**Smart Blind Stick**

Table of Contents

[**Smart Blind Stick** 1](#_Toc105430699)

[Chapter 1 5](#_Toc105430700)

[**1.1 Aim** 5](#_Toc105430701)

[**1.2 PROBLEM STATEMENT** 5](#_Toc105430702)

[Chapter 2 8](#_Toc105430703)

[**2.1 Circuit Diagram** 8](#_Toc105430704)

[**2.2 Components List** 9](#_Toc105430705)

[**2.3 Arduino Nano** 10](#_Toc105430706)

[**2.3.1 Tech specs** 10](#_Toc105430707)

[**2.4.0 Ultrasonic** 12](#_Toc105430708)

[**2.5 Buzzer** 14](#_Toc105430709)

[**2.5.1 Specification:** 14](#_Toc105430710)

[**2.6 Vibration Motor** 15](#_Toc105430711)

[**2.7 Switch Button on/off** 16](#_Toc105430712)

[**2.8 Battery 18650** 16](#_Toc105430713)

[**2.9 Raspberry pi 3 b+ UK** 18](#_Toc105430714)

[**2.9.1 FEATURES** 18](#_Toc105430715)

[**2.9.2 SPECIFICATIONS** 18](#_Toc105430716)

[**2.9.3 MECHANICAL DRAAWING** 20](#_Toc105430717)

[**2.10 Spo2** 21](#_Toc105430718)

[**2.10.1 Medical uses** 22](#_Toc105430719)

[**2.10.2 Advantages** 22](#_Toc105430720)

[**2.10.3 Limitations** 23](#_Toc105430721)

[**2.10.4 Equipment** 25](#_Toc105430722)

[**2.10.5 Mobile apps** 26](#_Toc105430723)

[**2.11 Water sensor** 27](#_Toc105430724)

[**2.11.1 Specifications** 27](#_Toc105430725)

[**2.12 GPS** 29](#_Toc105430726)

[**2.12.1 Hardware Overview of NEO-6M GPS Module** 29](#_Toc105430727)

[**2.12.2 specifications:** 31](#_Toc105430728)

[**2.12.3 Position Fix LED Indicator** 32](#_Toc105430729)

[**2.12.4 3.3V LDO Regulator** 32](#_Toc105430730)

[**2.12.5 Battery & EEPROM** 33](#_Toc105430731)

[**2.12.6 Antenna** 34](#_Toc105430732)

[**2.12.7 NEO-6M GPS Module Pinout** 35](#_Toc105430733)

[**2.12.8 Connections:** 36](#_Toc105430734)

[**2.12.9 Software Part:** 38](#_Toc105430735)

[**2.12.10 Setup for writing the Python Code:** 40](#_Toc105430736)

[**2.12.11 Python Code:** 42](#_Toc105430737)

[Chapter 3 44](#_Toc105430738)

[**3.1 Object Detection** 44](#_Toc105430739)

[**3.2 Image Processing** 44](#_Toc105430740)

[**3.3 Yolo v3** 45](#_Toc105430741)

[**3.4 OpenCV** 46](#_Toc105430742)

[**3.5 Hardware** 46](#_Toc105430743)

[**3.6 Working** 47](#_Toc105430744)

[**3.7 Block Diagram & Proposed Platform** 47](#_Toc105430745)

[**3.8 Result and Experiment** 47](#_Toc105430746)

[**3.9 FUTURE SCOPE** 48](#_Toc105430747)

[**3.9 Code** 49](#_Toc105430748)

[**3.9.1Test camera** 49](#_Toc105430749)

[**3.9.2 Text to Speech** 50](#_Toc105430750)

[**3.9.3 real time object detection with Text to Speech** 52](#_Toc105430751)

[Chapter 4 57](#_Toc105430752)

[**4.1.1 Web Site** 57](#_Toc105430753)

[**4.1.2 Website Design** 59](#_Toc105430754)

[**4.1.3 Code** 60](#_Toc105430755)

[**4.2 Google Firebase** 61](#_Toc105430756)

[**4.2.1 About Google Firebase** 61](#_Toc105430757)

[**4.3 USB modem** 64](#_Toc105430758)

[**4.3.1 Connecting 3G / 4G USB modem to Raspberry Pi 3** 64](#_Toc105430759)

# Chapter 1

## **1.1 Aim**

Abstract: Smart Blind Stick is a device designed to help guide the visually impaired by detecting objects and portray the information to them in the form of speech. This reduces the human effort and gives better understanding of the surrounding. Furthermore it also provides an opportunity for visually impaired people to move from one place to another without being assisted by others. The device can also be used in old age homes where old age people have difficulty in their day to day activities due to decreased vision. With this paper, the aim to aid people in need to “see” the surroundings. Since the field of artificial intelligence is doing great progress now and features like object detection is getting easier and computationally feasible, these

features are implemented in the paper. The paper focuses on object detection and classification on pictures which are captured by the device mounted on a stick whose information can then be relayed to the user in means of sound or speech.

## **1.2 PROBLEM STATEMENT**

According to World Health Organization (WHO), there are over 1.3 billion people who are visually impaired across the globe [1], out of which more than 36 million people are blind. India being the second largest population in the world, contributes 30% of the overall blind population. Although there are enough campaigns being conducted to treat these people, it has been difficult to source all the requirements. It is the era of artificial intelligence and it has gained immense traction due to large amount of data and ease of computation [2]. Using artificial intelligence it is possible to make these people’s life much easier. The goal is to provide a “secondary sight” until they have enough resources required to treat them. People with untreatable blindness can use this to make their everyday tasks much easier and simpler. II. INTRODUCTION The paper focuses on making a device which is portable and provides a “secondary vision”. The device consists of a Raspberry Pi, a Hi-Res Camera, an object detection algorithm (YOLO) and a text to speech unit (eSpeak).This unit can be mounted on a stick from where, it can capture images and process them. The device alerts the user if they come across any obstacles and give the description of what is in front of them. It can classify objects using directory of self-learned models.

Since computation on the raspberry pi is pretty good, the images captured by the camera can be locally processed. The insight gained from these self-learned models can generate what is present in the images. This information is then converted to text. Then the text-to-speech module can inform the user about their surroundings. Smart Blind Stick is capable of detecting 80 classes in real time. The steps involved to do so are, repeatedly capture the image from the

camera, pass these images through the classifier and the results obtained from this are then read from the text to speech engine. The time it takes for the object detection is 0.426 seconds on average. The text to speech may take time to relay the information to the user depending on the number of objects present in front of them. There’s an ultrasonic sensor which captures the distance of the nearest object and says it to the user along with the results from the object detection. To achieve this, the device has to have a smaller footprint yet high computation requirement. The device should be affordable as well. So, raspberry pi was chosen for the price to performance ratio and its small size. It is also opensource and has a huge community support since it works on GNU/Linux platform. Wide range of distributions can be found and there are many libraries and frameworks which ease the programming. There’s also a camera which can be interfaced with it very easily. Section 1 defines the problem the paper aims on solving and the motivation for the paper. Section 2 gives us an introduction of the paper and how to go forth to solve this problem with the paper briefly. Section 3 contains all the related work of devices which help the blind. Section 4 gives us the methodology of the paper i.e. how the device is made and the required frameworks and programs in detail. Section 5 contains the results of the paper and what the user is going to get out of this paper. Section 6 contains the conclusions that can be drawn from this paper and the steps that can be taken for future improvements to make it even better.

# Chapter 2

## **2.1 Circuit Diagram**

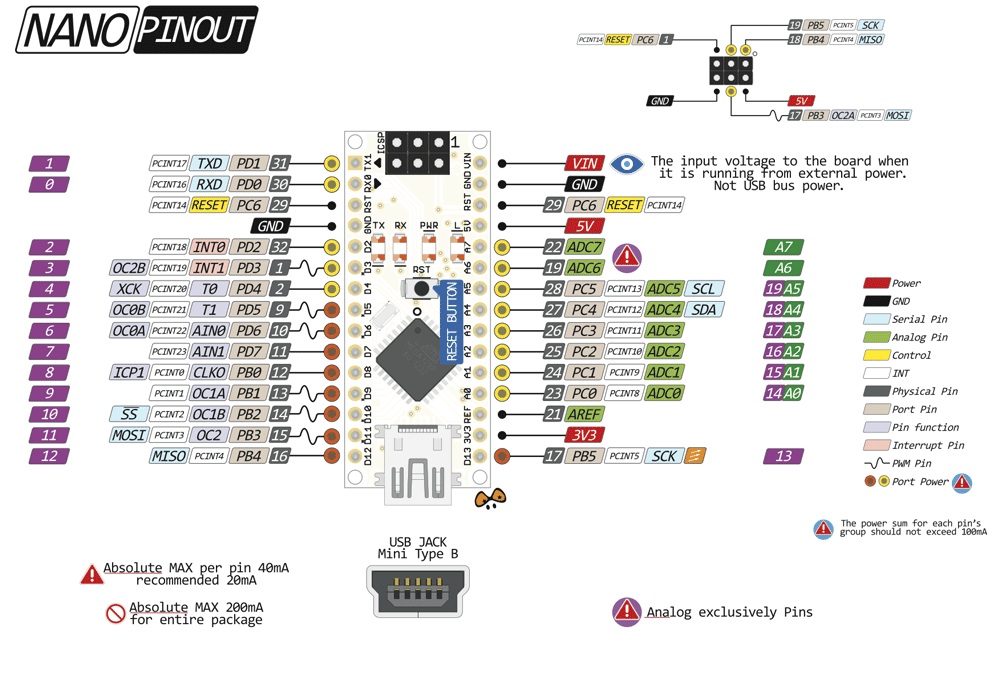
## **2.2 Components List**

1. **Arduino Nano**
2. **Ultrasonic**
3. **Buzzer**
4. **Vibration Motor**
5. **Switch Button on/off**
6. **Battery 18650**
7. **Raspberry pi 3 UK**
8. **Spo2**
9. **Water sensor**
10. **GPS**
11. **USB modem**

## **2.3 Arduino Nano**

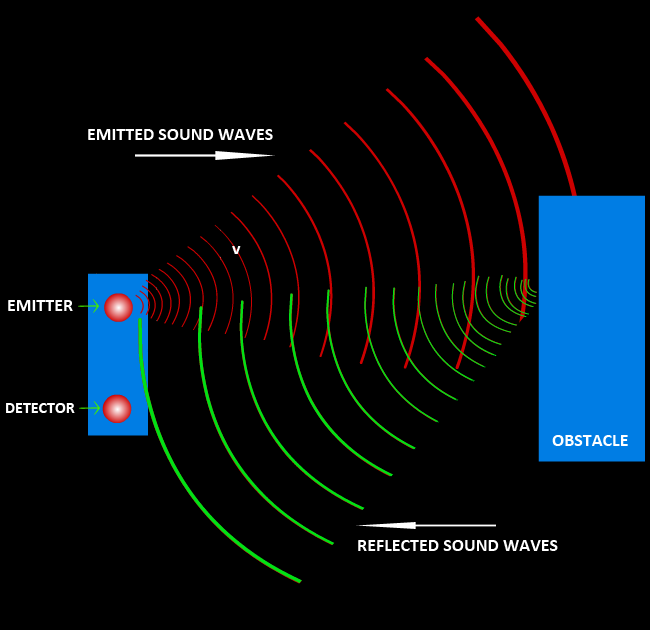
The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328

### **2.3.1 Tech specs**



|  |  |
| --- | --- |
| **MICROCONTROLLER** | ATmega328 |
| **ARCHITECTURE** | AVR |
| **OPERATING VOLTAGE** | 5 V |
| **FLASH MEMORY** | 32 KB of which 2 KB used by bootloader |
| **SRAM** | 2 KB |
| **CLOCK SPEED** | 16 MHz |
| **ANALOG IN PINS** | 8 |
| **EEPROM** | 1 KB |
| **DC CURRENT PER I/O PINS** | 40 mA (I/O Pins) |
| **INPUT VOLTAGE** | 7-12V |
| **DIGITAL I/O PINS** | 22 (6 of which are PWM) |
| **PWM OUTPUT** | 6 |
| **POWER CONSUMPTION** | 19 mA |
| **PCB SIZE** | 18 x 45 mm |
| **WEIGHT** | 7 g |
| **PRODUCT CODE** | A000005 |

## **2.4.0 Ultrasonic**

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target).

In order to calculate the distance between the sensor and the object, the sensor measures the time it takes between the emission of the sound by the transmitter to its contact with the receiver. The formula for this calculation is [D = ½ T x C](https://www.arrow.com/en/research-and-events/articles/ultrasonic-sensors-how-they-work-and-how-to-use-them-with-arduino) (where D is the distance, T is the time, and C is the speed of sound ~ 343 meters/second). For example, if a scientist set up an ultrasonic sensor aimed at a box and it took 0.025 seconds for the sound to bounce back, the distance between the ultrasonic sensor and the box would be:

|  |
| --- |
| **D = 0.5 x 0.025 x 343** |

or about 4.2875 meters.

## **نظام انذار الحرائق باستخدام حساس الغاز مع الاردوينو | GeeksValley2.5 Buzzer**

An active buzzer sensor module has a built-in oscillation circuit, thus the sound frequency is fixed. It is able to generate the sound itself. So, you can simply turn it on and off with an Arduino pin, just like the way of turning on and off a Led which is connected to Arduino board. Besides, this sensor starts beeping when it is being supplied with DC power supply.

### **2.5.1 Specification:**

* Material: ABS
* Color: Blue
* Work Voltage: 3.3-5V .
* PCB Dimension: 3.1cm\*1.3cm.
* Color: Blue
* VCC : 3.3V-5V
* GND : The Ground
* I/O : I/O Interface of SCM
* Size: 1.3cm x 3.3 cm x 1 cm

## **2.6 Vibration Motor**

Vibration motor is a DC motor in a compact size that is used to inform the users by vibrating on receiving signals. It has no sound.



Basic info:

* Working voltage:2.2V - 5V (5V recommended)
* Working current - 90 mA
* Stall current 120 mA
* Motor length - 12 mm

## **on off Kcd1 2 Position 6A 250V Rocker Switch - China 6A 250V Rocker Switch, Kcd1 2 Position Rocker Switch | Made-in-China.com2.7 Switch Button on/off**

This is not an I/O or Eye-Oh switch. It is a rocker switch labeled with the numbers One and Zero in a simplified sans-serif font. In binary arithmetic, One is “On” and Zero is “Off”. It does NOT mean “Input-Output”.

See this three position rocker switch? It is not IOII, it is signifying two possible On states, one labeled with a single One, the other with two Ones.

## **2.8 Battery 18650**

The capacity C of the battery is defined by the discharge current I and the discharge time t: C = I \* t  
I = constant discharge current  
t = duration from start of discharge to end of discharge  
Rated capacity: The rated capacity C represents the energy delivered by the battery at a discharge rate of 5 h (0.2 CA) expressed in mAh (milliampere-hours). The reference temperature is + 22 °C ± 3 °C, and the final discharge voltage is 3.0 V.

Available Capacity:

The factors that affect the available capacity are:

* Discharge rate
* End of discharge voltage
* Ambient temperature
* State of charge
* Cycle history

At the higher than nominal discharge rates the available capacity is accordingly reduced.

## **2.9 Raspberry pi 3 b+ UK**

A picture containing text, electronics, circuit

Description automatically generated

Raspberry Pi 3 Model B+ features a 1.4GHz 64-bit quad-core ARM Cortex-A53 CPU Broadcom processor. This single board computer provides dual-band 2.4GHz and 5GHz wireless LAN and Bluetooth 4.2/BLE. The Raspberry Pi 3 Model B+ offers faster Ethernet (Gigabit Ethernet over USB 2.0) and Power-over-Ethernet (PoE) capability via separate PoE HAT. This single board computer also provides improved Preboot Execution Environment (PXE) network, USB mass-storage booting, and improved thermal management.

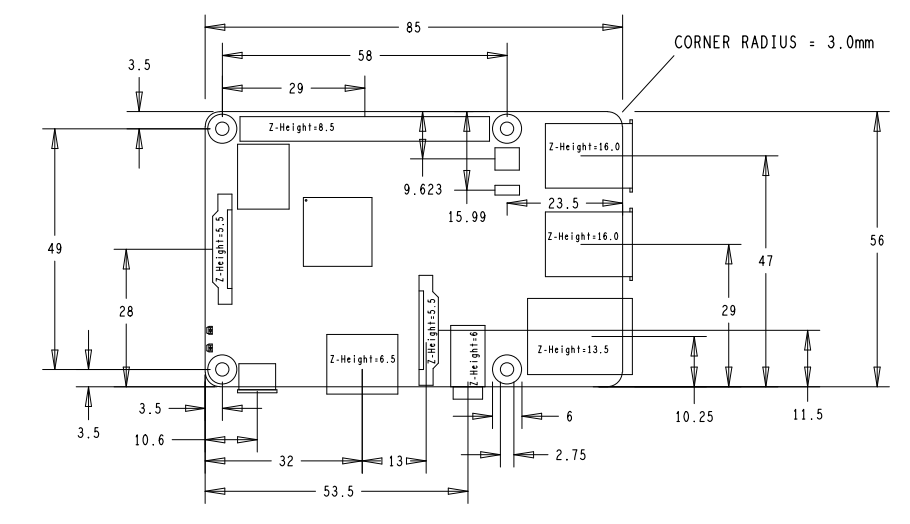
### **2.9.1 FEATURES**

* Broadcom BCM2837B0, Cortex-A53 64-bit SoC at 1.4GHz
* Dual-band 802.11ac wireless LAN
* Bluetooth 4.2
* Faster Ethernet (Gigabit Ethernet over USB 2.0)
* PoE support (with separate PoE HAT)
* Improved PXE network
* USB mass-storage booting
* Improved thermal management

### **2.9.2 SPECIFICATIONS**

* 1GB LPDDR2 SDRAM memory
* 5V/2.5A DC power input (micro USB) power supply
* Wireless connectivity:  
  + 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN and Bluetooth 4.2/BLE
  + Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps)
  + 4 x USB 2.0 ports
* Extended 40-pin Genera Purpose Input Output (GPIO) header
* Full-size HDMI video output
* 4-pole stereo audio output and composite video port
* Camera Serial Interface (CSI) camera port for connecting a Raspberry Pi camera
* Display Serial Interface (DSI) display port for connecting a Raspberry Pi touchscreen display
* Micro SD port for loading your operating system and storing data
* 0°C to 50°C operating temperature range
* 120mm x 75mm x 34mm dimension
* Weighs 75g

### **2.9.3 MECHANICAL DRAAWING**



## **2.10 Spo2**

Pulse oximetry is a noninvasive method for monitoring a person's oxygen saturation. Peripheral oxygen saturation (SpO2) readings are typically within 2% accuracy (within 4% accuracy in 95% of cases) of the more accurate (and invasive) reading of arterial oxygen saturation (SaO2) from arterial blood gas analysis. But the two are correlated well enough that the safe, convenient, noninvasive, inexpensive pulse oximetry method is valuable for measuring oxygen saturation in clinical use.

The most common approach is transmissive pulse oximetry. In this approach, a sensor device is placed on a thin part of the patient's body, usually a fingertip or earlobe, or an infant's foot. Fingertips and earlobes have higher blood flow rates than other tissues, which facilitates heat transfer.[1] The device passes two wavelengths of light through the body part to a photodetector. It measures the changing absorbance at each of the wavelengths, allowing it to determine the absorbances due to the pulsing arterial blood alone, excluding venous blood, skin, bone, muscle, fat, and (in most cases) nail polish.

Reflectance pulse oximetry is a less common alternative to transmissive pulse oximetry. This method does not require a thin section of the person's body and is therefore well suited to a universal application such as the feet, forehead, and chest, but it also has some limitations. Vasodilation and pooling of venous blood in the head due to compromised venous return to the heart can cause a combination of arterial and venous pulsations in the forehead region and lead to spurious SpO2 results. Such conditions occur while undergoing anesthesia with endotracheal intubation and mechanical ventilation or in patients in the Trendelenburg position.

### **2.10.1 Medical uses**

A pulse oximeter probe applied to a person's finger

A pulse oximeter is a medical device that indirectly monitors the oxygen saturation of a patient's blood (as opposed to measuring oxygen saturation directly through a blood sample) and changes in blood volume in the skin, producing a photo plethysmogram that may be further processed into other measurements. The pulse oximeter may be incorporated into a multiparameter patient monitor. Most monitors also display the pulse rate. Portable, battery-operated pulse oximeters are also available for transport or home blood-oxygen monitoring.

### **2.10.2 Advantages**

Pulse oximetry is particularly convenient for noninvasive continuous measurement of blood oxygen saturation. In contrast, blood gas levels must otherwise be determined in a laboratory on a drawn blood sample. Pulse oximetry is useful in any setting where a patient's oxygenation is unstable, including intensive care, operating, recovery, emergency and hospital ward settings, pilots in unpressurized aircraft, for assessment of any patient's oxygenation, and determining the effectiveness of or need for supplemental oxygen. Although a pulse oximeter is used to monitor oxygenation, it cannot determine the metabolism of oxygen, or the amount of oxygen being used by a patient. For this purpose, it is necessary to also measure carbon dioxide (CO2) levels. It is possible that it can also be used to detect abnormalities in ventilation. However, the use of a pulse oximeter to detect hypoventilation is impaired with the use of supplemental oxygen, as it is only when patients breathe room air that abnormalities in respiratory function can be detected reliably with its use. Therefore, the routine administration of supplemental oxygen may be unwarranted if the patient is able to maintain adequate oxygenation in room air, since it can result in hypoventilation going undetected.

Because of their simplicity of use and the ability to provide continuous and immediate oxygen saturation values, pulse oximeters are of critical importance in emergency medicine and are also very useful for patients with respiratory or cardiac problems,] especially COPD, or for diagnosis of some sleep disorders such as apnea and hypopnea. For patients with obstructive sleep apnea, pulse oximetry readings will be in the 70–90% range for much of the time spent attempting to sleep.

Portable battery-operated pulse oximeters are useful for pilots operating in non-pressurized aircraft above 10,000 feet (3,000 m) or 12,500 feet (3,800 m) in the U.S. where supplemental oxygen is required. Portable pulse oximeters are also useful for mountain climbers and athletes whose oxygen levels may decrease at high altitudes or with exercise. Some portable pulse oximeters employ software that charts a patient's blood oxygen and pulse, serving as a reminder to check blood oxygen levels.[citation needed]

Connectivity advancements have made it possible for patients to have their blood oxygen saturation continuously monitored without a cabled connection to a hospital monitor, without sacrificing the flow of patient data back to bedside monitors and centralized patient surveillance systems.

For patients with COVID-19, pulse oximetry helps with early detection of silent hypoxia, in which the patients still look and feel comfortable, but their SpO2 is perilously low.[5] This happens to patients either in the hospital or at home. Low SpO2 may indicate severe COVID-19-related pneumonia, requiring a ventilator.

### **2.10.3 Limitations**

Pulse oximetry solely measures hemoglobin saturation, not ventilation and is not a complete measure of respiratory sufficiency. It is not a substitute for blood gases checked in a laboratory, because it gives no indication of base deficit, carbon dioxide levels, blood pH, or bicarbonate (HCO3−) concentration. The metabolism of oxygen can be readily measured by monitoring expired CO2, but saturation figures give no information about blood oxygen content. Most of the oxygen in the blood is carried by hemoglobin; in severe anemia, the blood contains less hemoglobin, which despite being saturated cannot carry as much oxygen.[citation needed]

Because pulse oximeter devices are calibrated in healthy subjects, the accuracy is poor for critically ill patients and preterm newborns.

Erroneously low readings may be caused by hypoperfusion of the extremity being used for monitoring (often due to a limb being cold, or from vasoconstriction secondary to the use of vasopressor agents); incorrect sensor application; highly calloused skin; or movement (such as shivering), especially during hypoperfusion. To ensure accuracy, the sensor should return a steady pulse and/or pulse waveform. Pulse oximetry technologies differ in their abilities to provide accurate data during conditions of motion and low perfusion.

Obesity, hypotension (low blood pressure), and some hemoglobin variants can reduce the accuracy of the results. Some home pulse oximeters have low sampling rates which can significantly underestimate dips in blood oxygen levels. The accuracy of pulse oximetry deteriorates considerably for readings below 80%.

Pulse oximetry also is not a complete measure of circulatory oxygen sufficiency. If there is insufficient blood flow or insufficient hemoglobin in the blood (anemia), tissues can suffer hypoxia despite high arterial oxygen saturation.

Since pulse oximetry measures only the percentage of bound hemoglobin, a falsely high or falsely low reading will occur when hemoglobin binds to something other than oxygen:

Hemoglobin has a higher affinity to carbon monoxide than it does to oxygen, and a high reading may occur despite the patient's actually being hypoxemic. In cases of carbon monoxide poisoning, this inaccuracy may delay the recognition of hypoxia (low cellular oxygen level).

Cyanide poisoning gives a high reading because it reduces oxygen extraction from arterial blood. In this case, the reading is not false, as arterial blood oxygen is indeed high in early cyanide poisoning.[clarification needed]

Methemoglobinemia characteristically causes pulse oximetry readings in the mid-80s.

COPD [especially chronic bronchitis] may cause false readings.

A noninvasive method that allows continuous measurement of the dyshemoglobins is the pulse CO-oximeter, which was built in 2005 by Masimo. By using additional wavelengths, it provides clinicians a way to measure the dyshemoglobins, carboxyhemoglobin, and methemoglobin along with total hemoglobin.

Research has suggested that error rates in common pulse oximeter devices may be higher for adults with dark skin color, leading to claims of encoding systemic racism in countries with multi-racial populations such as the United States. Pulse oximetry is used for the screening of sleep apnea and other types of sleep-disordered breathing which in the United States are conditions more prevalent among minorities.

### **2.10.4 Equipment**

In addition to pulse oximeters for professional use, many inexpensive "consumer" models are available. Opinions vary about the reliability of consumer oximeters; a typical comment is "The research data on home monitors has been mixed, but they tend to be accurate within a few percentage points”. Some smart watches with activity tracking incorporate an oximeter function. An article on such devices, in the context of diagnosing COVID-19 infection, quoted João Paulo Cunha of the University of Porto, Portugal: "these sensors are not precise, that's the main limitation ... the ones that you wear are only for the consumer level, not for the clinical level”. Pulse oximeters used for diagnosis of conditions such as COVID-19 should be Class IIB medical grade oximeters. Class IIB oximeters can be used on patients of all skin colors, low pigmentation and in the presence of motion.[citation needed] When a pulse oximeter is shared between two patients, it should be either cleaned with alcohol wipes after each use or a disposable probe or finger cover to be used to prevent cross-infection.

According to a report by I Data Research the US pulse oximetry monitoring market for equipment and sensors was over $700 million in 2011.

### **2.10.5 Mobile apps**

Mobile app pulse oximeters use the flashlight and the camera of the phone, instead of infrared light used in conventional pulse oximeters. However, apps don't generate as accurate readings because the camera can't measure the light reflection at two wavelengths, so the oxygen saturation readings that are obtained through an app on a smartphone are inconsistent for clinical use. At least one study has suggested these are not reliable relative to clinical pulse oximeters.

## **2.11 Water sensor**

The Water Level Depth Detection Sensor for Arduino has Operating voltage DC3-5V and Operating current less than 20mA. The Sensor is the Analog type which produces analog output signals according to the water pressure with

The Water Level Sensor is an easy-to-use and cost-effective with high level/drop recognition sensor by having a series of parallel wires exposed traces measure droplets/water volume in order to determine the water level.

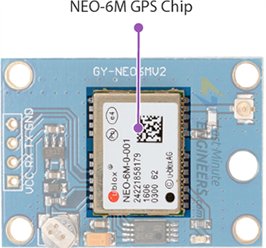
Easy to complete water to analog signal conversion and output analog values can be directly read Arduino development board to achieve the level alarm effect.

### **2.11.1 Specifications**

This sensor is supplied to 5V or 3.3V on VCC and GND pins. The pin S will give us an analog value between VCC and GND. So we will use the S pin as analog input connecting Arduino, the value read will be higher depending on the sensor surface is covered with water. This is because the water acts as a conductor, given that the water we use in our deposits not be pure water (H2O), since if water is nonconductive. But rarely we will use this type of sensors to measure the water level in a tank of pure water.

|  |  |
| --- | --- |
| Power supply | 3,3V ~ 5V |
| Current | < 20mA |
| Humidity sensitivity range | 10% ~ 90% |
| Pinout | +: VCC -: GND S: Analog output |

## **2.12 GPS**

Module that can track up to 22 satellites and identifies locations anywhere in the world. It may serve as a great launch pad for anyone looking to get into the world of GPS.

They are low power (suitable for battery powered devices), inexpensive, easy to interface with and are insanely popular among hobbyists.

### **2.12.1 Hardware Overview of NEO-6M GPS Module**

NEO-6M GPS Chip

At the heart of the module is a NEO-6M GPS chip from u-blox. The chip measures less than the size of a postage stamp but packs a surprising amount of features into its little frame.

It can track up to 22 satellites on 50 channels and achieves the industry’s highest level of sensitivity i.e. -161 dB tracking, while consuming only 45mA supply current.

Unlike other GPS modules, it can do up to 5 location updates a second with 2.5m Horizontal position accuracy. The u-blox 6 positioning engine also boasts a Time-To-First-Fix (TTFF) of under 1 second.

One of the best features the chip provides is Power Save Mode(PSM). It allows a reduction in system power consumption by selectively switching parts of the receiver ON and OFF. This dramatically reduces power consumption of the module to just 11mA making it suitable for power sensitive applications like GPS wristwatch.

The necessary data pins of NEO-6M GPS chip are broken out to a 0.1″ pitch headers. This includes pins required for communication with a microcontroller over UART. The module supports baud rate from 4800bps to 230400bps with default baud of 9600.

### **2.12.2 specifications:**

|  |  |
| --- | --- |
| Receiver Type | 50 channels, GPS L1(1575.42Mhz) |
| Horizontal Position Accuracy | 2.5m |
| Navigation Update Rate | 1HZ (5Hz maximum) |
| Capture Time | Cool start: 27sHot start: 1s |
| Navigation Sensitivity | -161dBm |
| Communication Protocol | NMEA, UBX Binary, RTCM |
| Serial Baud Rate | 4800-230400 (default 9600) |
| Operating Temperature | -40°C ~ 85°C |
| Operating Voltage | 2.7V ~ 3.6V |
| Operating Current | 45mA |
| TXD/RXD Impedance | 510Ω |

### **2.12.NEO-6M GPS Module - Position Fix LED Indicator3 Position Fix LED Indicator**

There is an LED on the NEO-6M GPS Module which indicates the status of Position Fix. It’ll blink at various rates depending on what state it’s in:

* No Blinking – It’s searching for satellites.
* Blink every 1s – Position Fix is found(The module can see enough satellites).

### **2.12.NEO-6M GPS Module - 3.3V Voltage Regulator4 3.3V LDO Regulator**

The operating voltage of the NEO-6M chip is from 2.7 to 3.6V. But the good news is that, the module comes with MIC5205 ultra-low dropout 3V3 regulator from MICREL.

The logic pins are also 5-volt tolerant, so we can easily connect it to an Arduino or any 5V logic microcontroller without using any logic level converter.

### **NEO-6M GPS Module - Battery and EEPROM2.12.5 Battery & EEPROM**

The module is equipped with an HK24C32 two wire serial EEPROM. It is 4KB in size and connected to the NEO-6M chip via I2C.

The module also contains a rechargeable button battery which acts as a super-capacitor.

An EEPROM together with battery helps retain the battery backed RAM (BBR). The BBR contains clock data, latest position data(GNSS orbit data) and module configuration. But it’s not meant for permanent data storage.

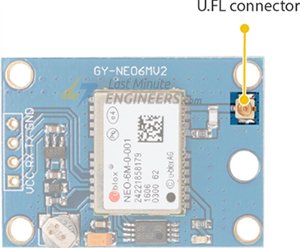
As the battery retains clock and last position, time to first fix (TTFF) significantly reduces to 1s. This allows much faster position locks.

Without the battery the GPS always cold-start so the initial GPS lock takes more time.

The battery is automatically charged when power is applied and maintains data for up to two weeks without power.

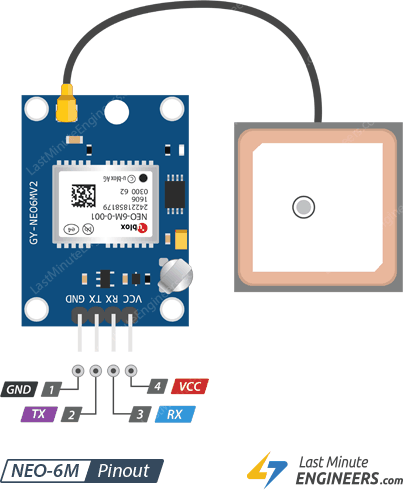
### **2.12.NEO-6M Patch Antenna6 Antenna**

An antenna is required to use the module for any kind of communication. So, the module comes with a patch antenna having -161 dBm sensitivity.



You can snap-fit this antenna to small U.FL connector located on the module.

Patch antenna is great for most projects. But if you want to achieve more sensitivity or put your module inside a metal case, you can also snap on any 3V active GPS antenna via the U.FL connector.



### **2.12.7 NEO-6M GPS Module Pinout**

The NEO-6M GPS module has total 4 pins that interface it to the outside world. The connections are as follows:

**GND** is the Ground Pin and needs to be connected to GND pin on the Arduino.

**TxD (Transmitter)** pin is used for serial communication.

**RxD (Receiver)** pin is used for serial communication.

**VCC** supplies power for the module. You can directly connect it to the 5V pin on the Arduino.

### **2.12.8 Connections:**

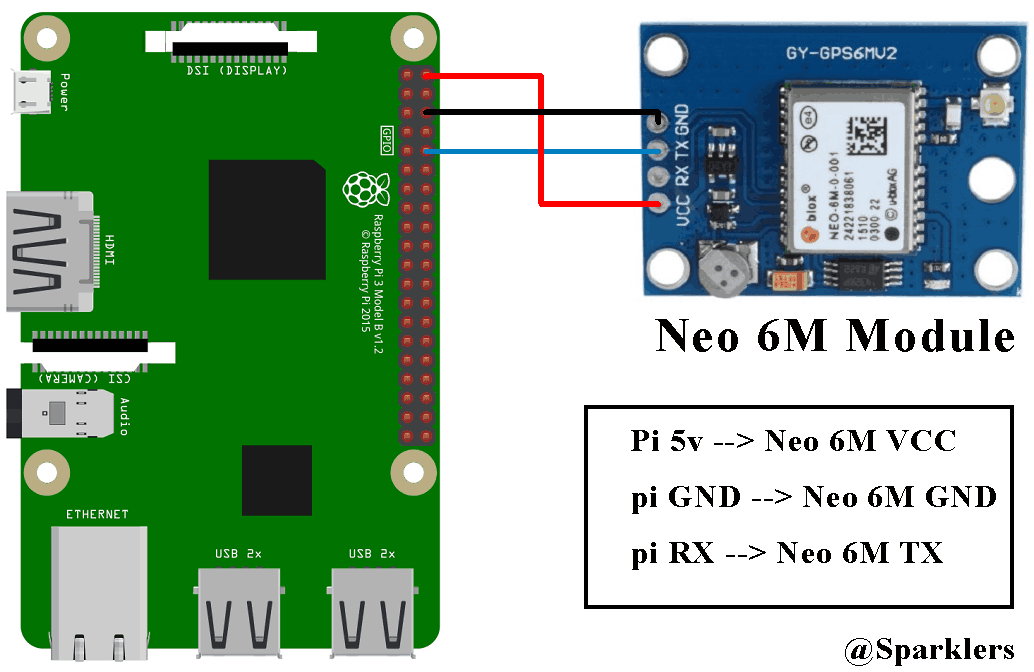
Here we only need to connect the Neo 6M module with Raspberry Pi which is quite easy. The connections are shown below:

Neo 6M VCC -----> Raspberry pi 5v

Neo 6M GND -----> Raspberry pi GND

Neo 6M RX -----> Raspberry pi TX (gpio 14) //Not required in our case

Neo 6M TX -----> Raspberry pi RX (gpio 15)



So we need the VCC of Neo 6M to be connected with 5v of Raspberry pi, GND of Neo 6M with GND of Raspberry pi and TX of Neo 6M with RX of Raspberry Pi so that the GPS module can send data to raspberry pi through the serial connection. That’s all about the hardware part and now let’s go for the software part.

### **2.12.9 Software Part:**

**Getting data from the GPS module:**

1. Install the lastest Rasbian OS in a memory card. For details visit [www.raspberrypi.org/documentation/installation/installing-images/](https://www.raspberrypi.org/documentation/installation/installing-images/).
2. Insert the memory card into raspberry pi and power it up.
3. Now here we need to modify few things. At first we need to edit the /boot/config.txt file. Now you need to open this file in any text editor. Here I am using nano:

sudo nano /boot/config.txt

At the end of the file add the follwing lines:

dtparam=spi=on

dtoverlay=pi3-disable-bt

core\_freq=250

enable\_uart=1

force\_turbo=1

It will look something like this:Text

Description automatically generated

Now save this by typing ctrl +x, then type y and press enter.

1. Raspbian uses the UART as a serial console and so we need to turn off that functionality. To do so we need to change the /boot/cmdline.txt file. For safety before editing the file make a backup of that using the following command:

sudo cp /boot/cmdline.txt /boot/cmdline\_backup.txt

sudo nano /boot/cmdline.txt

dwc\_otg.lpm\_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes rootwait quiet splash plymouth.ignore-serial-consoles

sudo reboot

sudo cat /dev/ttyAMA0

Now you will see a lots of data like shown in the below image. That basically means that its working. To stop this type Ctrl + c .

Text

Description automatically generated

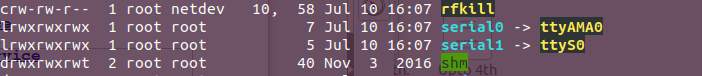
### **2.12.10 Setup for writing the Python Code:**

1. Now before we write the python code to get the GPS data we need set up few things again. By default the Raspberry Pi uses serial port for this “console” login so if we want to use the serial port to get data from the GPS module we need to disable the console login. Now there are two serial ports in Raspberry pi 3: serial0 and serial1. But in between them serial0 will point to GPIO pins 14 and 15, so we have to use serial 0. Now to see which port is connected with serial0 use the following command:

ls -l /dev

There are two possible outputs:

* **If your output looks like this:**



As you can see serial0 is linked with ttyAMA0. So to disable the console you need to use the following commands:

sudo systemctl stop serial-getty@ttyAMA0.service

sudo systemctl disable serial-getty@ttyAMA0.service

* **But if your output looks like this:**



That means serial0 is linked with ttyS0. So to disable the console you need to use the following commands:

sudo systemctl stop serial-getty@ttyS0.service

sudo systemctl disable serial-getty@ttyS0.service

### **2.12.11 Python Code:**

1. Now we need to install a python library:

pip install pynmea2

1. Now finally we are ready to write the code:

import serial

import time

import string

import pynmea2

while True:

port="/dev/ttyAMA0"

ser=serial.Serial(port, baudrate=9600, timeout=0.5)

dataout = pynmea2.NMEAStreamReader()

newdata=ser.readline()

if newdata[0:6] == "$GPRMC":

newmsg=pynmea2.parse(newdata)

lat=newmsg.latitude

lng=newmsg.longitude

gps = "Latitude=" + str(lat) + "and Longitude=" + str(lng)

print(gps)

If you run this python code you will see a output like this:

Text

Description automatically generated

I am not moving this device that’s why it’s giving me the same GPS location. But as you can see it’s working properly.

# Chapter 3

## **3.1 Object Detection**

Before Object recognition has developed rapidly, starting with the deep learning–based convolutional neural network (CNN) technique [5] that drew attention at the ImageNet 2012 competition. The CNN, however, was accurate with object classification, but it was difficult to determine where inside the image the object was located. Subsequently, the model for solving this problem was the region-based consolidated neural network (R-CNN), which uses a linear regression method. However, due to the slow speed of the R-CNN, Fast R-CNN was developed. It utilizes a deep learning technique to not only classify the object but also to find the area the object is located in. Nonetheless, there was a limit in that the above model's object recognition processing speed was insufficient for real-time object recognition. Since then, You Only Look Once (YOLO), which comprises all the processes of object recognition as a deep learning network, has emerged, and technologies with fast detection speeds, such as Single Shot MultiBox Detector (SSD), have been developed. YOLO estimates the type and location of objects using regression inference on the problem of area selection and classification. On the other hand, SSD does not create candidate areas separately, but recognizes objects using a feature map of various sizes. Since it does not generate candidate areas, it is faster to train than the Faster R-CNN and is more accurate than YOLO because it uses different sizes of feature map.

## **3.2 Image Processing**

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Image processing basically includes the following three steps:

* Importing the image to the system.
* Analyzing and manipulating the image.
* Output in which result can be altered image or report that is based on image analysis.

## **3.3 Yolo v3**

In this project we have used YOLO v3 which is faster than the prior version. It works three times faster, at 320 ×320 YOLOv3 runs 22ms at 28.2 map. It has a similar performance but 3.8× faster. The most notable characteristic of v3 is that it makes 3 distinct scales of detections. YOLO v3 is a fully convolutional neural network and it generates its resultant output by applying a 1 x 1 kernel to a feature map. In YOLO v3, the recognition is obtained by implementing 1 x 1 detection kernels to three-size feature maps at three different regions in the network.

Within each boundary the network predicts 4 coordinates tx,ty,tw,th. Whereas if a cell is offset in the upper left corner of the image by (cx,cy) and prior bounding boxes has pw, ph width and height respectively then the prediction is done .

## **3.4 OpenCV**

Techniques for Object Recognition in Images and Multi- Object Detection and segmentation is the most significant and testing central undertaking of Computer vision. It is a basic part in numerous applications, for example, image search, scene understanding, and so far. However it is as yet an open issue because of the assortment and multifaceted nature of item classes and foundations. The most effortless approach to identify and fragment an item from a picture is the shading based techniques. The team and the foundation ought to have a critical shading distinction so as to effectively portion objects utilizing shading based strategies. OpenCV usually captures images and videos in 8-bit, unsigned integer, BGR format. In other words, captured images can be considered as 3 matrices; BLUE, GREEN RED (hence the name BGR) with integer values ranging from 0 to 255.In genuine pictures, these pixels are little to such an extent that the natural eye can't separate.

## **3.5 Hardware**

We are using Raspberry Pi 3 Model B+ which has Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz and 1GB LPDDR2 SDRAM with class 10 micro SD card where the OS and project is stored, the reason behind we are using the class 10 memory card is that it helps to retrieve data at higher speed so that the project can take lesser time to execute The camera used in our system is Raspberry Pi Camera Module v2 which has Sony IMX219 8megapixel sensor to feed images at 30 frame per second to this trained model, Ultrasonic Distance Sensor HC-SR04.

## **3.6 Working**

We are using Python3 for this project, the camera is initialized by using OpenCV library and the camera starts capturing frames with the rate of 30 frames per second to the algorithm. Then the system uses YOLO v3 which is trained on the COCO dataset and Dark Neural Network (DNN) to identify the object kept before the user.

The object identified is later converted to an audio segment using gTTs which is a python library. The audio segment is the output of our system that gives the spatial location and name of the object to the person. Now by using this information the person can have a visualization of the objects around him. The proposed system will even protect the person from colliding to the objects around will secure him from injuries & Ultrasonic sensor will detect object & give the distance between came.

## **3.7 Block Diagram & Proposed Platform**

Diagram

Description automatically generated

## **3.8 Result and Experiment**

The proposed system will be able to identify the object in front of the camera and will later on convert it into mp3 using gTTS. The proposed system is very low cost, FIG 3 shows the whole system which is a Raspberry pi 3b+, Bluetooth headphones and a power bank in order to provide power to the raspberry pi. The hardware implementation is shown in the given figure.

## **3.9 FUTURE SCOPE**

This project is for the blind people who are incapable to see this colorful and beautiful world, our initiative will support them to have a better life. By this project one will be able to understand what object is present in front of him and by continuous research and development our team will be able improve this product by feeding more data to the Deep Learning algorithm by which the accuracy of the model will increase as well as the power of the algorithm to recognize more objects will increase. The object recognition system can be applied in the area of surveillance system, face recognition, fault detection, character recognition etc. The objective of this thesis is to develop an object recognition system to recognize the 2D and 3D objects in the image. The performance of the object recognition system depends on the features used and the classifier employed for recognition. This research work attempts to propose a novel feature extraction method for extracting global features and obtaining local features from the region of interest.

## **3.9 Code**

### **3.9.1Test camera**

import cv2

cap = cv2.VideoCapture(0)

# Check if the webcam is opened correctly

if not cap.isOpened():

    raise IOError("Cannot open webcam")

while True:

    ret, frame = cap.read()

    frame = cv2.resize(frame, None, fx=0.5, fy=0.5, interpolation=cv2.INTER\_AREA)

    cv2.imshow('Input', frame)

    c = cv2.waitKey(1)

    if c == 27:

        break

cap.release()

cv2.destroyAllWindows()

### **3.9.2 Text to Speech**

from gtts import gTTS

import vlc

# import Os module to start the audio file

import os

mytext = 'temperature is high please stop and check cooling system'

# Language we want to use

language = 'en'

myobj = gTTS(text=mytext, lang=language, slow=False)

myobj.save("temp.mp3")

# Play the converted file

#os.system("start temp.mp3")

media = vlc.MediaPlayer("temp.mp3")

# start playing video

media.play()

### **3.9.3 real time object detection with Text to Speech**

# USAGE

# python3 object\_detection\_with\_Text\_to\_Speech.py --prototxt MobileNetSSD\_deploy.prototxt.txt --model MobileNetSSD\_deploy.caffemodel --source webcam

# import the necessary packages

from imutils.video import VideoStream

import numpy as np

import sys

import argparse

import imutils

import time

import cv2

import vlc

from translate import Translator

from gtts import gTTS

import os

last = 0

# construct the argument parse and parse the arguments

ap = argparse.ArgumentParser()

# ap.add\_argument("--prototxt", required=True,

#   help="path to Caffe 'deploy' prototxt file")

# ap.add\_argument("--model", required=True,

#   help="path to Caffe pre-trained model")

# ap.add\_argument("--source", required=True,

#   help="Source of video stream (webcam/host)")

ap.add\_argument("-c", "--confidence", type=float, default=0.2,

    help="minimum probability to filter weak detections")

args = vars(ap.parse\_args())

# initialize the list of class labels MobileNet SSD was trained to

# detect, then generate a set of bounding box colors for each class

CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",

    "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",

    "dog", "horse", "motorbike", "person", "pottedplant", "sheep",

    "sofa", "train", "tvmonitor"]

COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))

# load our serialized model from disk

print("[INFO] loading model...")

net = cv2.dnn.readNetFromCaffe("MobileNetSSD\_deploy.prototxt.txt", "MobileNetSSD\_deploy.caffemodel")

# initialize the video stream, allow the cammera sensor to warmup,

print("[INFO] starting video stream...")

vs = cv2.VideoCapture(0)

time.sleep(2.0)

detected\_objects = []

# loop over the frames from the video stream

while True:

    # grab the frame from the threaded video stream and resize it

    # to have a maximum width of 400 pixels

    ret, frame = vs.read()

    frame = imutils.resize(frame, width=800)

    # grab the frame dimensions and convert it to a blob

    (h, w) = frame.shape[:2]

    blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),

        0.007843, (300, 300), 127.5)

    # pass the blob through the network and obtain the detections and

    # predictions

    net.setInput(blob)

    detections = net.forward()

    # loop over the detections

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

        # extract the confidence (i.e., probability) associated with

        # the prediction

        confidence = detections[0, 0, i, 2]

        # filter out weak detections by ensuring the `confidence` is

        # greater than the minimum confidence

        if confidence > args["confidence"]:

            # extract the index of the class label from the

            # `detections`, then compute the (x, y)-coordinates of

            # the bounding box for the object

            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")

            # draw the prediction on the frame

            label = "{}".format(CLASSES[idx],

                confidence \* 100)

            detected\_objects.append(label)

            cv2.rectangle(frame, (startX, startY), (endX, endY),

                COLORS[idx], 2)

            y = startY - 15 if startY - 15 > 15 else startY + 15

            cv2.putText(frame, label, (startX, y),

                cv2.FONT\_HERSHEY\_SIMPLEX, 0.5, COLORS[idx], 2)

            if (last != label):

                print (label)

                last = label

                translator= Translator(from\_lang="english",to\_lang="arabic")

                translation = translator.translate(label)

                print(translation)

                language = 'en'

                myobj = gTTS(text=translation, lang=language, slow=False)

                myobj.save("output.mp3")

                # Play the converted file

                #os.system("start output.mp3")

                # creating vlc media player object

                media = vlc.MediaPlayer("output.mp3")

                # start playing video

                media.play()

    # show the output frame

    #cv2.imshow("Frame", frame)

    key = cv2.waitKey(1) & 0xFF

    # if the `q` key was pressed, break from the loop

    if key == ord("q"):

        break

# do a bit of cleanup

cv2.destroyAllWindows()

# Chapter 4

## **4.1.1 Web Site**

A Web site is a related collection of World Wide Web (WWW) files that includes a beginning file called a home page. A company or an individual tells you how to get to their Web site by giving you the address of their home page. From the home page, you can get to all the other pages on their site. For example, the Web site for IBM has the home page address of http://www.ibm.com. (The home page address actually includes a specific file name like *index.html* but, as in IBM's case, when a standard default name is set up, users don't have to enter the file name.) IBM's home page address leads to thousands of pages. (But a Web site can also be just a few pages.)

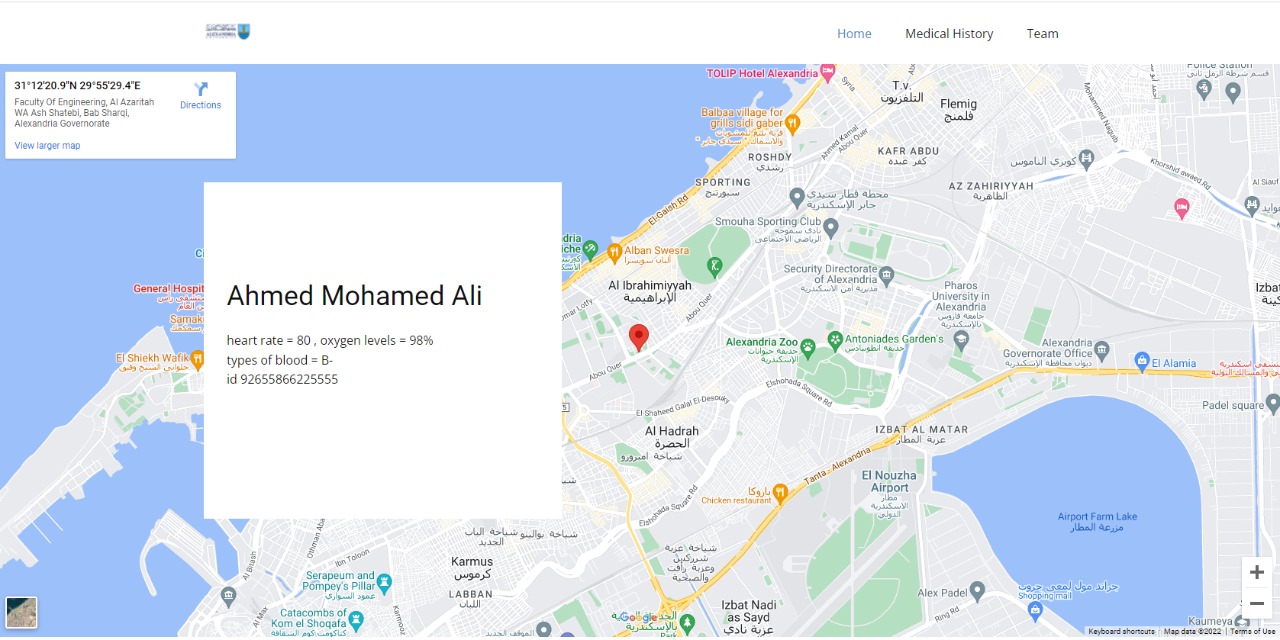
Since *site* implies a geographic place, a Web site can be confused with a Web server. A server is a computer that holds the files for one or more sites. A very large Web site may be spread over a number of servers in different geographic locations. IBM is a good example; its Web site consists of thousands of files spread out over many servers in world-wide locations. But a more typical example is probably the site you are looking at, whatis.com. We reside on a commercial space provider's server with a number of other sites that have nothing to do with Internet glossaries.

A synonym and less frequently used term for Web site is "Web presence." That term seems to better express the idea that a site is not tied to specific geographic location, but is "somewhere in cyberspace." However, "Web site" seems to be used much more frequently.

You can have multiple Web sites that cross-link to files on each other’s' sites or even share the same files.

### **4.1.2 Website Design**

Obtaining and monitoring user information and their health condition



Graphical user interface, text, application

Description automatically generated

### **4.1.3 Code**

## **4.2 Google Firebase**

Firebase is a mobile and web application development platform. Firebase is made up of complementary features that developers can mix-and-match to fit their needs. The team is based in San Francisco and Mountain View, California. The company was founded in 2011 by Andrew Lee and James Templin. Firebase's initial product was a Realtime database, which provides an API that allows developers to store and sync data across multiple clients. Over time, it has expanded its product line to become a full suite for app development. The company was acquired by Google in October 2014 and a significant number of new features were featured in May 2016 at Google I/O.

### **4.2.1 About Google Firebase**

Firebase evolved from Envolve, a prior startup founded by Templin and Lee in 2011. Envolve provided developers an API that enables the integration of online chat functionality into their websites. After releasing the chat service, Templin and Lee found that the service was being used to pass application data that wasn't chat messages. Developers were using Envolve to sync application data such as game state in Realtime across their users. Templin and Lee decided to separate the chat system and the real-time architecture that powered it, founding Firebase as a separate company in April 2012.

Firebase raised $1.4 million in seed funding in May 2012 from Flybridge Capital Partners, Greylock Partners, NEA and others. The company further raised $5.6 million in Series A funding from Union Square Ventures and Flybridge Capital Partners in June 2013. On October 21, 2014, Firebase announced it had been acquired by Google for an undisclosed amount. On October 13, 2015, Google acquired Divshot to merge it with the Firebase team, for an undisclosed amount. Before the acquisition, Divshot had raised $1.18 million in two rounds of funding, according to TechCrunch. Since the acquisition, Firebase has grown inside Google and expanded their services to become a unified platform for mobile developers. Firebase now integrates with various other Google services to offer broader products and scale for developers. The vision of Firebase stays the same and aim to help developers build better apps and grow successful businesses.

Code

Upload to firebase Realtime database

import RPi.GPIO as GPIO

from time import sleep

from firebase import firebase

import Adafruit\_DHT

firebase = firebase.FirebaseApplication('https://YOUR\_FIREBASE\_URL.firebaseio.com/', None)

firebase.put("/", "/lat", "0.00")

firebase.put("/", "/long", "0.00")

                firebase.put("/","/lat",str\_lat)

                firebase.put("/","/long",str\_long)

        else:

                print('Failed to get reading. Try again!')

        sleep(5)

## **4.3 USB modem**

### **4.3.1 Connecting 3G / 4G USB modem to Raspberry Pi 3**

In this tutorial we are attaching an 3G/4G USB modem to Raspberry Pi3 and setting everything up

for automatic connection during startup.

The modems used are: 3G – ZTE MF668 (network unlocked, but not essential)

4G – Huawei E3372

Pre-requsite:

Remove any additional devices you may have connected to USB ports – Other than a keyboard.

As always we want to make sure system is up to date. So we enter:

sudo apt-get update

(and press ENTER)

Now we make sure we don’t have network-manager or ModemManager installed.

Both seem to be incompatible with above mentioned devices and cause problems with connection.

So in order to check or remove those you can enter in Konsole:

sudo apt-get remove network-manager

and

sudo apt-get remove ModemManager

(and press ENTER)

A message indicating something about “not found” would indicate the program wasn’t installed in

the first place.

Now we can move to actually installing modem.

As many USB modems can also work as USB storage device we need to ensure that device is only

operated and seen as a modem by the system. We achieve this by installing USB Modeswitch with

following command:

sudo apt-get install ppp usb-modeswitch

Luckily for us it appears that Modeswitch automatically installs with correct settings for our modem

and no further settings are required to keep modem in modem mode.

3G Modem:

For those wanting to use the 4G – Huawei E3372 modem you can skip this step and start using it

right away. - See section 4G Modem below.

I’ve spent many days trying to make this work only to find other software was causing a conflict

which prevented successful operation of modem. Hence important to ensure you haven’t already

installed any other software. If so, please remove such software prior to proceeding.

Now we need an oldie but good software to make this work. - Sakis3g

it appears to get much harder now to find an active link for it. The below mentioned link still

worked for me in October 2019.

Enter following commands and hit ENTER after each line:

sudo apt-get install ppp

wget "http://raspberry-at-home.com/files/sakis3g.tar.gz"

sudo mkdir /usr/bin/modem3g

sudo chmod 777 /usr/bin/modem3g

sudo cp sakis3g.tar.gz /usr/bin/modem3g

cd /usr/bin/modem3g

sudo tar -zxvf sakis3g.tar.gz

sudo chmod +x sakis3g

Now we can start sakis3g in menu mode with:

/usr/bin/modem3g/sakis3g --interactive "menu" "console"

( may need “sudo” in front of this line)

In the menu we should see “1. Connect wit 3G”

Hit ENTER and it will ask for an interface number. Now strangely enough it doesn’t appear to

matter much which interface number you select, in my case both worked. But you can try the

different options and see if anything changes.

Hit ENTER again and a message will indicate modem is being prepared.

After a few seconds a menu will pop up again asking which connection should be used. Now this

depends entirely on your network provider. In my case Optus. So I select the “Optus 3G

(CONNECT)” option.

-The word in the bracket is rather important as it refers toy your nework provider’s APN. Make sure

this is correct. (Check on your provider’s website regarding APN settings if necessary)

Now highlight the appropriate option and hit ENTER.

A message will appear “Fixing connection”. This step might take a full minute or so and then you

will get a message “MF668 connected to “YOUR PROVIDER” “

Hit ENTER and now you can select the “Exit” option in the menu.

- Congratulations, you are now connected to 3G network. -

Making it auto start during startup:

Most of us prefer not having to start connection manually, especially when running Pi in headless

mode.

We can achieve this by using rc.local

Type:

sudo nano etc rc.local

You should see some remarks at the top of the file (remarks are starting with #).

Scroll down before you get to the “exit 0” line. Hit ENTER to create a new line – this has to be

above the “exit 0”.

Type:

sudo usr/bin/modem3g/sakis3g connect USBINTERFACE=”2” APN”CONNECT”

Make sure you use your network provider’s APN instead of “CONNECT” and also the interface

which was previously successfully used. (1,2,3,or4)

Hit CTR & X and then confirm saving of file by pressing Y.

Now reboot system and check if connection is starting automatically:

sudo reboot now

4G Modem

\*Fortunately this is much easier as it should work out of the box providing you’ve followed the

steps under “Pre-requisite”

This device shows up on a desktop computer (at least on Ubuntu) as an Ethernet connection. So I

did connect it first to my desktop computer in order to check for correct function and also noted the

APN in the modem menu by typing: http:// 192.168.8.1 into a browser. Followed by clicking on

“Settings” and then “Profile Management”

(Check instructions provided with your USB-modem.)

However, did notice on the Pi it can at times take a minute or so to connect. - See status light on

USB modem.