



IoT Social Distancing And Monitoring Robot

Oğuzhan Karahan¹, Jordi Hako Nintcheu²

^{1,2} Kocaeli University, Department of Electronics and Telecommunications Engineering, Kocaeli, Turkey

¹oguzhan.karahan@kocaeli.edu.tr, ²jordihako@gmail.com

Abstract

Coronavirus has affected a lot of people around the world, with the rate of infected people still increasing. This virus is transmitted between people through direct, indirect, or close contact with infected people. To help prevent the social spread of COVID-19, this paper presents a new smart social distancing monitoring and managing robotic system that allows individuals to keep social distances between others in indoor and outdoor environments, avoiding exposure to COVID-19 and slowing its spread locally and across territories. The proposed robotic system consists of a module to alert people not respecting social distancing and a module to send information about people not respecting social distancing to an online database from where it is displayed on a graphical user interface(GUI) for several review purposes.

Keywords: Social distancing, monitoring, managing, IOT robot.

1. Introduction

COVID-19 belongs to a big family of viruses that normally causes moderate to mean upper-respiratory-tract ailments and irritations. It was reported for the first time in Wuhan, China, at the end of December 2020. The World Health Organisation (WHO) has declared COVID-19 a pandemic, and a global coordinated effort is required to stop the spread of the virus. The transmission of COVID-19 remains unclear, though evidence from other viruses indicates that the disease may spread through direct or indirect contact with an infected person.

According to (İLKBABAR & SUNGU, 2020) the idea of letting machines do most of the decision makings had increased, leading to the development of new technologies, and the decision-making capabilities of machines.

To further emphasize the impotence which machine learning had in the handling of the COVID-19 pandemic, (KAYA, 2021) shows us in their article on the use of machine learning algorithms such as the Flower Pollination Algorithm(FPA), how well machine learning could be used to help examine, analyze and monitor COVID cases over territories.

This research could be taken further as shown in (Nehir ÖZNUR MUZ et al., 2020) where the author further illustrates cases of the use of machine learning and artificial intelligence in combating the Coronavirus

During the ongoing COVID-19 disaster, the Internet of Things (IoT) also has played a significant role in a diverse range of healthcare applications. In general, IoT networks consist of several low-cost, small-size, and low-power consumption devices that can be attached to any person or be embedded in any object.

Several IoT technologies have been used to help cater to the needs of the COVID pandemic. Such techniques include the Blended Learning model with IoT-based technology whose efficiency (SİRİPONGDEE et al., 2020) reviewed in their article.

Social distancing is of the utmost importance for people who are at a higher risk of contracting COVID-19. Social distancing is the maintenance of a distance(the presumed safe distance) of at least 1.5m within two or more people in indoor and outdoor spaces to minimize the spread of the virus. It also limits close contact with others in outdoor and indoor spaces, as people can spread the virus before they know that they are sick.

Recently, social distancing was proven to be an effective practice in minimizing the spreading of COVID-19. Therefore, social distancing has prompted IOT engineers, researchers, and developers to find technological solutions to fight against the spread of the COVID-19 virus. Several mobile applications and IoT devices have been developed recently to work against the spread of COVID-19.

Because of the nature of the virus and the high spread rate, either indoor or outdoor, when human contact exceeds the predefined social distance space, this work presents a robotic system that will assure and monitor

the social distance between individuals during runtime. This document explains the experiment which involves the building of the hardware and software structure of the IoT robot. This includes technological aspects like the designing and building of a PCB(Containing interactives) which will serve as a contact point with the outer world, a website for reporting cases of people at risk, and an object detector model for identifying people and calculating distances. This robotic system will be responsible for providing information about cases of social distancing not respected and alerting.

The proposed system was experimentally studied to test the functionality and usability of the proposed device. Our robot has been shown to work as expected except for the fact that it doesn't stop exactly at the required position. We, later on, found that this issue was due to hardware constraints and could be solved by investing more in hardware infrastructure.

2. Materials and Methods

The first stage in the development of this project required the design of a PCB, that would merge all the different interactives of the robot. These interactives are an ultrasonic sensor to detect objects that are extremely close to the robot and prevent it from capturing the video stream correctly, a motor driver circuit for the stepper motor to control the robot's movements, and an alarm system to alert people when social distancing isn't respected.

We used the raspberry pi model 4B on this project to handle all our programming needs. The raspberry pi's wifi module is used to handle the IoT connections(That is the connection between the online website and the local robotic system). The pi's camera module is used for shooting, video recording, and object recognition. The raspberry pi's processors are used for image processing, object recognition, distance calculation, motor control, and alarming. Plugged into the raspberry pi's GPIO pins is the whole PCB containing the different interactives. The overall setup is encased in a two-piece mechanical case as a head and a tail.

The second step was the design of the above-mentioned mechanical case, which is made up of 2 pieces. The lower piece serves as a support stand for the whole robot, while the upper piece serves as a rotating head, containing all the robot's circuitry. This design was done using an Ultimaker 3D printer, with 3D models designed using a 3D modeling application.

The next step included the design of a live website that would serve as a host and a display page for the data collected by our robot. This was done using some basic popular hypertext, styling, and scripting languages such as HTML, CSS, and JAVASCRIPT. A database is also included for the storing of the data received from the local robot.

Finally, to bring the system altogether, 2 APIs were needed, one on the server-side(the local robot), and one on the client-side(the online website). This was done using the popular PHP language for interactions with the database and data communication from and to the website (alongside some javascript with the AJAX framework) and the python language for data communication from and to the local robot.

2.1 Algorithm

As mentioned earlier in this article, the goal of this project was to create a robot, capable of detecting cases of non-respect of social distance, reporting these cases with an image backup, and finally alerting people of the non-respect of the social distance as illustrated in the figure below.

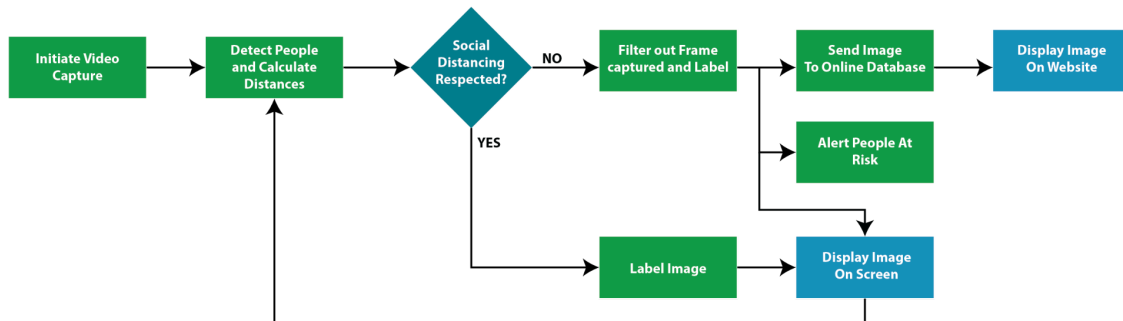


Figure 1: Working model of the IoT social distancing and monitoring robot

More concretely, once the robot's ON/OFF switch button is turned on, the raspberry pi and the various PCB interactives are turned on automatically, waiting for a remote start signal from the robot's user. This start signal

is sent over an SSH connection set between the raspberry pi and the user's computer. Once the start signal is sent, the robot's head starts rotating at a 180° angle. Meanwhile, the robot's camera starts capturing the video stream. Using its processors, the robot detects all the people present in the video stream and then calculates the distance between every 2 people. If this distance happens to be less than the required social distance of 1.5m, the robot captures the people in question and sends the image to an online database from which it is displayed on a website as a report. As proof of the social distance nonobservance, the robot annotates the image with red squares around the people involved, and prints out the number of people at risk.

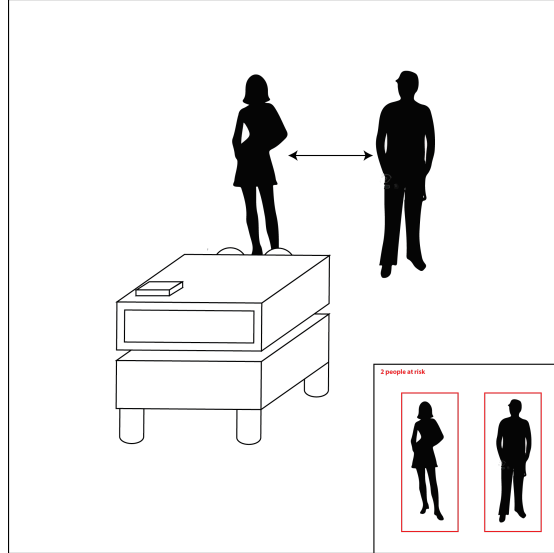


Figure 2. Social distance calculation and labeling

2.2 Apparatus

The implementation of this project required some powerful hardware capable of handling object recognition and other tasks at the same time. Hence our choice of hardware was as follows.

Raspberry pi: A raspberry pi is used to handle both the processing(object detection) heavy activity of our detector module and the power-consuming activity of the motor. It is convenient as it is a credit card-sized computer that can handle a lot of heavy processing computer activities (Raspberry Pi, n.d.). It is also used to handle the API communications between the robot and the website. The modules used are the raspberry pi camera module, the GPIO pins, and the wifi module (Raspberrypi, n.d.).

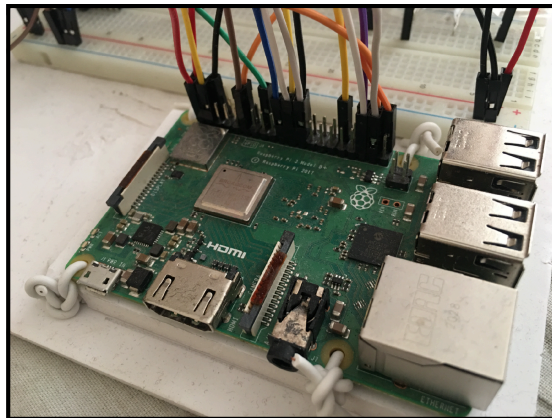


Figure 3. The raspberry pi model 4B

For this project, we opted for a recent version of the raspberry pi with up-to-date functionalities. So we decided to use the raspberry pi 4 model B for its robustness and its better processing capabilities.

The raspberry pi can be controlled in different ways. For this project, we opted to connect to the raspberry pi and control it remotely through an SSH connection (Libguide, 2021).

The integrated camera module is used on the robot for video stream capture and shooting. This camera is set up as shown in (Raspberrypi, n.d.) and controlled using a dedicated code block which can be found in the source code of this project.

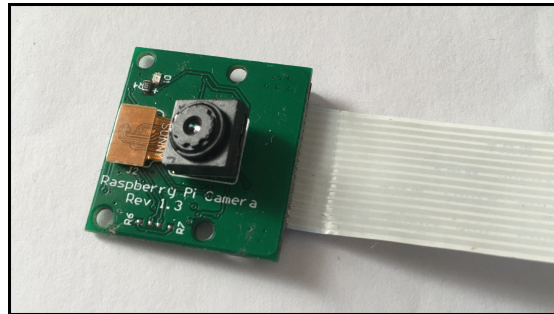


Figure 4. The raspberry pi model 4B's camera module

The raspberry pi GPIO pins are used to handle the exchange of data to and from and to the PCB containing the interactives.

Ultrasonic sensor: An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converting the reflected sound into an electrical signal.

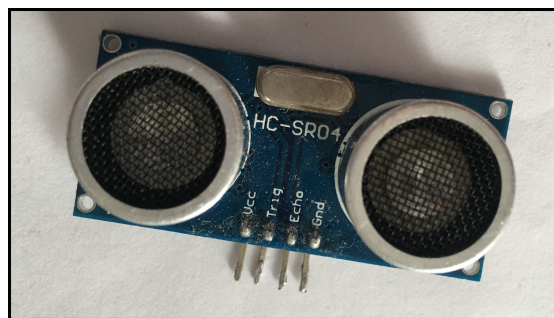


Figure 5. HC-SR04 ultrasonic sensor

The ultrasonic sensor used on this robot is the HC-SR04. It is an ultrasonic ranging module that provides a 2 cm to 400 cm non-contact distance measurement function. The ranging accuracy can reach up to 3mm and the effectual angle is $< 15^\circ$. It can be powered by a 5V power supply which is the case in this project.

Motor and driver circuit: As the motor for this project we used the 28 BYJ-48 Step Motor with ULN2003A Step Motor driver circuit, which we incorporated into the main PCB and whose schematic you can find in section 2.3.

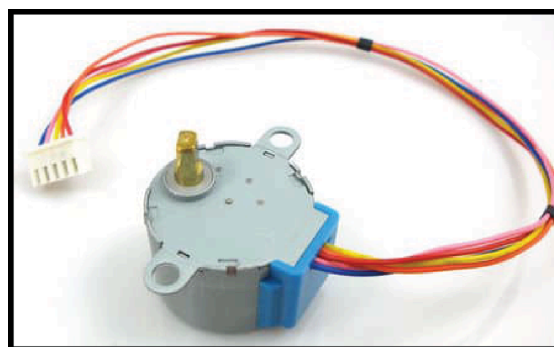


Figure 6. 28 BYJ-48 motors

The stepper motor working principle is Electro-Magnetism. It includes a rotor which is made with a permanent magnet whereas a stator is with electromagnets. Once the supply is provided to the winding of the stator then the magnetic field will be developed within the stator.

The 28BYJ-48 is a 5-wire unipolar stepper motor that runs on 5 volts.

One of the best things about this motor is that it can be positioned accurately, one ‘step’ at a time. This is one of the main reasons why we chose to use this motor rather than motors like the DC motors or Servo motors on this project. The other advantage is that it is relatively precise in its movement and it is quite reliable since the motor does not use contact brushes.

According to the datasheet, when the 28BYJ-48 motor runs in full-step mode, each step corresponds to a rotation of 11.25° . That means there are 32 steps per revolution ($360^\circ/11.25^\circ = 32$).

In addition, the motor has a 1/64 reduction gear set. (Actually, it's 1/63.68395 but for most purposes, 1/64 is a good enough approximation).

What this means is that there are actually 32×63.68395 steps per revolution = $2037.8864 \sim 2038$ steps!

The motor needs a ULN2003 driver circuit to function which we rebuilt in our final PCB to save space.

Buzzer: The buzzer is a sounding device that can convert audio signals into sound signals. It is usually powered by DC voltage. It is widely used in alarms, computers, printers, and other electronic products as sound devices.

The buzzer on this robot is used to alarm people at risk after the robot detects that these people are not respecting the social distance.

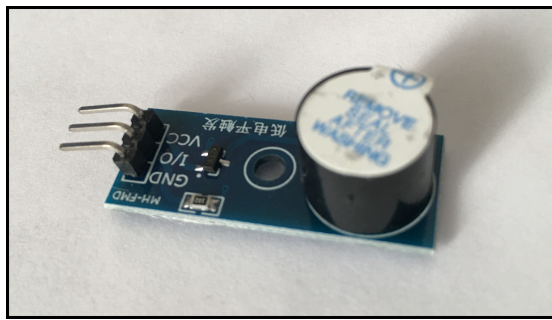


Figure 7. Buzzer

2.3 PCB design

The robot contains a PCB which is included for convenience. The PCB brings together most of the interactives we mentioned above. As you will realize below, part of the logic of the robot is directly included in/handled by the hardware to reduce the number of tasks that the processors have to handle. The schematics are made up of different modules.

The first module is the power module. This module is made up of a simple two-state switch to allow current to power the main circuit in the ON mode and to disconnect the main circuit in the OFF mode. This also contains 2 LEDs, a green one which is lit when the main circuit is powered (the ON mode), and a red one which is lit when the robot is connected to a power source but the whole circuit is not powered (the off mode).

The second module is the motor module. This module contains a ULN2003 integrated circuit, which is an IC driver for the 28 BYJ-48 motors. It also contains 4 LEDs which serve as control LEDs to show that the motor is actually powered as expected.

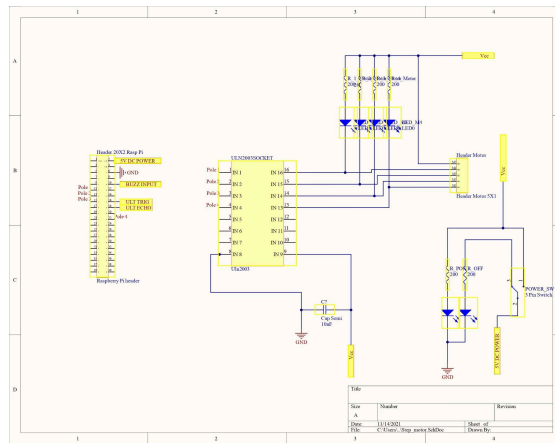


Figure 8. Power and Motor modules schematics

The 3rd module is the buzzer module that contains two LEDs, a buzzer, a PNP transistor, and a CD4070BE XOR logic gate. The LEDs are set to blink one at a time thanks to the XOR gate block. When the buzzer is on (Meaning social distancing is not respected) The red LED blinks, and when the buzzer is off (Meaning social distancing is not respected), the green LED blinks.

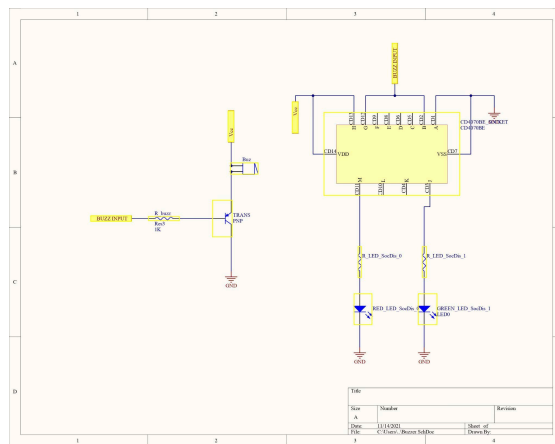


Figure 9. Power and Motor modules schematics

The 4th module is the ultrasonic sensor module, which simply consists of an ultrasonic sensor and voltage divider, made up of 2 resistors. This voltage divider makes it possible to convert the 5V signal that the ultrasonic sensor produces to a 3V signal to suit the 3V raspberry pi GPIO pinout.

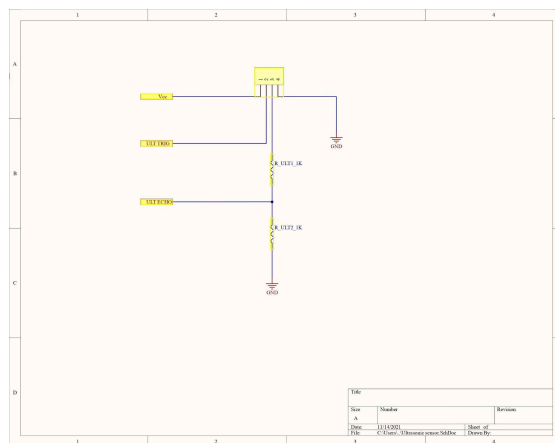


Figure 10. Ultrasonic sensor module schematic

All together these schematics were merged and converted to a single PCB layout using Altium designer.

After this 3D design, the PCB top and bottom views were printed separately on greaseproof paper. These views were imposed on the top and bottom of a double-sided copper plate using an iron machine.

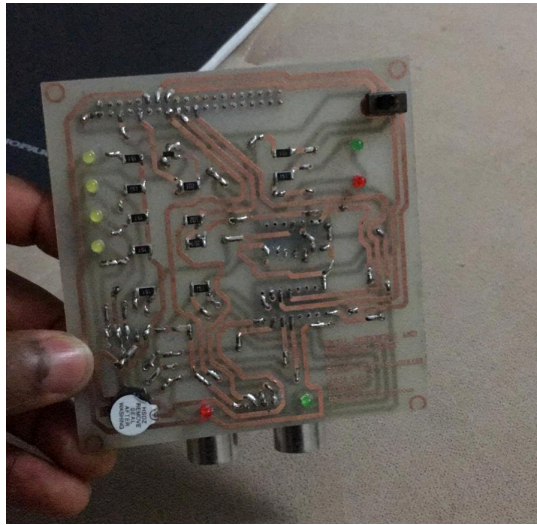


Figure 11. Final PCB with interactives installed

The resultant PCB was immersed in a mixture of Hydrogen peroxide and Acetic acid in order to remove the remains of the copper not covered by the print. Then all the holes were created, all joints filled and all components added using solder wire and a solder machine.

2.4 Image recognition and image processing

This is an extremely important part of this project. One of the main problems we faced while deciding on which technologies would be used for implementing this project was the absence of an affordable solution to calculate the distance between people scattered in public places. Hence the main solution we could find to handle this task was to use object recognition and image processing, that is, machine learning.

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 an image or characteristics/features associated with that image (Anbarjafi, 2020).

Object recognition, in the context of machine vision, is the ability of software to identify objects, places, people, writing, and actions in images. Computers can use machine vision technologies in combination with a camera and artificial intelligence software to achieve image recognition (TechTarget, 2021).

To perform these tasks of image processing and object detection, we needed a powerful lightweight object detection algorithm. Our best alternative was to use the yolov3 tiny detection algorithm.

YOLOv3 (You Only Look Once, Version 3) is a real-time object detection algorithm that identifies specific objects in videos, live feeds, or images. YOLO uses features learned by a deep convolutional neural network to detect an object. Versions 1-3 of YOLO were created by Joseph Redmon and Ali Farhadi (Redmon, n.d.).

Using this algorithm on the raspberry pi is quite not an easy task. It required the proper setup of the raspberry pi. The following pieces of software are installed on the raspberry pi in order to ensure its accuracy and speed in object detecting, without overstraining the processors.

Tensorflow: TensorFlow is a Python-friendly open-source library for numerical computation that makes machine learning faster and easier. We installed TensorFlow on the raspberry pi 4B using the steps found in an article (Q-engineering, 2021).

Opencv: OpenCV was by far the most difficult to install into the raspberry pi as it just doesn't work when installed with the PIP command as in the windows operating system. Hence the successful steps we found are mentioned in the article (Rosebrock, 2018). This software package is installed from the source in order to guarantee that it works. We also made sure to add a symbolic link to the OpenCV library in our python libraries directory. This is extremely important as the pack will just not be recognized(or found) otherwise by our python interpreter even though it is installed.

Darknet NNpack: Though we finally didn't use the darknet NNpack framework, we found it useful when it comes to speeding up the detection of objects using YOLO. Darknet is a framework written in C that permits us to fasten object detections using the YOLO algorithm.

Darknet is primarily not made for systems like the Rasbean OS used on the raspberry pi. So we found ourselves obliged to use a version that was created to be used on constrained hardware such as the raspberry pi. So just like the Opencv pack, we needed to install the software from the source as mentioned in (Ertugrul, 2020). We found out that this article wasn't complete for us, as we needed to install some OpenCV dependencies for the MAKE function to proceed. We finally found the solution in an article (Code Pioneer Network, 2017). You'll realize that the website is in Chinese but we just translated and read it in order to find our solution. The research was hard but was needed. We finally succeeded in resolving the OpenCV-not-found error using the following line of code. "*Sudo apt-get install libopencv-dev*".

2.5 Software

The software for this project is mainly separated into 2 parts, an online part, made up of a database and a website, and a local part made up of the robot and its various apparatus. These two sections communicate together with the help of APIs.

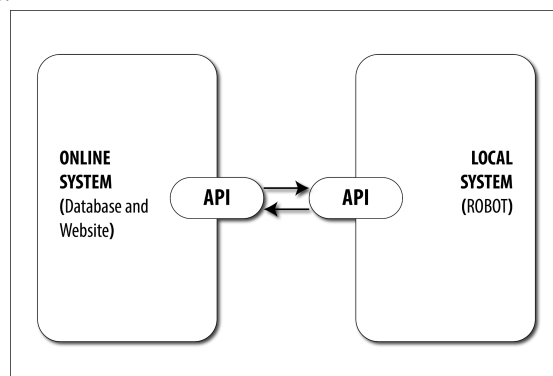


Figure 12. IoT robot software's working model

All the software for the local system is installed on the raspberry pi and written in the Python language. As mentioned in section 2.3, some of the robot's logic functions are directly handled by the hardware to reduce the number of tasks that the processor has to handle. Hence, the software for this project (that is, the part hosted on the raspberry pi) is mostly machine learning intensive.

Nonetheless, there are specialized functions included for handling the output signals sent to the PCB containing the interactives, the video stream capture, and the exchange of data between the local robot and the online database.

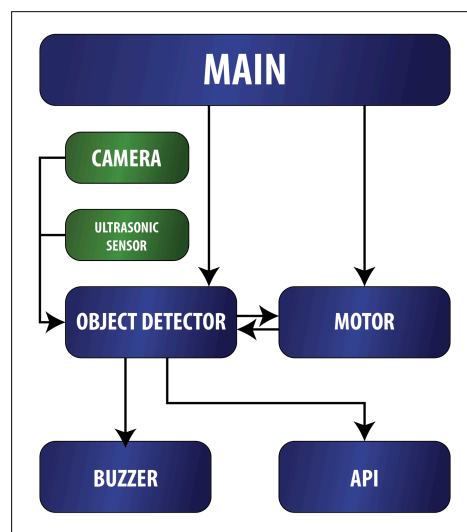


Figure 13. IoT robot local software's working model

Main module: The main module is the runner of the whole robot. It launches the robot and stops it when needed. It uses python threading to handle the motor's motion and object detection simultaneously.

```
1  from Interactives import motor
2  from Interactives import detector
3  import threading
4
5  t1 = threading.Thread(target=detector.detect)
6  t2 = threading.Thread(target=motor.rotate)
7
8  t1.start()
9  t2.start()
10
11
12
```

Figure 14. Main module python code

Object detector module: The detector is the most important module as it takes images of the current scene, detects all objects, filters out humans, calculates the distances between all these people, and decides which people do not respect the social distance. It also gives a signal to all other modules, such as the motor for it to stop, and the buzzer for it to buzz, and sends data to an online and offline database through the API module.

Motor module: The motor module controls the movements of the robot for 180 degrees. It waits for a stop and starts signal from the detector in order to stop and restart when needed.

Buzzer: The buzzer is set to buzz and stop 10 times each time a case of social distance not respected is detected by the robot.

```
1  import RPi.GPIO as GPIO
2  import time
3  from Interactives import motor
4
5
6  def buzz(sec):
7      buzzer_pin = 15
8
9      GPIO.setmode(GPIO.BCM)
10     GPIO.setup(buzzer_pin, GPIO.IN)
11     GPIO.setup(buzzer_pin, GPIO.OUT)
12
13     print('I am buzzing due to soc_dis not respected')
14
15     for i in range(0,10):
16         GPIO.output(buzzer_pin, GPIO.HIGH)
17         time.sleep(sec/2)
18         GPIO.output(buzzer_pin, GPIO.LOW)
19         time.sleep(sec/2)
20
21     GPIO.output(buzzer_pin, GPIO.HIGH)
22
23     motor.start()
```

Figure 15. Buzzer module python code

Api(Local): The API is the module made to send data to the website's database and to the pi's folders. It handles all the communications of the robot with external devices such as the online website's database (Edutechional, 2018).

```
1  import requests
2  import os
3  #import pandas as pd
4
5  def send(img_name, num_peop):
6
7      img_folder = "https://muhtasi-jordihako.com/Images/"
8      img_link = os.path.join(img_folder, img_name)
9
10     data_name = ["img_name", "num_peop", "img_link"]
11     data = [img_name, num_peop, img_link]
12
13     payload = dict(zip(data_name, data))
14
15     resp = requests.post('http://muhtasi.jordihako.com/api.php', params=payload)
16     print(resp.text)
17
```

Figure 16. Local API python code

The software for the online part of this project is relatively simple.

An **API(Online)** written in PHP language, as shown in (Traversy Media, 2018), receives the data packages(payloads) sent by the robot containing the number of people at risk in the image, the name of the image, and the link pointing to where the image is stored in the website file system.

```

1  <?php
2
3  include_once 'init.php';
4
5  $img_name = $_GET['img_name'];
6  $num_peop = $_GET['num_peop'];
7  $img_link = $_GET['img_link'];
8
9  $sql = "INSERT INTO Images_captured (Filename, No_people, File_link)
10 VALUES ('$img_name', '$num_peop', '$img_link')";
11
12 if ($conn->query($sql) == TRUE) {
13     echo "New record created successfully";
14 } else {
15     echo "Error: " . $sql . "<br>" . $conn->error;
16 }

```

Figure 17. Online API PHP code

This data is stored in a SQL database, from which the website's table is fed using some PHP scripts directly integrated into the website's code.

```

51 <?php
52
53 include_once 'init.php';
54 $sql = "SELECT * FROM Images_captured";
55 $result = $conn->query($sql);
56
57 if ($result->num_rows > 0) {
58     while($row = $result->fetch_assoc()) {
59         echo "<tr><td>" . $row["id"] . "</td><td>" . $row["Filename"] . "</td><td>" .
60             $row["No_people"] . "</td><td><img src='" . $row["File_link"] . "' width='90%'></td></tr>";
61     }
62 } else {
63     echo "No Data to show";
64 }
65
66 ?>

```

Figure 18. Website inline PHP code

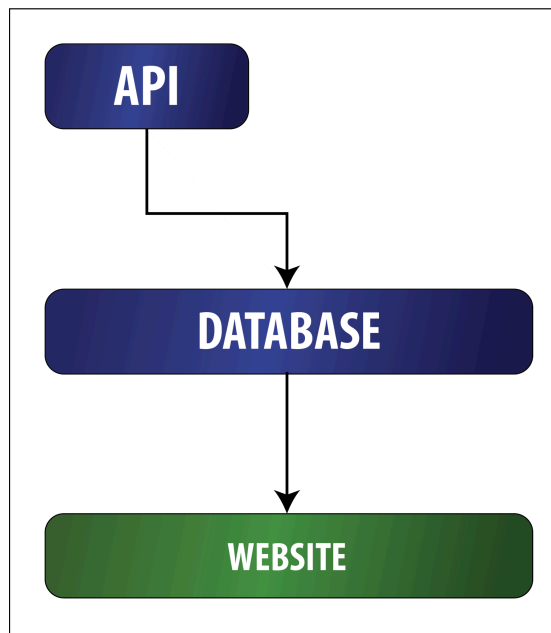


Figure 19. IoT robot online software's working model

2.6 Mechanics and case design

One of the main abilities we wanted to give the robot was the capability of watching what is happening 180deg around it. Hence, we created a convenient design to handle this motion.

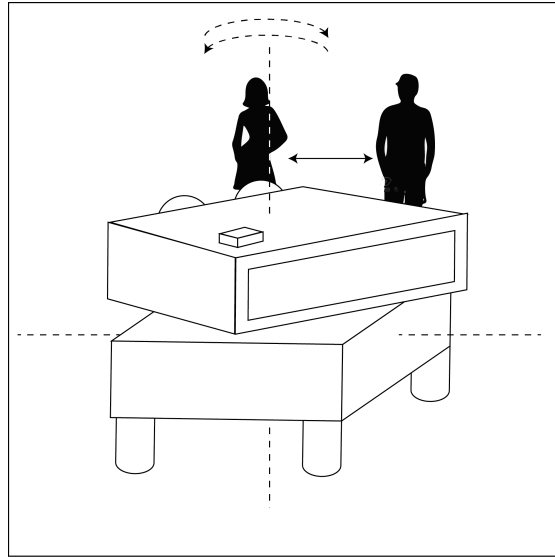


Figure 20. Robot's case angle of rotation

The robot is made of two rectangular boxes joined together by a pivot(that of the motor). The lower case is mainly set to provide stability and balance to the robot. Hence it is made of a thin box with 4 legs, one at each extremity of the box to help handle the load of the upper box. The upper case is set to contain all the circuitry of the robot, that is the PCB, the raspberry pi, and the motor. This makes it a little heavy and increases the load on the pivot, so we opted to use some metallic screws to help everything stay in place during rotation.

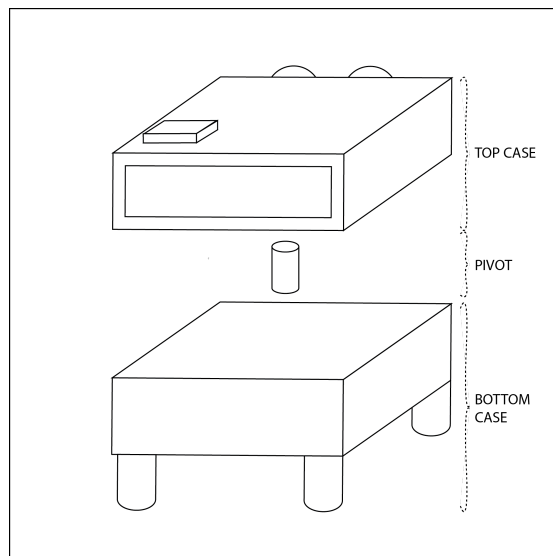


Figure 21. Robot's top case, bottom case, and pivot

The whole case was designed on a 3D design software and printed using a Ultimax 3D printer and black and grey filament as shown in the figure below;

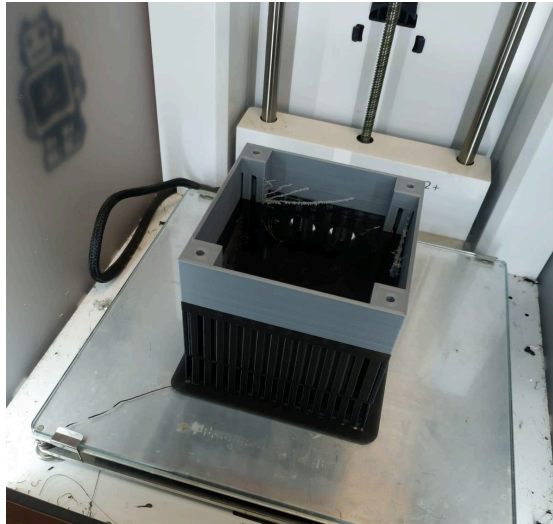


Figure 22. Case's 3D printing

The 3D printed case was cleaned up of support blocks, the PCB, raspberry pi, and motor were mounted and the robot was ready for the test as shown below.



Figure 23. Robot's final look

2.7 Website design

The website is designed using HTML, CSS Javascript, and PHP. This website retrieves the data stored in the database by the robot's APIs, identifies the location of each image using the link specified in the database, and displaces the information on a table.

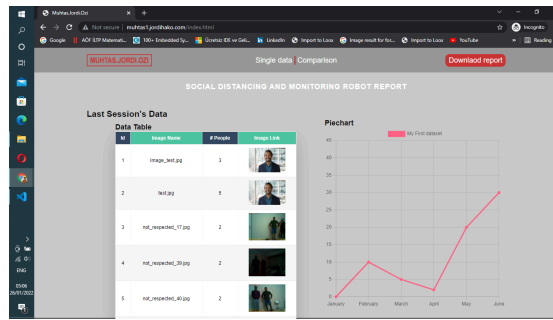


Figure 24. Website

The website is updated in real-time by an inline PHP code block. This permits it to display the image of a new case of non-respect of social distancing immediately after it is detected. The website's code can be found in this article's attachments.

2.8 Website functioning and application

The website is powered by PHP and HTML and CSS. It gets most of its data from a database hosted on the same WebHost. The database contains 4 main columns; the ID column, a column to store the number of people in the image, a column to store the name of the image that serves as proof of people not respecting social distancing, and a column to store a link to where the image is stored on the website's storage.

A PHP loop runs on the database continuously and displays all the data present in the specified storage by retrieving the image corresponding to the link in the database.

A table written in HTML and CSS is used to display the content of the database as a report of the people not respecting social distancing in the last work session of the robot. And a chart is used to help analyze the changes in trends within the data over time.

A download button is included to permit any user to download the report generate on the website for various uses.

3. Conclusion

The implementation of this project using the raspberry pi wasn't easy due to CPU constraints, heating, and a lot of other issues independent of our means. This could definitely be improved using some techniques such as using more than 1 raspberry pi, adding a cooling fan to the system, and so on. We will leave this for the future due to time constraints.

The robot has also been designed in such a way that it can be repurposed. That is, the same robot can be used to accomplish other tasks just by changing the code in the raspberry pi.

For example, one purpose we are planning to use the robot for in the future is image collection for machine learning purposes.

The mechanics of the robot also need to be improved, especially at the level of the pivot. As we mentioned above, the pivot needs to manage the heavyweight of the PCB and its supporting elements, and the torque produced by the robot's rotation in the current model.

To round up, even though we met some constraints with the hardware, we were able to achieve outstanding results with what we had in hand. Our robot works as expected with little time and position latencies, which can easily and effectively be handled by adding another raspberry pi to the system to benefit from its processor.

References

- Anbarjafi, G. (2020). *1. Introduction to image processing | Digital Image Processing*. Sisu@UT. Retrieved May 12, 2022, from <https://sisu.ut.ee/imageprocessing/book/1>
- Code Pioneer Network. (2017). *include/darknet.h:25:43: fatal error: opencv2/highgui/highgui_c.h: No such file or directory*. Code Pioneer Network. Retrieved March 24, 2022, from <https://www.codeleading.com/article/11573190878/>
- EduTechional. (2018, February 17). *How to Communicate with APIs in Python with the Requests Package*. YouTube. Retrieved March 24, 2022, from <https://youtu.be/QovKok-2u9k>
- Ertugrul, E. (2020, September 5). *Building and running darknet with nnpack on raspberry pi*. egemen ertugrul. Retrieved March 24, 2022, from <https://egemenertugrul.github.io/blog/Darknet-NNPACK-on-Raspberry-Pi/>
- İLKBHAR, F., & SUNGU, E. (2020, December). Analysis of Artificial Intelligence Technologies Used In The Covid-19 Outbreak Process. *International Journal of Applied Mathematics Electronics and Computer*, 8(4), 154-162. <https://dergipark.org.tr/tr/pub/ijamec/issue/57538/800910>

- Kaufman, D. (n.d.). *PCB Design Steps & Complete Guide*. Cirexx. Retrieved March 24, 2022, from <https://www.cirexx.com/pcb-design-steps/>
- KAYA, E. (2021, 10 30). A Novel Approach Based to Neural Network and Flower Pollination Algorithm to Predict Number of COVID-19 Cases. *Balkan Journal Of Electrical And Computer Engineering*, 9(4), 327-336. <https://dergipark.org.tr/tr/pub/bajece/issue/65264/932391>
- Libguide. (2021, June 22). *SSH - Remote Login - Raspberry Pi*. LibGuides. Retrieved March 24, 2022, from <https://libguides.nyit.edu/c.php?g=469894&p=3365470>
- Nazario, B. (2021, December 22). *Coronavirus & COVID-19 Overview: Symptoms, Risks, Prevention, Treatment & More*. WebMD. Retrieved March 24, 2022, from <https://www.webmd.com/lung/coronavirus>
- Nehir ÖZNUR MUZ, F., KILINÇ, A., & Fatih ÖNSÜZ, M. (2020, 09 28). COVID-19 PANDEMİSİNDE YAPAY ZEKANIN KULLANIMI. *ESTÜDAM Halk Sağlığı Dergisi*, 5, 1. <https://dergipark.org.tr/tr/pub/estudamhsd/issue/56854/798676>
- PRIME, E. (2020, May 30). *How To Install & Set Up Raspberry Pi OS - Pi4 Pi3 Pi2*. YouTube. Retrieved March 24, 2022, from <https://youtu.be/y45hsd2AOpw>
- Q-engineering. (2021, December 12). *Install TensorFlow 2.2.0 on Raspberry Pi 4*. Q-engineering. Retrieved March 24, 2022, from <https://qengineering.eu/install-tensorflow-2.2.0-on-raspberry-pi-4.html>
- Raspberrypi. (n.d.). *Getting started with the Camera Module - Introduction | Raspberry Pi Projects*. Projects | Raspberry Pi Projects. Retrieved March 24, 2022, from <https://projects.raspberrypi.org/en/projects/getting-started-with-picamera>
- Raspberry Pi. (n.d.). *What is a Raspberry Pi?* Raspberry Pi. Retrieved March 24, 2022, from <https://www.raspberrypi.org/help/what-is-a-raspberry-pi/>
- Razmadze, D. (2020, October 15). *Tutorial: Control Raspberry Pi GPIO via HTTP Web Server*. YouTube. Retrieved March 24, 2022, from https://youtu.be/owgRkU_-4lw
- Redmon, J., Redmon, & Ali. (2018). YOLOv3: An Incremental Improvement. *arXiv*. Retrieved May 12, 2022, from <https://pjreddie.com/darknet/yolo/>
- Rosebrock, A. (2018, September 26). *Install OpenCV 4 on your Raspberry Pi*. PyImageSearch. Retrieved March 24, 2022, from <https://www.pyimagesearch.com/2018/09/26/install-opencv-4-on-your-raspberry-pi/>
- SİRİPONGDEE, K., PIMDEE, P., & TUNTIWONGWANICH, S. (2020, 06 15). A blended learning model with IoT-based technology: effectively used when the COVID-19 pandemic? *Journal for The Education Of Gifted Young Scientist*, 8(2), 1. <https://dergipark.org.tr/tr/pub/jegys/issue/53184/698869>
- TechTarget. (2021, 10). *What Is Image Recognition? Definition from SearchEnterpriseAI*. TechTarget. Retrieved May 12, 2022, from <https://www.techtarget.com/searchenterpriseai/definition/image-recognition>
- Traversy Media. (2018, May 27). *PHP REST API From Scratch [1] - Database & Read*. YouTube. Retrieved March 24, 2022, from <https://youtu.be/OEWXbpUMODk>