Chapter 1

Introduction

1.1 Description

Visual impairment is defined as the limitation of actions and functions of the visual system. The National Eye Institute defines low vision as a visual impairment not correctable by standard glasses, contact lenses, medication or surgery that interferes with the ability to perform activities of daily living.

The number of different vision conditions that can affect a person's eyesight are varied in the others may have a larger affect. Various conditions require only eyeglasses or contact lenses in order to correct the person's vision. Other conditions may require surgery. Additional health concerns can affect a person's vision as well, such as Diabetes or Glaucoma.

Low Vision is sometimes also referred to as, 'Vision Loss,' means that even though a person may use eyeglasses, contact lenses, medication, or surgical techniques to improve their vision; they still have difficulty seeing. 'Diabetic Retinopathy,' is a condition in which Diabetes has damaged tiny blood vessels inside the person's retina, causing low vision.

'Age-Related Macular Degeneration,' is a condition in which the cells in a person's retina that allow them to see fine details have died. 'Glaucoma,' is a condition in which the fluid pressure in a person's eyes slowly rises, damaging their optic nerve.

According to the International Classification of Diseases There are 4 levels of visual function:

Blindness

Normal vision

Severe visual impairment

Moderate visual impairment

Moderate visual impairment combined with severe visual impairment are grouped under the term low vision: low vision taken together with blindness represents all visual impairment.
There are some different terms used to describe levels of vision disability. These terms include, 'Partially-Sighted,' 'Low-Vision,' 'Legally Blind,' and, 'Totally Blind.' Partially-Sighted means the person has some form of visual disability that may require special education. Low-Vision usually is used to refer to persons who experience a more severe loss of vision that is not necessarily limited to distance vision. Persons with low-vision may be unable to read a newspaper at an average distance with eyeglasses or contacts, and may need large print or Braille. Persons who are legally blind have less than 20/200 vision in their better eye, or a very limited field of vision, often 20 degrees at its widest point. Persons who are totally blind are unable to see and often use Braille or other non-visual forms of media. Eye disorders lead to vision loss; visual impairment is a consequence of a functional loss of vision rather than the eye disorder itself. Retinal degeneration, muscular problems, albinism, corneal disorders, congenital disorders, and infections can also lead to vision impairment.

1.2 Problem Formulation

As we have got into the brief exposure of what is visual imparity, the question arises what problems are faced by them in general. They are generally termed to be dependent on others and always are being seen helpless which the absolute opposite of what they want. In order to assist them there is a lot of technology that will aid them to make them independent but the end question that springs back is that if it is actually making them independent and also enabling them free movement. Whatever techs exist in the market is making them trade off their freedom of movement with the achievement of normal human like sight. This trade off may be by holding a stick or carrying a backpack which is heavy. That may be much more frustrating for them than a normal stick and asking for help rather being all bulked up.

1.3 Motivation

BrainPort uses a camera to capture visual data. The optical information -- light that would normally hit the retina that the camera picks up is in digital form, and it uses radio signals to send the ones and zeroes to the CPU for encoding. Each set of pixels in the camera's light sensor corresponds to an electrode in the array. The CPU runs a program that turns the camera's electrical information into a spatially encoded signal. The encoded signal represents differences in pixel data as differences in pulse characteristics such as frequency, amplitude and duration. Multidimensional image information takes the form of variances in pulse current or voltage, pulse duration, intervals between pulses and the number of pulses in a burst, among other parameters. But the problem with Brainport was at any given moment you need to have the simulation circuitry in your tongue to transfer the information to the sensory parts. Inspired by this problem and also motivated with the need of enabling free movement and a low cost solution we have come up with Oculus.

1.4 Proposed Solution

The architecture is primarily based on Raspberry pi 3 model B. The system would use a raspberry pi 3 model B. The power source to the pi is a portable rechargeable power bank. To enable capturing of the image a Web-camera of good quality is being used along with the raspberry pi. The ultrasonic sensors are being connected to the pi via breadboard and not directly to prevent overvoltage to the sensor.

Fig. 1.1 Components used

The Object detection is done along with the combination of Mobile Net + Single Shot Detector algorithm used in Open CV approach. The voice output is obtained by using espeak enabling the TTS conversion. All of this works on the raspberry pi model B which runs on Raspbian OS (Stretch). The entire setup is being mounted on the cap. The reason for mounting the setup on the cap is because it gives and elevated vision to the camera and the sensors in order to capture the image and do the object detection and give the speech output. The speech output is not given directly. The speech output is given in terms of audio through the 3.5mm port on the raspberry pi itself. In order to identify objects the Caffe Framework is being used.

It is a pre-trained model on the COCO dataset that has a number of classes among which the object detected may belong. The set steps followed by OCULUS are as follows:

1. Capture the image through the Web-Camera.
2. Store the image in the memory and give it for image classification by SSD+Mobile_Net.
3. The text file is being generated that contains the information of the objects in the image and accordingly.
4. The text file is then given to e-speak that enables speech output to the visually impaired person.
5. In case the person reaches too close to the object the ultrasonic sensors will give a default speech of "You are too close to the object"

Fig 1.2: Proposed solution

The algorithm and the framework being used are as follows:

A. Single Shot Detection (SSD)

SSD is simple relative to methods that require object proposals because it completely eliminates proposal generation and subsequent pixel or feature resampling stages and encapsulates all computation in a single network. This makes SSD easy to train and straightforward to integrate into systems that require a detection component. The core of SSD is predicting category scores and box offsets for a fixed set of default bounding boxes using small convolutional filters applied to feature maps. To achieve high detection accuracy we produce predictions of different scales from feature maps of different scales, and explicitly separate predictions by aspect ratio. [2]

B.MobileNets

The Mobile Net structure is built on depth wise separable convolutions except for the first layer which is a full convolution. By defining the network in such simple terms we are able to easily explore network topologies to find a good network. All layers are followed by a batch norm and ReLU nonlinearity with the exception of the final fully connected layer which has no nonlinearity and feeds into a softmax layer for classification. [1]

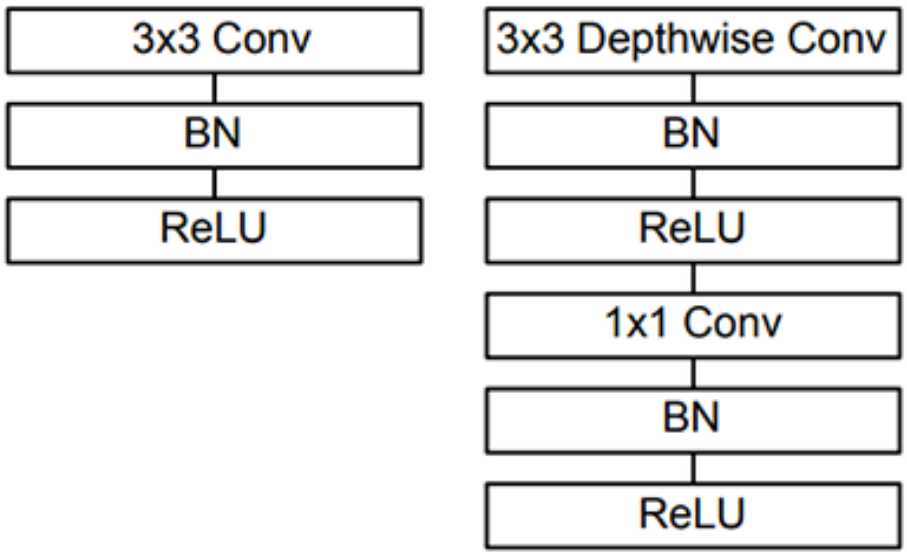

Fig 1.3. (Left) Standard convolutional layer with batch normalization and ReLU. (Right) Depthwise separable convolution with depthwise and pointwise layers followed by batch normalization and ReLU (figure and caption from Liu et al.).

C. CAFFE Framework

The reason Caffe framework is being chosen is because it is expressive architecture encourages application and innovation. Models and optimization are defined by configuration without hard-coding. Switch between CPU and GPU by setting a single flag to train on a GPU machine then deploy to commodity clusters or mobile devices. It gives a classification for 20 types which can be accordingly used to map the detected object to the desired classes.

D. Combining Mobile Net and SSD

If we combine both the MobileNet architecture and the Single Shot Detector (SSD) framework, we arrive at a fast, efficient deep learning-based method to object detection.

E.Espeak

eSpeak is an open source software text-to-speech synthesizer that is available in many languages, including Indonesian. In synthesizing speech, eSpeak uses Formant synthesis method. This allows many languages to be provided in a small size. The produced speech is clear and can be used at high speeds. A simple method to get some feedback from the Raspberry Pi is to use Text To Speech (TTS). Espeak is one of the way to do so. By using Espeak the file in which the output text is being stored is given as the input which gives the audio as output.

1.5 Scope of the Project

The project is a combination of both hardware and software. The raspberry pi model B requires a power supply which will be provided by an external battery which will be rechargeable. Also, the camera module is connected via a web cam. The ultrasonic sensors are also connected to the raspberry pi via a small bread board. Since we are trying to implement the hardware on a cap it is necessary for us to keep it light weight so that the person wearing it doesn't feel the burden of carrying any such a bulky object on the head and also provides adequate speech guides to user. Once after the image is classified, we delete the image to optimize memory utilization. Moreover, in our project we have specified 20 objects for classification. So, it will try to fit into these objects of the image captured. Also, the GPS module is not included in our primarily as it reduces the cost of the cap.

Chapter 2

Review of Literature

"Object Detection on Raspberry Pi" This paper enables us to create a bot that accepts an image or video from the camera as a source. The camera is made a 360degree camera by mounting it on a stepper engine. The image/video is being analyzed by using open CV and the output is given in the form of audio. The goal is to construct a model that can recognize the protest of indicated shading that make utilization of open source equipment and that chips away at the premise of visual information caught from an ordinary webcam which has a reasonable lucidity. Having a picture preparing calculation which distinguishes a protest first and after that tracks it the length of it is in the observable pathway of the camera. As the protest moves, the PC/portable PC/implanted Board offers flag to engine to turn the camera which is mounted on a stepper engine. We executed the proposed calculation on Raspberry Pi board utilizing OpenCV on Linux foundation. [5]

"Object Detection and Tracking using Image Processing" This paper mainly focuses on the basis to implement the object detection and tracking based on its color, which is a visual based project i.e., the input to the project will be the video/image data which is continuously captured with the help of a webcam which is interfaced to the Raspberry Pi. The visual data captured by the webcam is processed in the Raspberry Pi and the object is detected based on the color or shape and if the object is detected, the servo motor is rotated in such a way that wherever the object moves, the camera will be pointing to that object. It will detect the object and it tracks that object by moving the camera in the direction of the detected object. The visual data captured by the webcam is processed in the Raspberry Pi and the object is detected based on the color or shape and if the object is detected, the servo motor is rotated in such a way that wherever the object moves the camera will be pointing to that object. Here, the servos are controlled by the help of a Microcontroller board called Arduino board through its PWM pins. They can control the angle of servo rotations by the Arduino board i.e., by varying the pulse widths. [6]

"Image Processing Target tracking Robot using Raspberry pi" This paper focuses on how the ultrasonic sensors can be used to detect the distance by the processing done on raspberry pi.
This Project mainly focuses on the basis to implement the object detection and tracking based on its color and shape. It is a real time visual based project that means input of the project is images or video which is continuously captured with the help of pi camera located on front side of the robot chassis and connected to Raspberry pi with the help of RMC connector. By using this method, we can easily detect and track any object which may be Ball or Book or any instrument, Tools. This project can be used for surveillance purpose in many security applications. [7]

"Camera based Text to Speech Conversion, Obstacle and Currency Detection for Blind Persons" This paper shows an idea about using espeak in order read the text using raspberry pi and give audio output. The proposed system helps the vision impaired people in their daily activities. This model helps to read the text and converts it in to audio form. It also helps to find out the obstacle by using PIR sensor and to find out the currency through use of a database. The PIR sensor alerts the blind user about the obstacle but cannot name the object. In future it can be further extended by using the latest obstacle detecting technologies and also helps in identifying the currency in real time. [4]

Chapter 3

System Analysis

3.1 Functional Requirements

There are a number of features that the Oculus offers out of which major features that are being highlighted are as follows:

• Capture image and Store

• Ultrasonic Sensors object detection

• Image to Speech Processing

3.1.1 Capture Image and Store

3.1.1.1 Description and Priority

This is used to capture the surrounding image in the sight where the blind person is looking wearing the cap. The primary working starts with capturing of the image from the camera module. The camera module is a part of the raspberry pi which is connected on the onboard storage which stores the image which is used for comparison with the existing objects in the trained objects. After the image is being processed for object the image is being deleted and the memory space is released.

3.1.1.2 Stimulus/Response Sequences

In case if the image has no match with any objects for the trained classes:

• Send this Image for processing in order to give a speech output to the user about the surroundings.

• The audio output is given through espeak.

In case if there is a match:

• Store the image as the part of the temporary image.

• Send the image for detecting the object.

• Delete the image after the audio output is delivered.

3.1.1.3 Functional Requirements

• This would give the images in the required format which is required by the raspberry pi to convert it to speech.

• Also, images will enable a better view of the surrounding and by processing a more descriptive narrative can be obtained as per the user's wish.

REQ1: Power supply to the raspberry pi.

REQ2: Adequate lighting.

REQ3: Camera working,

3.1.2 Ultrasonic Sensors Object Detection

3.1.2.1 Description and Priority

It is possible that the camera takes blurry image which is difficult for detection for an object. Also it is possible that while the processing is done the person moves too close to the object. In that case ultrasonic sensors are used as they detect the distance between the person and the object. If they are close to the object at the distance less then 50cm then a default audio is given. It is being attached on the cap and are used to detect any obstacle in the path.

3.1.2.2 Stimulus/Response Sequences

• On sensing any obstacle, it returns the distance in terms of centimeters.

• It checks for the given condition whether it is less than 50cm.

• If it is less than 50 in that case it sends an audio output to the individual indicating danger or obstacle.

3.1.2.3 Functional Requirements

• The ultrasonic sensor is used detects if the object is at the range of less than 50cm and notifies about it.

• Ultrasonic sensor also sends a signal indicating the user about the danger in terms of audio output through the headphones.

REQ1: Sensors on the cap in front.

REQ2: Connection with the raspberry pi.

REQ3: Raspberry pi works when the power source is applied.

3.1.3 Image to Speech Processing

3.1.3.1 Description and Priority

It is used to process the captured image in order to convert it to text. The image is being processed and the text file is being produced. Mobile Net + SSD is being used to detect the object when the image is being captured. The Caffe model has 20 classes out of which object which is used to detect the objects. After the detection then the text file containing the object classes is created. The espeak reads the text file and the output is given in headphones.

3.1.3.2 Stimulus/Response Sequences

• If there is no match and if the sensors detect a distance less than 50cm then an audio output is given.

• Otherwise the respective objects that are being detected are being given by the audio output.

Functional Requirements

• Description of the environment is provided.

• Based on the analysis appropriate detection of the object is given.

• Information from ultrasonic sensors is given in the form of distance.

REQ1: Camera module taking pictures.

REQ2: Working Ultrasonic Sensor.

REQ3: Power supply.

3.2 Non-Functional Requirements:

Performance Requirements:

• It should capture the image properly.

• The processing from to image to speech should be quick in order to provide real time information about the surroundings.

• Also the ultrasonic sensor should work independently and return the distance in order check the condition specified.

Safety Requirements:

• The hardware module integration should be placed such that no connections are being harmed.

• Proper padding should be given inside the cap so as to keep the individual's head safe from the components in direct contact with the body.

• The components are being covered by a larger cap so that the connections do not hinder the blind person from using the system.

Security Requirements:

• It should be ensured that there is no remote access possible on the raspberry pi chip in order to manipulate with it.

• Also, the captured images stored should be properly stored which should not be accessible to any outsider.

• In terms of hardware it should not be possible to tamper with any physical wiring or device used this is being achieved by the use of bigger cap for covering.

Software Quality Attributes:

• Availability: The cap at any point of time should be able to capture the image and then process it to give voice audio output of the detected object.

• Correctness: The information should be appropriate depending on which the individual will take the decision. The objects detected should be correctly classified.

• Robustness: The system should be fast enough so that the processing is done quickly and also achieve correct classification of objects.

• Testability: The system must be easy to test since it is based on raspberry pi working on Raspbian OS and also it should be able to connect the raspberry pi to the HDMI easily.

• Compatibility: It is standalone hardware unit working individually on cap unit so it should be compatible with the camera and ultrasonic sensor.

3.3 Specific Requirements:

Hardware Requirements:

• Raspberry Pi model B
• Camera Module with ribbon Clip
• Ultrasonic Sensors
• Rechargeable battery
• Cap
• Breadboard

Software Requirements:

• Raspbian OS installed on the Raspberry pi model B
• Python 2.7.
• Ubuntu for initial configuration of Raspberry pi.
• Espeak

Language Used:

• Python

It is requires a pair of headphones which is to be connected to the 3.5mm headphone jack which yields the speech output. The image is automatically deleted as soon as the speech output for the respective image is being given. In case of lack of light the image is captured is black in that case the object detection is difficult.

3.4 Use case diagram and description:

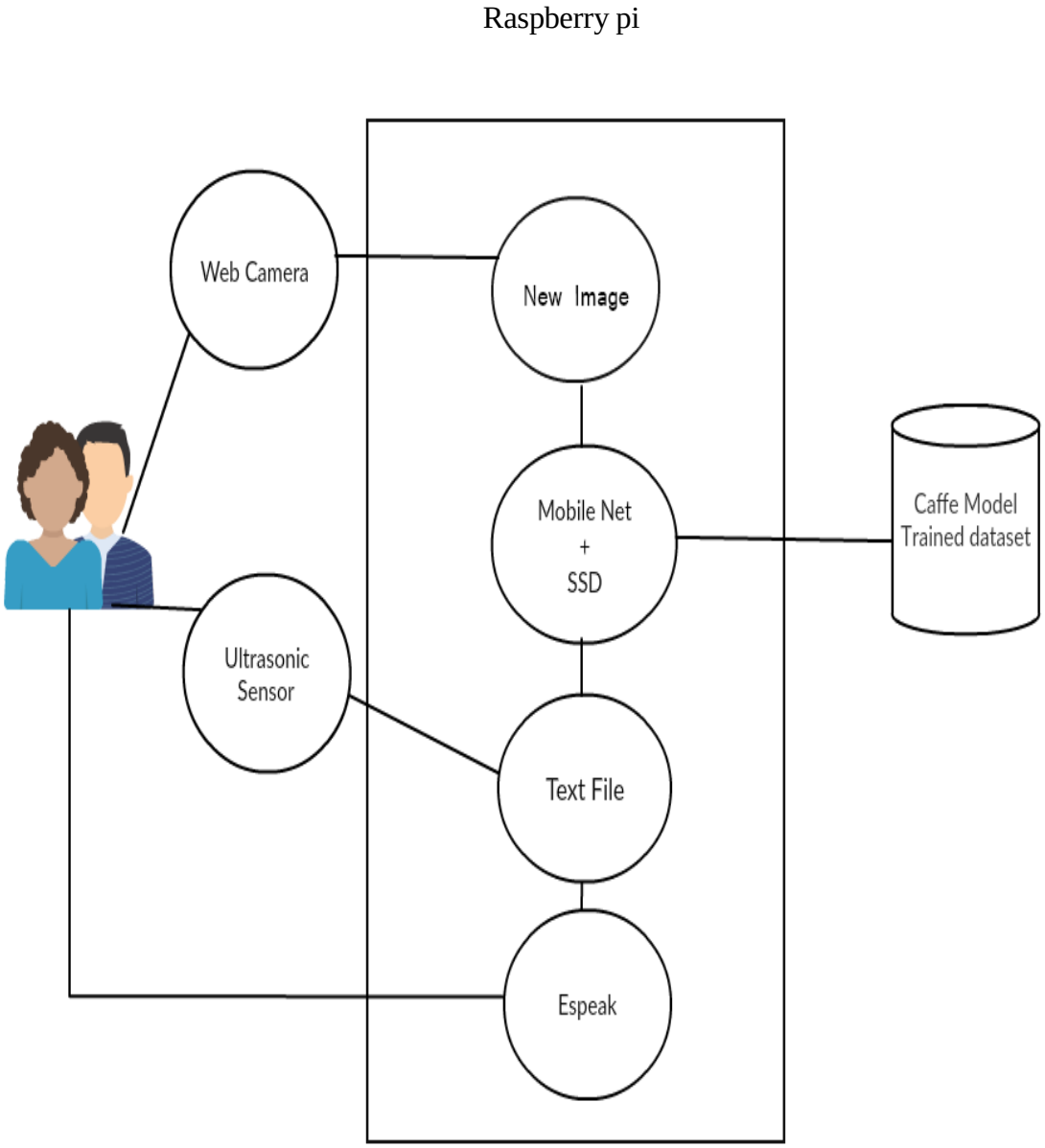

Fig 3.1 Use case diagram

The Use case diagram shows the entire interaction of the user with the system. The system majorly comprises of the raspberry pi chip which is the heart of the system. The camera and the ultrasonic sensor are the two ways by which the person can achieve a path of communication with the raspberry pi. The communication begins with the person wearing the cap and then turning on the device. After that the camera takes the image through the camera module and then sends the data to the raspberry pi and it checks whether there is an object which can belong to any of the predefined classes of the trained model. If there is a match then it detects the objects from the image and the corresponding text files is being created mentioning the object name and then being stored into the text database. The text file contains the information about the surrounding that is the object that's being identified. This text file is then passed to the espeak and then the audio output is given through the 3.5mm jack. The audio is to be heard of by using headphones. In case of new image of the surrounding is taken in that case the new image s processed and once the processing is done it is deleted to free up space while the ultrasonic sensors detect any obstacles. The Ultrasonic sensors senses the distance from the object and an audio is being generated in response to it. The output from both the modes of input reaches the user by means of sound.

Chapter 4

Analysis Modeling

4.1 Data Modeling

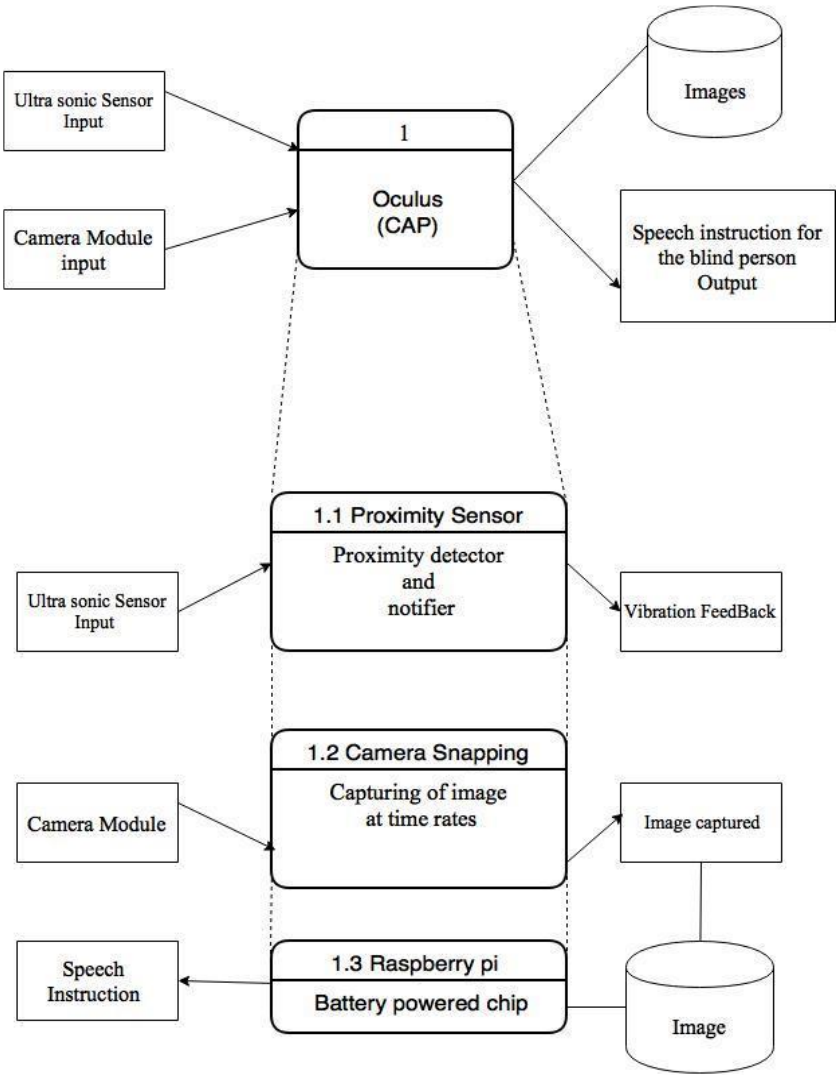

Fig 4.1. Data model

4.2 Activity diagram

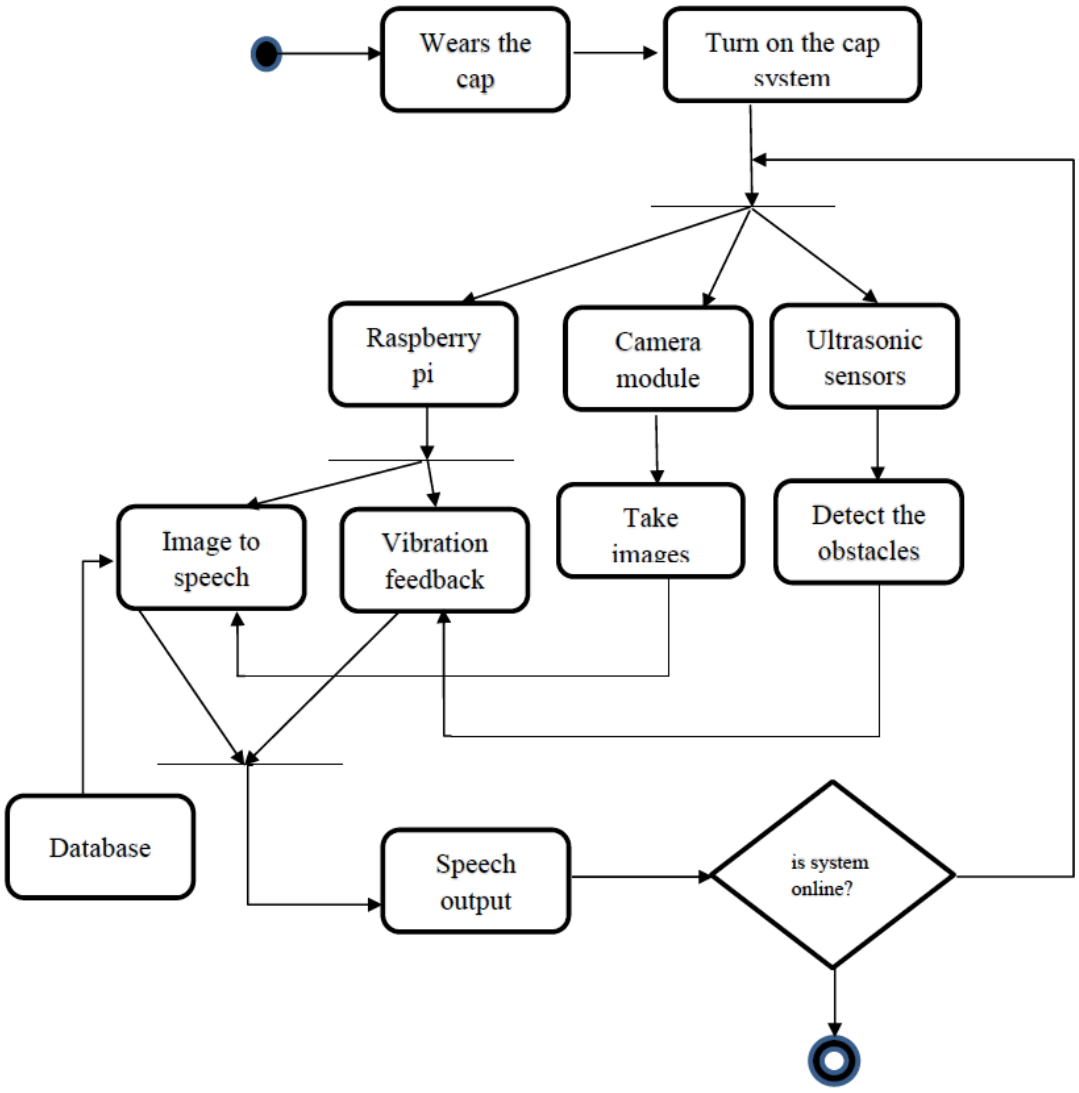

Fig 4.2. Activity diagram

4.3 Functional modeling

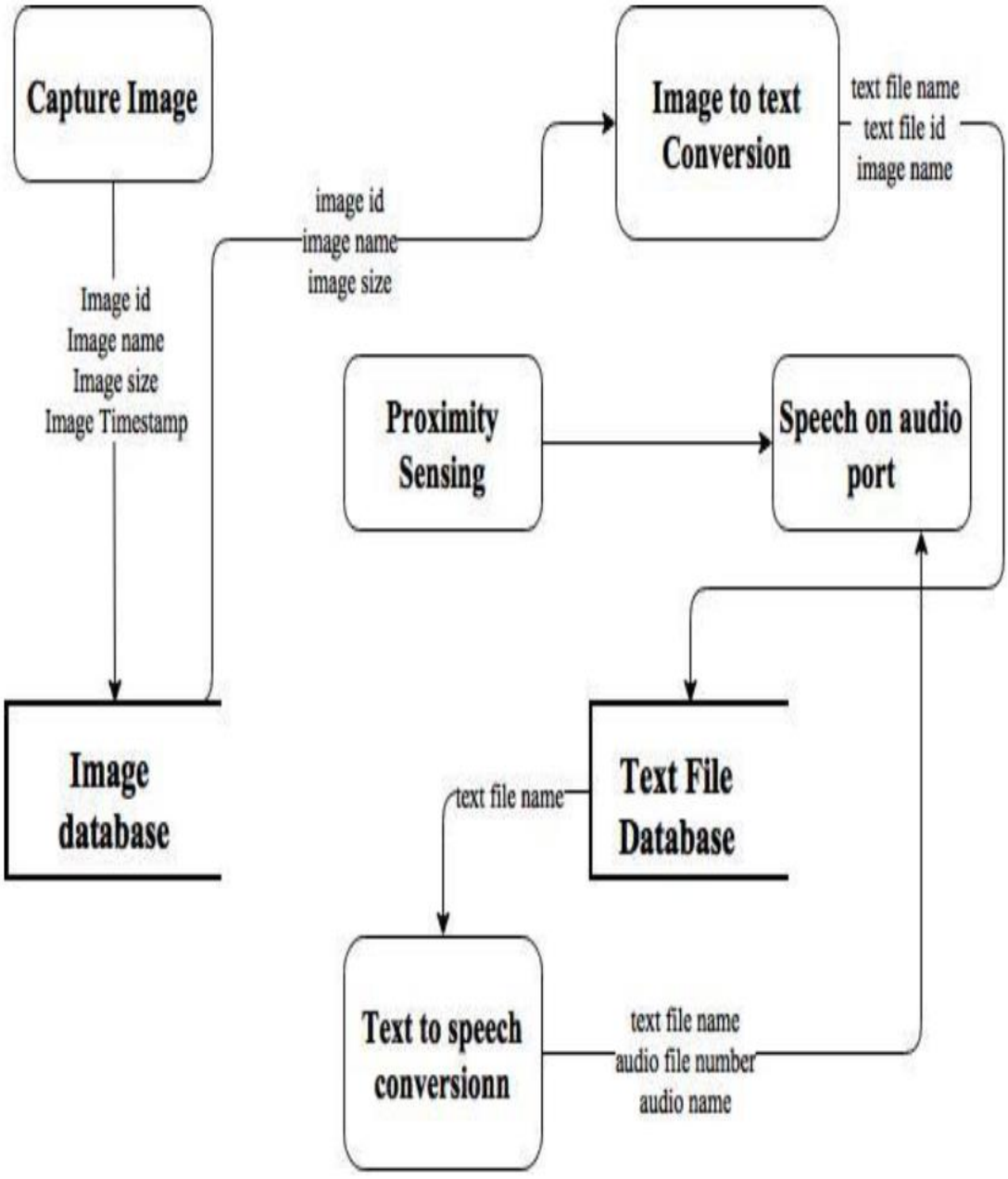

Fig 4.3. Functional Modeling

4.4 Timeline chart

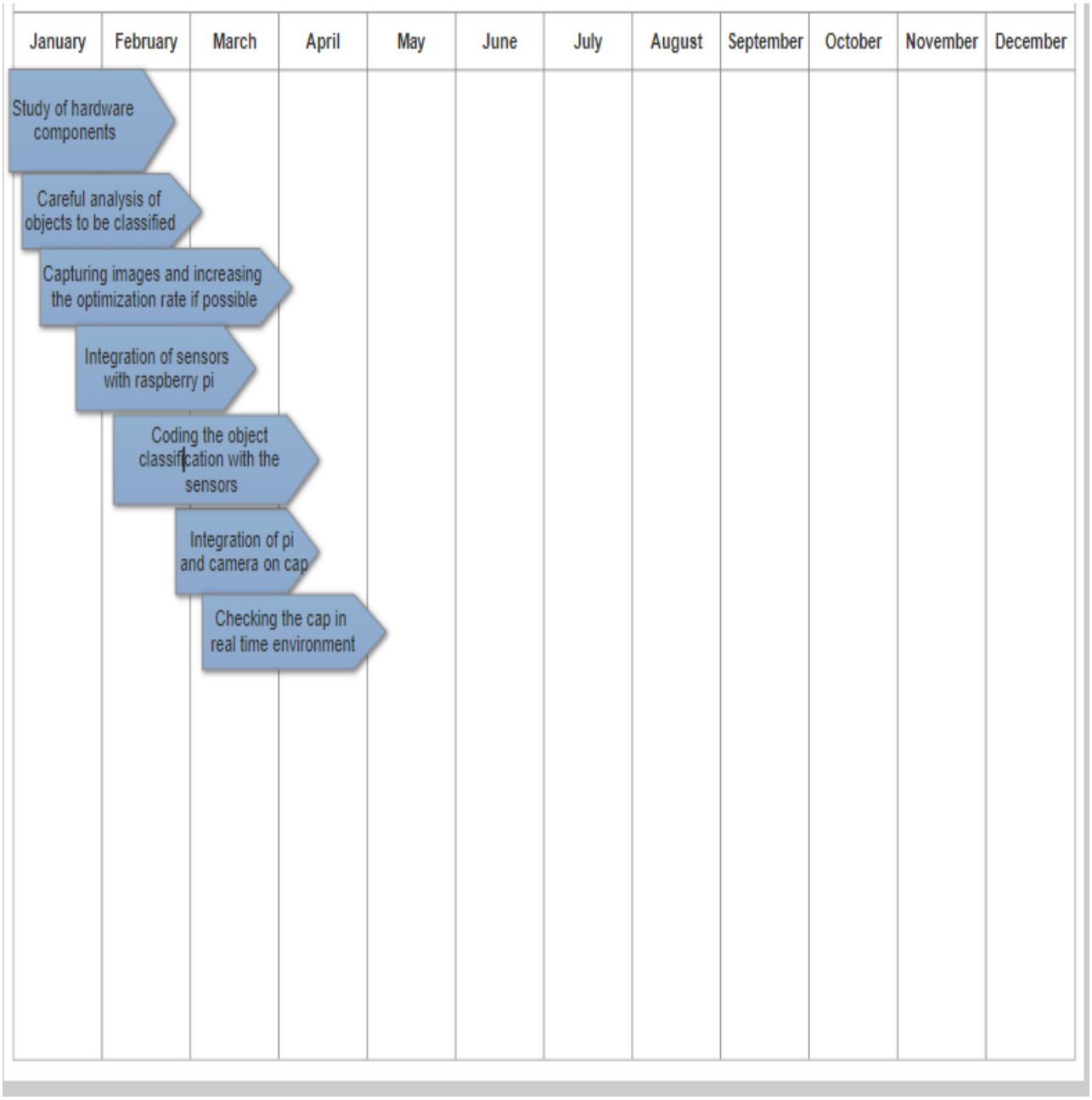

Fig 4.4. Timeline Chart

Chapter 5

Design

5.1 Architectural design:

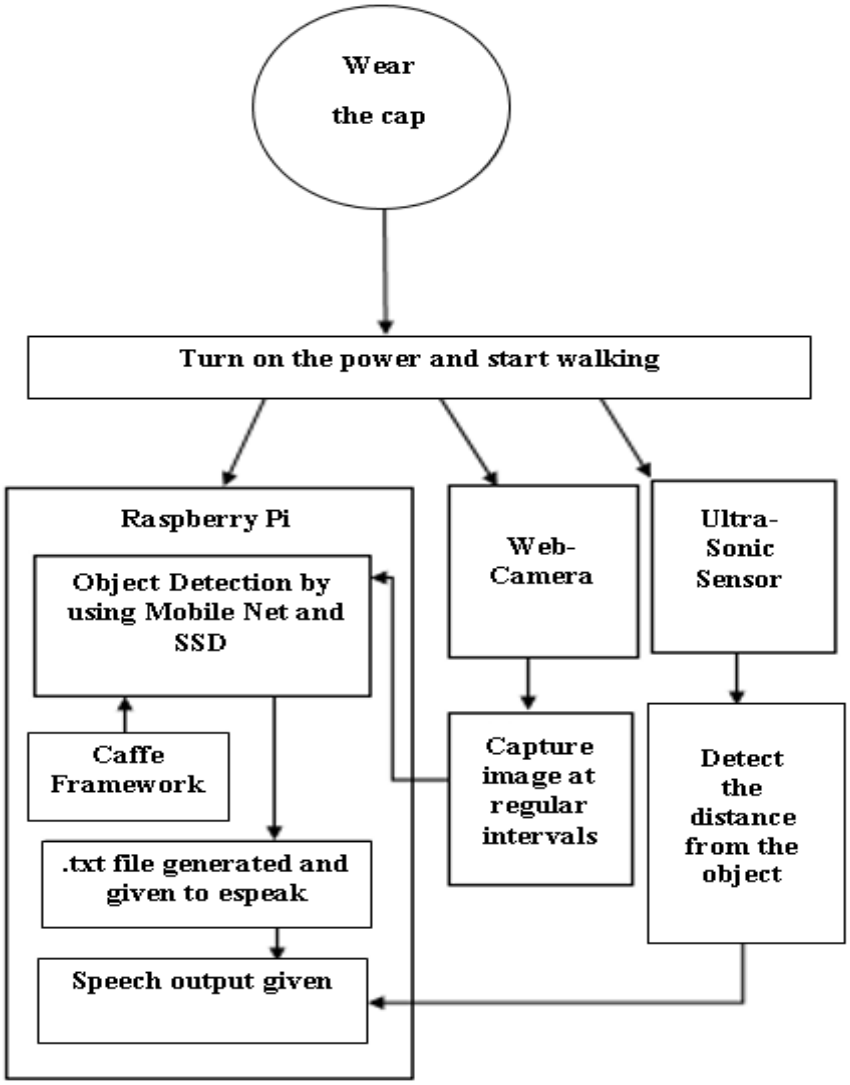

Fig 5.1. Architectural design

The architecture used in this case is being embedded on the cap. The raspberry pi does the job of processing and classifying the detected objects into the classes that is being defined for it. Apart from it, there is Web Camera and Ultrasonic sensor that works with the pi.

The Web Camera first clicks a picture after that it is given to the pi for further processing. The raspberry pi detects the objects present in the image and then tries to find the classes it may belong. After the detection is complete the corresponding objects with their class labels is being stored in the text file. Also the ultrasonic sensor detects the distance of the person from the object and returns it to the pi. If the distance is less than 50cm it gives a speech output being too close to the object.

The text file contains formatted text which is being read by Espeak and the output is given in the form of speech to user. The entire setup is being powered by a portable charger.

Chapter 6

Implementation

6.1 Algorithms / Methods Used

6.1.1 Methods used:

- Object detection with Deep learning (Convolutional neural network) and OpenCV
- MobileNets + Single Shot Detectors (SSD)
- Caffe Model

In this, we have used MobileNets and Single Shot Detectors (SSD) for fast and efficient deep learning based object detection and deep neural network (dnn) module in OpenCV to build our object detector. We call these networks "MobileNets" because they are designed for resource constrained devices such as your smartphone. MobileNets differ from traditional CNNs through the usage of depthwise separable convolution. [1]

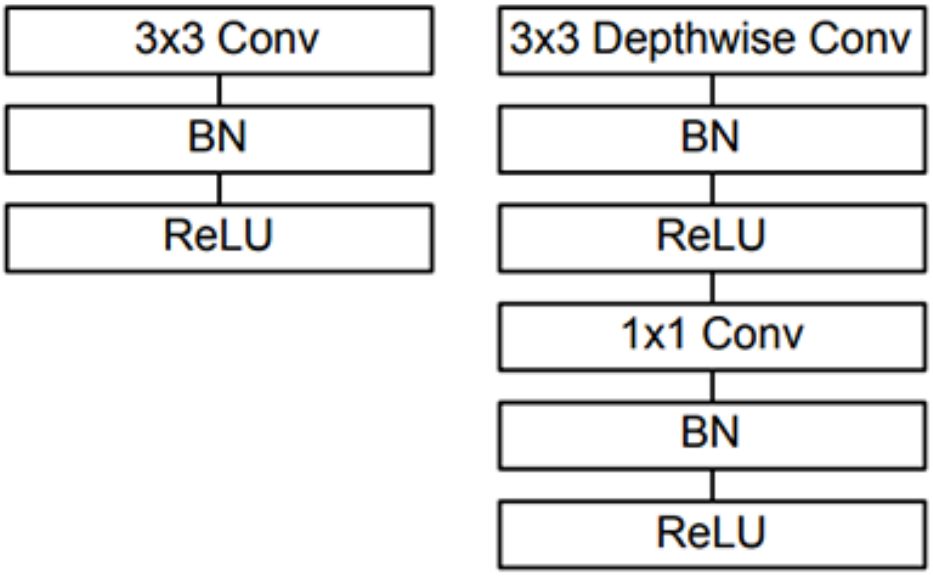

Fig 6.1: (Left) Standard convolutional layer with batch normalization and ReLU. (Right) Depthwise separable convolution with depthwise and pointwise layers followed by batch normalization and ReLU

The model that we have used is a trained Caffe model of the original TensorFlow. The MobileNet SSD was first trained on the COCO dataset (Common Objects in Context).We can therefore detect 20 objects in images (+1 for the background class), including airplanes, bicycles, birds, boats, bottles, buses, cars, cats, chairs, cows, dining tables, dogs, horses, motorbikes, people, potted plants, sheep, sofas, trains, and tv monitors.

6.2 Working of the project

MobileNet SSD + deep neural network (dnn) module in OpenCV to build our object detector.

Code for implementation:

```
import numpy as np
import argparse
import cv2
import os
from subprocess import call
import RPi.GPIO as GPIO
import time

ap = argparse.ArgumentParser()
ap.add_argument("-c", "--confidence", type=float, default=0.2,
        help="minimum probability to filter weak detections")
args = vars(ap.parse_args())

CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",
        "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
        "dog", "horse", "motorbike", "person", "pottedplant", "sheep",
        "sofa", "train", "tvmonitor", "mobile"]
COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))
```

```
print("[INFO] loading model...")
net = cv2.dnn.readNetFromCaffe("/home/pi/Downloads/MobileNetSSD_deploy.prototxt.txt",
"/home/pi/Downloads/MobileNetSSD_deploy.caffemodel")

cam = cv2.VideoCapture(0)
cv2.namedWindow("test")

img_counter = 0
i = 0
distance = 0

def sense():
    GPIO.setmode(GPIO.BCM)
    TRIG = 23
    ECHO = 24
    print "Distance.."
    GPIO.setup(TRIG,GPIO.OUT)
    GPIO.setup(ECHO,GPIO.IN)
    GPIO.output(TRIG,False)
    print "Waiting for sensor to settle"
    GPIO.output(TRIG,True)
    time.sleep(0.00001)
    GPIO.output(TRIG,False)

    while GPIO.input(ECHO)==0:
        pulse_start = time.time()

    while GPIO.input(ECHO)==1:
        pulse_end = time.time()
    pulse_duration=pulse_end - pulse_start
    distance = pulse_duration * 17150
```

```
    distance = round(distance, 2)
    print "Distance:",distance,"cm"
    GPIO.cleanup()
    time.sleep(1)
    return distance

while(cam.isOpened()):
    ret, frame = cam.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imshow("test", gray)
    k = cv2.waitKey(1)
    if(i%100)==0:
        dist = sense()
    if k%256 == 27:
        print("Escape hit, closing...")
        break

    elif (i%100)==0:
        img_name = "image_{}.jpg".format(img_counter)
        path = "/home/pi/Downloads/Images"
        cv2.imwrite(os.path.join(path, img_name), gray)
        print("{} written!".format(img_name))
        img_counter += 1
        image = cv2.imread(os.path.join(path, img_name))
        (h, w) = image.shape[:2]
        blob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)), 0.007843, (300, 300),
127.5)
        print("[INFO] computing object detections...")
        net.setInput(blob)
        detections = net.forward()
        f = open("new.txt","w")
```

```
        f.write("Caution\n")

        for i in np.arange(0, detections.shape[2]):
            confidence = detections[0, 0, i, 2]

            if confidence > args["confidence"]:
                idx = int(detections[0, 0, i, 1])
                box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
                (startX, startY, endX, endY) = box.astype("int")

                label = "{}".format(CLASSES[idx])
                print("[INFO] {}".format(label))

                cv2.rectangle(image, (startX, startY), (endX, endY),
                                        COLORS[idx], 2)
                y = startY - 15 if startY - 15 > 15 else startY + 15
                cv2.putText(image, label, (startX, y),
                cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)
                f.write("There is a {}\n".format(label))
        f.close()
        with open('new.txt', 'r'):
            call('espeak -f new.txt',shell=True)
            if(dist < 100):
                os.system("espeak -s70 'You are close to the object'")

    i+=1
cam.release()
cv2.destroyAllWindows()
```

Chapter 7

Testing

7.1 Test cases

Test Case ID	TC001
Test Case Summary	To verify whether the camera is working properly or not.
Prerequisites	1. Camera is connected to Raspberry pi 2. Power source is connected
Test Procedure	1. Run the code for checking the working of camera. 2. Wait for the image to be clicked. 3. Check the image clicked.
Expected Result	1. The code should not run in loop. 2. The lights of the camera are on and the image is clicked.
Actual Result	The image is clicked and viewed successfully.
Status	Pass
Test Environment	• Hardware: Raspberry pi, Webcam • OS: Raspbian • Language: python
Date of Creation	12/02/2018
Date of Execution	03/03/2018

Test Case ID	TC002
Test Case Summary	To verify the detection of objects.
Prerequisites	1. Camera is connected to Raspberry pi 2. Power source is connected
Test Procedure	1. Run the code. 2. Wait until the object is detected 3. Once object is detected, the audio output is given to the user.
Expected Result	1. The object is detected. 2. The audio output was given to the user
Actual Result	1. The object is detected. 2. The audio output was given to the user
Status	Pass
Test Environment	• Hardware: Raspberry pi, Webcam • OS: Raspbian • Language: python
Date of Creation	20/02/2018
Date of Execution	10/03/2018

Test Case ID	TC003
Test Case Summary	To check the working of sensors.
Prerequisites	1. Camera is connected to Raspberry pi 2. Power source is connected 3. Sensor is connected
Test Procedure	1. Run the code. 2. Wait until the sensor sense any obstacle.
Expected Result	The sensor sensed the obstacle and returned the output.
Actual Result	The sensor sensed the obstacle and returned the output.
Status	Pass
Test Environment	• Hardware: Raspberry pi, Webcam • OS: Raspbian • Language: python
Date of Creation	01/03/2018
Date of Execution	21/03/2018

7.2 Type of Testing used

- **Black-box testing:** Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. We have tested the working of:
  - Raspberry pi
  - Webcam
  - Sensor

- **System Testing:** System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.
  We have tested that all the hardware components like raspberry pi, camera and sensors are working together correctly or not.

- **Performance testing:** Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.

Performance testing techniques:

- **Load testing -** It is the simplest form of testing conducted to understand the behavior of the system under a specific load. The testing of raspberry pi behavior on real time environment and on integrating camera and sensor.
- **Stress testing -** It is performed to find the upper limit capacity of the system and also to determine how the system performs if the current load goes well above the expected maximum. The testing on raspberry pi is done to check if the load is balanced when integrated with camera and sensor. Also the camera is tested by checking how much time it can be kept on. The sensor is tested to check the upper limit capacity of sensor detection length.

Chapter 8

Results and Discussions

The current stage of the project offers a low cost solution for obstacle detection and classification in one of the 20 classes. The entire set up is being powered on a Raspberry pi Model B. At the given instant the source of power is the portable charger which holds the capacity of 5000 mah.

The connection of the Ultrasonic sensor is done via a circuitry that is present on the breadboard covered by the larger hat. The entire setup is being mounted on the hat which is being covered by a larger hat to cover the components that might get displaced if not covered. The headphones are being provided from inside the cap so that the wires are not much of a concern.

The booting up of the hat is done by just connecting the magnetic wire to the power source of raspberry pi and pressing the button on the portable battery. This starts the raspberry pi and boots up the device and the program starts running directly. The initial setup that notifies that the system is ready and the person can start walking is by given an audio output “Caution”. When this audio output is obtained the user can start walking.

The ultrasonic sensor sits at the front detecting the distance from the object. It is constrained by the distance of 50cm. If the user enters the proximity of less than 50cm of the object in that case an audio output is given enabling that the user is too close to the object. It just notifies that there is an object which is close by. This is used in case if the processing takes a while or the person reaches very near to the object.

The current context of the project is limited to the eyesight vision that limits to the obstacles for example stairs, pothole, and railway platform. This are not being identified as they are not captured in the web camera nor they can be detected via the ultrasonic sensor since they are on the hat.

In case when an image is captured by the web Camera it is being divided into frames to scan for the objects across the image. Once a corresponding object is found then the class label to which the object belongs along with a formatted text format is stored in a .txt file. The output after object detection of an image can be shown as follows

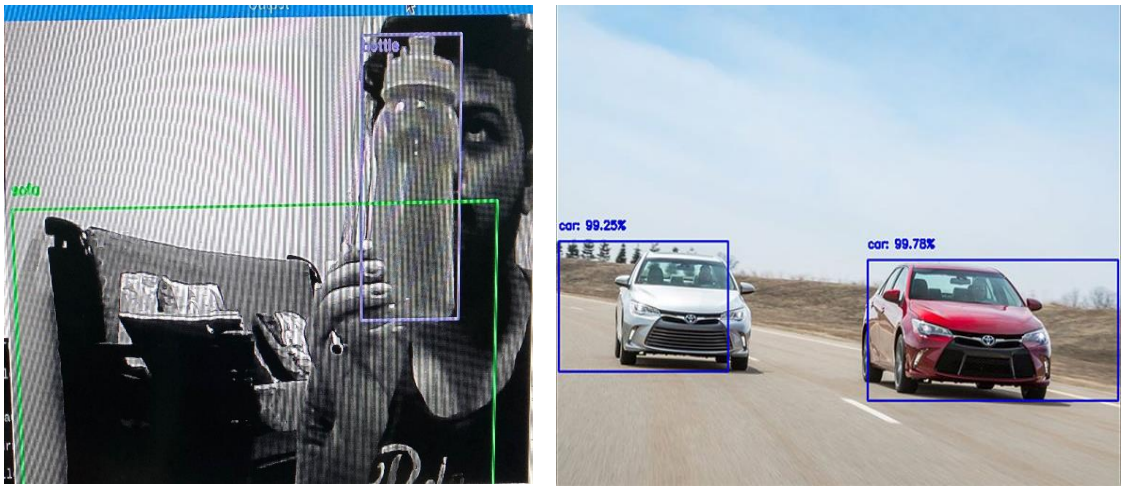

Fig 8.1 Object detection Output

The image has two objects being detected which is the bottle and the sofa. The corresponding formatted text for image is being shown below

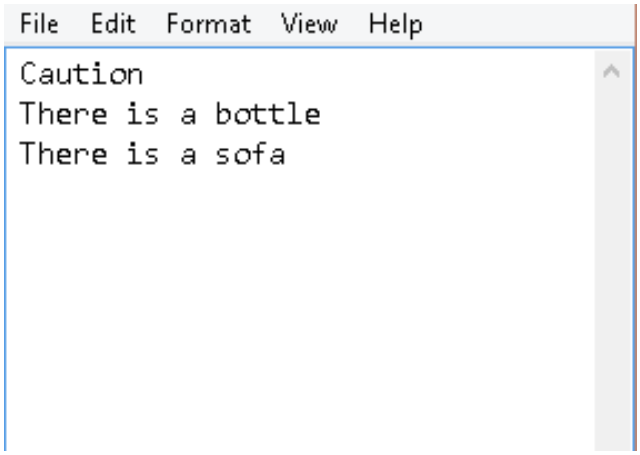

Fig 8.2 Text file for the corresponding object detection

Chapter 9

Conclusions & Future Scope

In this project from the results it can be concluded that the current focus is on the eyesight level. Oculus aims on providing vision at the eyesight level. It is in the form of elevated vision that detects any obstacles captured in the image which can be classified. The ultrasonic sensor work is to notify the user when he/she is close to the object. In terms of what is close for a person is less than 50cm with the object. In the current time what oculus can do is detect the object and classify it among the 20 classes that are present. Apart from it, it can also notify the user about if they are too close to the object.

For the current iteration we have mounted the entire setup of raspberry pi portable charger and ultrasonic sensor with the circuitry on the hat. The entire thing is covered by another cap in order to hide the components not enabling the complex nature view of the system and keeping it simple and tangle free.

Moreover the insight of the insight behind the project was to enable the comparative lost cost solution as compared to what is available in the market. The major cost that escalates the cost is the cost of the raspberry pi model B. The setup is kept for the offline use because it is possible that the range of internet may not always be strong to achieve this processing from the server end and the delay may be more than what is desired which will cost the failure of our objective to provide quick detection and classification of the objects.

There are lots of improvements that can be achieved with the upcoming version of Oculus, since this is first step towards the society to contribute our part as a group of individuals. Some of the improvements that can be achieved are as follows:

- Adding more number of classes for classification
- Reducing the time for classification
- Adding a GPS module for navigation
- Enabling depth sensing
- Integrating Ultrasonic Sensors in the shoes and connecting it to raspberry pi via Wi-Fi
- Getting a more compressed embedded circuitry so that the wiring is reduced
- Providing an easy access to on and off button

Appendix

Appendix I

Caffe Model

Caffe is a deep learning framework made with expression, speed, and modularity in mind. It is developed by Berkeley AI Research (BAIR) and by community contributors. Yangqing Jia created the project during his PhD at UC Berkeley. Caffe is released under the BSD 2-Clause license.

Expressive architecture encourages application and innovation. Models and optimization are defined by configuration without hard-coding. Switch between CPU and GPU by setting a single flag to train on a GPU machine then deploy to commodity clusters or mobile devices.

Extensible code fosters active development. In Caffe's first year, it has been forked by over 1,000 developers and had many significant changes contributed back. Thanks to these contributors the framework tracks the state-of-the-art in both code and models.

Speed makes Caffe perfect for research experiments and industry deployment. Caffe can process over 60M images per day with a single NVIDIA K40 GPU*. That's 1 ms/image for inference and 4 ms/image for learning and more recent library versions and hardware are faster still. It is believed that Caffe is among the fastest convnet implementations available.

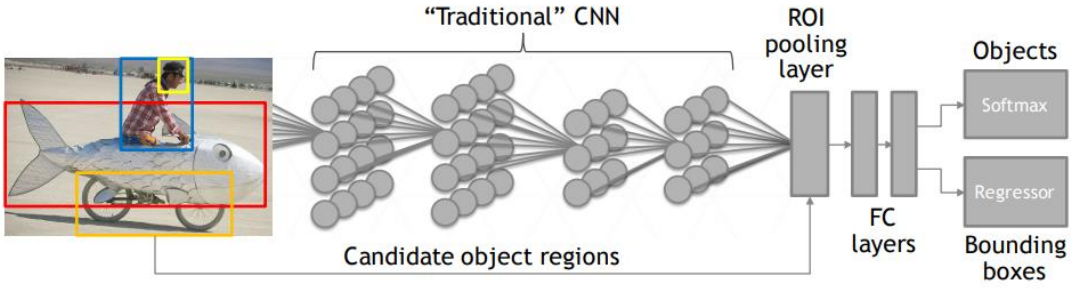

Fig A1.1 Caffe model

Lots of researchers and engineers have made Caffe models for different tasks with all kinds of architectures and data: check out the model zoo! These models are learned and applied for problems ranging from simple regression, to large-scale visual classification, to Siamese networks for image similarity, to speech and robotics applications. To help share these models, we introduce the model zoo framework a standard format for packaging Caffe model info.

A caffe model is distributed as a directory containing:

- Solver/model prototxt(s)
- YAML frontmatter
- Information about what data the model was trained on, modeling choices, etc.
- License information.
- Other helpful scripts.

This simple format can be handled through bundled scripts or manually if need be.

Appendix II

Ultrasonic Distance Sensor

Sound consists of oscillating waves through a medium (such as air) with the pitch being determined by the closeness of those waves to each other, defined as the frequency. Only some of the sound spectrum (the range of sound wave frequencies) is audible to the human ear, defined as the "Acoustic" range. Very low frequency sound below Acoustic is defined as "Infrasound", with high frequency sounds above, called "Ultrasound". Ultrasonic sensors are designed to sense object proximity or range using ultrasound reflection, similar to radar, to calculate the time it takes to reflect ultrasound waves between the sensor and a solid object. Ultrasound is mainly used because it's inaudible to the human ear and is relatively accurate within short distances.

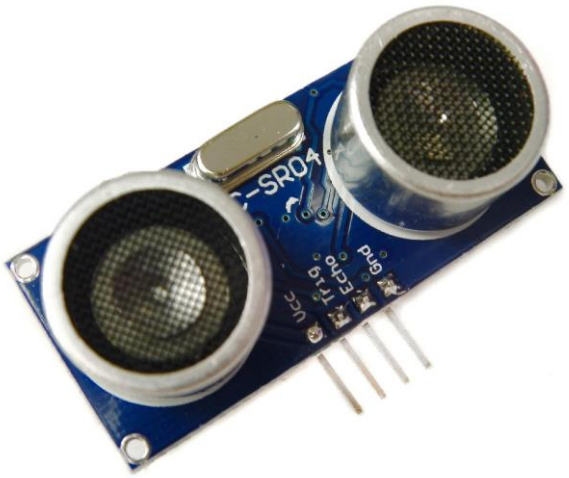

Fig A2.1 Ultrasonic sensor

A basic ultrasonic sensor consists of one or more ultrasonic transmitters (basically speakers), a receiver, and a control circuit. The transmitters emit a high frequency ultrasonic sound, which bounce off any nearby solid objects. Some of that ultrasonic noise is reflected and detected by the receiver on the sensor. That return signal is then processed by the control circuit to calculate

the time difference between the signal being transmitted and received. This time can subsequently be used, along with some clever math, to calculate the distance between the sensor and the reflecting object.

The HC-SR04 Ultrasonic sensor used for the Raspberry Pi has four pins: ground (GND), Echo Pulse Output (ECHO), Trigger Pulse Input (TRIG), and 5V Supply (Vcc). The power is given to the module using Vcc, ground it using GND, and use our Raspberry Pi to send an input signal to TRIG, which triggers the sensor to send an ultrasonic pulse. The pulse waves bounce off any nearby objects and some are reflected back to the sensor. The sensor detects these return waves and measures the time between the trigger and returned pulse, and then sends a 5V signal on the ECHO pin. ECHO will be "low" (0V) until the sensor is triggered when it receives the echo pulse. Once a return pulse has been located ECHO is set "high" (5V) for the duration of that pulse. Pulse duration is the full time between the sensor outputting an ultrasonic pulse, and the return pulse being detected by the sensor receiver. The assembly of the entire circuit is shown as follows:

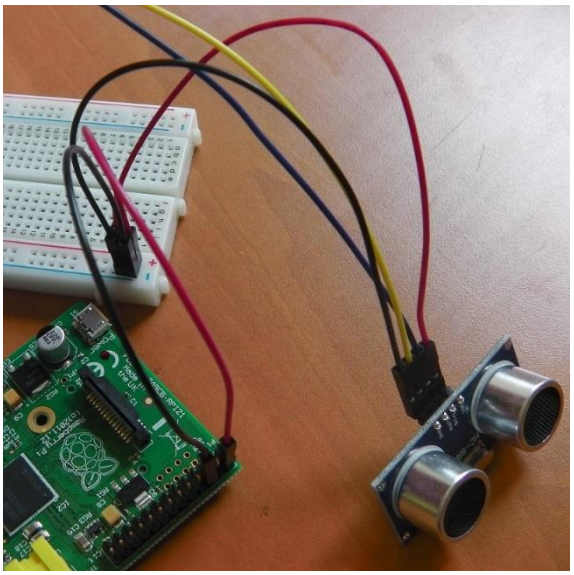

Fig A2.2 Basic circuit

Literature Cited

[1] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, Google Inc "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications" arXiv: 1704.04861v1 [cs.CV] 17 Apr 2017

[2] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg UNC Chapel Hill, Zoox Inc. ,Google Inc. ,University of Michigan, Ann-Arbor "SSD: Single Shot MultiBox Detector" arXiv:1512.02325v5 [cs.CV] 29 Dec 2016

[3] Nur Aziza Azis, Rose Maulidiyatul Hikmah, Teresa Vania Tjahja and Anto Satriyo Nugroho Center for Information and Communication Technology (PTIK) Agency for the Assessment and Application of Technology (BPPT), Jalan M.H. Thamrin No. 8, Jakarta 10340, Indonesia "Evaluation of Text-to-Speech Synthesizer for Indonesian Language Using Semantically Unpredictable Sentences Test: IndoTTS, eSpeak, and Google Translate TTS" Proc. of International Conference on Advanced Computer Science & Information Systems, 2011

[4] D. B. K. Kamesh*, S. Nazma, J. K. R. Sastry and S. Venkateswarlu Department of Electronics and Computer Engineering, KL University, Vaddeswaram - 522 502, Guntur District, Andhra Pradesh, India "Camera based Text to Speech Conversion, Obstacle and Currency Detection for Blind Persons" Indian Journal of Science and Technology, Vol 9(30), DOI: 10.17485/ijst/2016/v9i30/98716, August 2016

[5] Onkar R. Kirpan, Pooja I. Baviskar, Shivani D. Khawase, Anjali S. Mankar, Karishma A. Ramteke Department of Computer Science and Engineering Ragiv Gandhi College of Engineering and Research, Nagpur University, Nagpur, India "Object Detection on Raspberry Pi" International Journal of Engineering Science and Computing, March 2017

[6] Vijayalaxmi, K.Anjali,B.Srujana, P.Rohith Kumar "OBJECT DETECTION AND TRACKING USING IMAGE PROCESSING" Global Journal of Advanced Engineering Technologies, Special Issue (CTCNSF-2014)

[7] Pallavi P. Saraikar, Prof. K.S.Ingle "Image Processing Target tracking Robot using Raspberry pi" International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 5, Issue 6, June 2017

Acknowledgements

It gives us immense pleasure to express our deepest sense of gratitude and sincere thanks to our highly respected and esteemed guide Asst. Prof. Aruna Gawade, for her valuable guidance, encouragement and help for completing this work. Her useful suggestions for this whole work and co-operative behavior is sincerely acknowledged.

We would like to express our sincere thanks our Head of Department, Dr. Narendra M. Shekokar for giving us this opportunity to undertake this project. We would also like to thank Dr. Hari Vasudevan, principal for his whole hearted support.

We also wish to express our gratitude to the entire Computer Engineering Department for their kind hearted support. Also providing us required guidance in our respective problem areas.

We would like to express our sincere thanks to all the non-teaching who helped us directly or indirectly during this project work.