[14] A. S. Manaf, E. Joseph, S. P. M, and A. Ahmed, “Effective fast response smart stick for blind people,” *Int. J. Eng. Res. Technol.*, vol. 7, no. 8, June 2019, doi: https://doi.org/10.17577/IJERTCONV7IS08057

[15] I. A. Satam, M. N. Al-Hamadani, and A. H. Ahmed. (2019). Design and implement a smart blind stick. *J. Adv. Res. Dynam Control Syst.*, vol. 11, no. 8, pp. 42–47.

[16] L. Chen, J. Su, M. Chen, W. Chang, C. Yang, and C. Sie, "An implementation of an intelligent assistance system for visually impaired/blind people," in *2019 IEEE Int. Conf. Consum. Elec.*, Las Vegas, NV, USA, pp. 1–2, 2019.

[17] T. Nadu, “Arduino based walking stick for visually impaired,” *Int. Res. J. Eng. Technol.*, vol. 5 no.3, Mar. 2018

[18] R. Dhanuja, F. Farhana, and G. Savitha, “Smart blind stick using Arduino,” vol. 5, no. 3, pp. 2553–2555, Mar. 2018.

[19] N. Dey, A. Paul, P. Ghosh, C. Mukherjee, R. De, and S. Dey, ”Ultrasonic sensor based smart blind stick,” in *Int. Conf. Curr. Trend. Converg. Technol.*, Coimbatore, 2018, pp. 1–4, https://doi.org/10.1109/ICCTCT.2018.8551067

[20] M. Shanmugam, J. Victor, M. Gupta and K. Saravanakumar, ”Smart stick for blind people,” in *National Conf. Emerg. Trends Info. Technol.*, Christ University, Bengaluru, Mar., 2017.

[21] M. Singhalais, “Object detection using SSD mobilenetV2 using tensorflow API : Can detect any single

class from coco datas," *Medium*. https://medium.com/@techmayank2000/object-detection-using-ssd-mobilenetv2-using-tensorflow-api-can-detect-any-single-class-from-31a31bbd0691(accessed Dec. 18, 2022).

[22] A. Kumar, Z. J. Zhang, and H. Lyu, "Object detection in real time based on improved single shot multi-box detector algorithm," *J. Wireless Commun. Network.*, vol. 2020, no. 1, Art. no. 204, Oct. 2020, doi: https://doi.org/10.1186/s13638-020-01826-x

[23] A. Younis, L. Shixin, S. Jn, and Z. Hai, "Real-Time object detection using pre-trained deep learning models MobileNet-SSD. in *Proc. 6th Int. Conf. Comput. Data Eng.*, vol. 978-1-4503-7673–0, pp. 44–48, Mar. 2020.

[24] R. Kavitha, P. Subha, R. Srinivasan, and M. Kavitha, "Implementing opencv and dlib open-source library for detection of driver's fatigue," in *Innov. Data Commun. Technol. Appl.*, Singapore, 2022, pp. 353–367. doi: https://doi.org/10.1007/978-981-16-7167-8_26.

[25] A. Ponnusamy, "CNN based face detector from dlib," *Medium*. https://towardsdatascience.com/cnn-based-face-detector-from-dlib-c3696195e01c (accessed Dec. 19, 2022).

[26] S. R. Boyapally, "Facial recognition and attendance system using dlib and face_recognition libraries," https://papers.ssrn.com/abstract=3804334 (accessed Dec. 27, 2022).

[27] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in *Computer Vision – ECCV 2014*. Cham, 2014, pp. 740–755.

[28] A. Piltch, "How to set up a raspberry pi for the first time," *Tom's Hardware*. https://www.tomshardware.com/how-to/set-up-raspberry-pi (accessed Feb. 27, 2023).

[29] V. Gr, "How to use raspberry pi2 with a laptop display using VNC server," *Instructables*. https://www.instructables.com/How-to-Use-Raspberry-Pi2-With-a-Laptop-Display-Usi/ (accessed Feb. 27, 2023).

[30] Q-engineering, "Install tensorflow 2.1.0 on raspberry Pi 4 - Q-engineering," Q-engineering. https://qengineering.eu/install-tensorflow-2.1.0-on-raspberry-

pi-4.html (accessed Feb. 27, 2023).

[31] S. Leet, “Jupyter notebook on raspberry Pi,” *Instructables*. https://www.instructables.com/Jupyter-Notebook-on-Raspberry-Pi/ (accessed Feb. 27, 2023).

[32] A. Singh and MACFOS, “Installing opencv using cmake in raspberry Pi,” *Robu.* https://robu.in/installing-opencv-using-cmake-in-raspberry-pi/ (accessed Feb. 27, 2023).

[33] V. Phutak, R. Kamble, S. Gore, M. Alave, and R. R. Kulkarni, “Text to speech conversion using raspberry - PI,” vol. 4, no. 2, pp. 91–293, Feb. 2019.

[34] Y. Li, H. Huang, Q. Xie, L. Yao, and Q. Chen, “Research on a surface defect detection algorithm based on MobileNet-SSD,” *Appl. Sci.*, vol. 8, no. 9, Art. no. 9, Sep. 2018, doi: https://doi.org/10.3390/app8091678

[35] H. Ali, M. Khursheed, S. K. Fatima, S. M. Shuja, and S. Noor, "Object recognition for dental instruments using SSD-MobileNet," in *Int. Conf. Info. Sci. Commun. Technol.*, 2019, pp. 1–6, doi: https://doi.org/10.1109/CISCT.2019.8777441

[36] S. Hossain, “Smart blind stick using ultrasound distance measurement sensor system,” Bechelor thesis, Dep. Elec. Elec. Eng., Daffodil Int. Univ., Dahak, Bangladesh, 2020.

[37] M. Aamir et al., “Spatiotemporal change of air-quality patterns in hubei province—a pre-to Post-Covid-19 analysis using path analysis and regression,” *Atmosphere*, vol. 12, no. 10, Art. no. 1338, 2021, doi : https://doi.org/10.3390/atmos12101338

[38] U. A. Bhatti, Z. Zeeshan, M. M. Nizamani, S. Bazai, Z. Yu, and L. Yuan, “Assessing the change of ambient air quality patterns in Jiangsu Province of China pre-to post-COVID-19,” *Chemosphere*, vol. 288, Art. no. 132569, Feb. 2022, doi : https://doi.org/10.1016/j.chemosphere.2021.132569

[39] R. U., Bazai, S. Ullah, U. A. Bhatti, S. A. A. Shah, and H. Ahmad, “Utilizing blockchain technology to enhance smart home secu rity and privacy,” in *Int. Conf. Info. Technol. Appl.*, pp. 76–86, Singapore, 2022.

Annexture

1. Object Recognition

```
# coding: utf-8
# # Object Detection
# # Imports
# In[1]:
import speech_recognition as sr
import numpy as np
import os
import six.moves.urllib as urllib
import sys
import tarfile
import tensorflow as tf
import zipfile
from distutils.version import StrictVersion
from collections import defaultdict
from io import StringIO
from matplotlib import pyplot as plt
from PIL import Image
# This is needed since the notebook is stored in the object_detection folder.
sys.path.append("..")
    from object_detection.utils import ops as utils_ops
# Env setup
# In[2]:
# This is needed to display the images.
# get_ipython().magic(u'matplotlib inline')
    from utils import label_map_util
    from utils import visualization_utils as vis_util
# This is needed since the notebook is stored in the object_detection folder.
# # Model preparation
# ## Variables
#
# Any model exported using the `export_inference_graph.py` tool can be loaded here simply by changing `PATH_TO_CKPT` to point to a new .pb file.
#
# By default we use an "SSD with Mobilenet" model here. See the [detection model zoo](https://github.com/tensorflow/models/blob/master/object_detection/g3doc/detection_model_zoo.md) for a list of other models that can be run out-of-the-box with varying speeds and accuracies.
# In[4]:
# What model to download.
MODEL_NAME = 'ssdlite_mobilenet_v2_coco_2018_05_09'
MODEL_FILE = MODEL_NAME + '.tar.gz'
DOWNLOAD_BASE = 'http://download.tensorflow.org/models/object_detection/'
# Path to frozen detection graph. This is the actual model that is used for the object detection.
PATH_TO_FROZEN_GRAPH = MODEL_NAME + '/frozen_inference_graph.pb'
```

```
# List of the strings that is used to add correct label for each box.
PATH_TO_LABELS = os.path.join('data', 'mscoco_label_map.pbtxt')
NUM_CLASSES = 90
# ## Download Model
## In[5]:
#print( " Downloading model ")
#
#opener = urllib.request.URLopener()
#opener.retrieve(DOWNLOAD_BASE + MODEL_FILE, MODEL_FILE)
#tar_file = tarfile.open(MODEL_FILE)
#for file in tar_file.getmembers():
#   file_name = os.path.basename(file.name)
#   if 'frozen_inference_graph.pb' in file_name:
#    tar_file.extract(file, os.getcwd())
#
#print (" Loading frozen model into memory")
## ## Load a (frozen) Tensorflow model into memory.
# In[6]:
  detection_graph = tf.Graph()
  with detection_graph.as_default():
  od_graph_def = tf.GraphDef()
  with tf.gfile.GFile(PATH_TO_FROZEN_GRAPH, 'rb') as fid:
   serialized_graph = fid.read()

od_graph_def.ParseFromString(serialized_graph)

tf.import_graph_def(od_graph_def, name='')
# ## Loading label map
# Label maps map indices to category names, so that when our convolution network predicts `5`, we know that this corresponds to `airplane`.  Here we use internal utility functions, but anything that returns a dictionary mapping integers to appropriate string labels would be fine
# In[7]:
category_index = label_map_util.create_category_index_from_labelmap(PATH_TO_LABELS, use_display_name=True)
def load_image_into_numpy_array(image):
  (im_width, im_height) = image.size
  return np.array(image.getdata()).reshape(
     (im_height, im_width, 3)).astype(np.uint8)
label_map = label_map_util.load_labelmap(PATH_TO_LABELS)
categories = label_map_util.convert_label_map_to_categories(label_map, max_num_classes=NUM_CLASSES, use_display_name=True)
```

```
category_index =
label_map_util.create_category_index(categories)
# For the sake of simplicity we will use only 2 images:
# image1.jpg
# image2.jpg
# If you want to test the code with your images, just add path to the images to the TEST_IMAGE_PATHS.
PATH_TO_TEST_IMAGES_DIR = 'test_images'
TEST_IMAGE_PATHS =
[ os.path.join(PATH_TO_TEST_IMAGES_DIR,
'image{}.jpg'.format(i)) for i in range(1, 2) ]
# Size, in inches, of the output images.
IMAGE_SIZE = (12, 8)
# In[11]:
    def
run_inference_for_single_image(image, graph):
    with graph.as_default():
     with tf.Session() as sess:
     # Get handles to input and output tensors
     ops =
tf.get_default_graph().get_operations()
     all_tensor_names =
{output.name for op in ops for output in op.outputs}
     tensor_dict = {}
     for key in [
        'num_detections',
'detection_boxes', 'detection_scores',
        'detection_classes',
'detection_masks'
      ]:
       tensor_name = key + ':0'
       if tensor_name in
all_tensor_names:
       tensor_dict[key] =
tf.get_default_graph().get_tensor_by_name(
        tensor_name)
       if 'detection_masks' in
tensor_dict:
       # The following processing is only for single image
       detection_boxes =
tf.squeeze(tensor_dict['detection_boxes'], [0])
       detection_masks =
tf.squeeze(tensor_dict['detection_masks'], [0])
       # Reframe is required to translate mask from box coordinates to image coordinates and fit the image size.
       real_num_detection =
tf.cast(tensor_dict['num_detections'][0], tf.int32)
       detection_boxes =
tf.slice(detection_boxes, [0, 0],
[real_num_detection, -1])
       detection_masks =
tf.slice(detection_masks, [0, 0, 0],
[real_num_detection, -1, -1])
       detection_masks_reframed =
utils_ops.reframe_box_masks_to_image_masks(
```

detection_masks, detection_boxes, image.shape[0], image.shape[1])
detection_masks_reframed = tf.cast(
tf.greater(detection_masks_reframed, 0.5), tf.uint8)
# Follow the convention by adding back the batch dimension
tensor_dict['detection_masks'] = tf.expand_dims(
detection_masks_reframed, 0)
image_tensor = tf.get_default_graph().get_tensor_by_name('image_tensor:0')
# Run inference
output_dict = sess.run(tensor_dict,
feed_dict={image_tensor: np.expand_dims(image, 0)})
# all outputs are float32 numpy arrays, so convert types as appropriate
output_dict['num_detections'] = int(output_dict['num_detections'][0])
output_dict['detection_classes'] = output_dict[
'detection_classes'][0].astype(np.uint8)
output_dict['detection_boxes'] = output_dict['detection_boxes'][0]
output_dict['detection_scores'] = output_dict['detection_scores'][0]
if 'detection_masks' in output_dict:
output_dict['detection_masks'] = output_dict['detection_masks'][0]
return output_dict
for image_path in TEST_IMAGE_PATHS:
image = Image.open(image_path)
# the array based representation of the image will be used later in order to prepare the
# result image with boxes and labels on it.
image_np = load_image_into_numpy_array(image)
# Expand dimensions since the model expects images to have shape: [1, None, None, 3]
image_np_expanded = np.expand_dims(image_np, axis=0)
# Actual detection.
output_dict = run_inference_for_single_image(image_np, detection_graph)
# Visualization of the results of a detection.
vis_util.visualize_boxes_and_labels_on_image_array(
image_np,
output_dict['detection_boxes'],
output_dict['detection_classes'],
output_dict['detection_scores'],
category_index,

```
instance_masks=output_dict.get('de
tection_masks'),

use_normalized_coordinates=True,
     line_thickness=5)

plt.figure(figsize=IMAGE_SIZE)
     plt.imshow(image_np)
```

2. Face Recognition

```
import face_recognition
import numpy as np
from PIL import Image, ImageDra
w
from IPython.display import displa
y
# This is an example of running fac
e recognition on a single image
# and drawing a box around each p
erson that was identified.
# Load a sample picture and learn h
ow to recognize it.
matiullah_image = face_recognitio
n.load_image_file("matiullah.img")
matiullah_face_encoding = face_re
cognition.face_encodings(matiullah
_image )[0]
# Load a second sample picture and
 learn how to recognize it.
Ishaq_ahmed = face_recognition.lo
ad_image_file("Ishaq ahmad")
ishaqahmed_face_encoding = face_
recognition.face_encodings(Ishaq_
ahmed )[0]
# Create arrays of known face enco
dings and their names
known_face_encodings = [
   matiullah_face_encoding ,
   ishaqahmed_face_encoding
]
known_face_names = [
   "Mati Ullah",
   "Ishaq Ahmad"
]
        print('Learned encoding for',
len(known_face_encodings), 'imag
es.')
# Load an image with an unknown
face
        unknown_image = face_reco
gnition.load_image_file("multiplef
acesimage.jpg")
# Find all the faces and face encodi
ngs in the unknown image
        face_locations = face_recogn
ition.face_locations(unknown_ima
ge)
        face_encodings = face_recog
nition.face_encodings(unknown_i
mage, face_locations)
# Convert the image to a PIL
format image so that we can draw o
n top of it with the Pillow library
# See http://pillow.readthedocs.io/ f
or more about PIL/Pillow
        pil_image = Image.fromarray
(unknown_image)
# Create a Pillow ImageDraw Draw
 instance to draw with
        draw = ImageDraw.Draw(pil
_image)
# Loop through each face found in t
he unknown image
        for (top, right, bottom, left), f
ace_encoding in zip(face_locations,
 face_encodings):
```

```
    # See if the face is a match for th
e known face(s)
            matches = face_recognition.
compare_faces(known_face_encodi
ngs, face_encoding)
             name = "Unknown"
    # Or instead, use the known face
with the smallest distance to the ne
w face
            face_distances = face_recog
nition.face_distance(known_face_e
ncodings, face_encoding)
             best_match_index = np.arg
min(face_distances)
            if matches[best_match_inde
x]:
             name = known_face_names
[best_match_index]
    # Draw a box around the face usi
ng the Pillow module
              draw.rectangle(((left, top),
(right, bottom)), outline=(0, 0, 255))
    # Draw a label with a name belo
w the face
             text_width, text_height = dr
aw.textsize(name)
            draw.rectangle(((left, botto
m - text_height -
            10), (right, bottom)), fill=(0,
 0, 255), outline=(0, 0, 255))
            draw.text((left + 6, bottom -
 text_height -
 5), name, fill=(255, 255, 255, 255))
# Remove the drawing library from
 memory as per the Pillow docs
del draw
# Display the resulting image
          display(pil_image)
```

3. Distance Measurement

```
import RPi.GPIO as GPIO
import time
import signal
import sys
# use Raspberry Pi board pin
numbers
GPIO.setmode(GPIO.BCM)
# set GPIO Pins
pinTrigger = 18
pinEcho = 24
def close(signal, frame):
            print("\nTurning off
ultrasonic distance detection...\n")
            GPIO.cleanup()
            sys.exit(0)
signal.signal(signal.SIGINT, close)
# set GPIO input and output
channels
GPIO.setup(pinTrigger,
GPIO.OUT)
GPIO.setup(pinEcho, GPIO.IN)
while True:
            # set Trigger to HIGH
            GPIO.output(pinTrigger,
True)
            # set Trigger after 0.01ms to
LOW
            time.sleep(0.00001)
            GPIO.output(pinTrigger,
False)
            startTime = time.time()
            stopTime = time.time()
            # save start time
            while 0 ==
GPIO.input(pinEcho):
                        startTime =
time.time()
```

```
        # save time of arrival
        while 1 ==
GPIO.input(pinEcho):
                stopTime =
time.time()
        # time difference between
start and arrival
        TimeElapsed = stopTime -
startTime
        # multiply with the sonic
speed (34300 cm/s)
        # and divide by 2, because
there and back
        distance = (TimeElapsed *
34300) / 2
        print ("Distance: %.1f
cm" % distance)
        time.sleep(1)
```