How Raspberry Pi Microcomputers and Computer Vision Artificial Intelligence Can Be Utilized to Provide Aid to the Visually Impaired Community

Siddhant Bhardwaj, Cheshire High School, 11th Grade

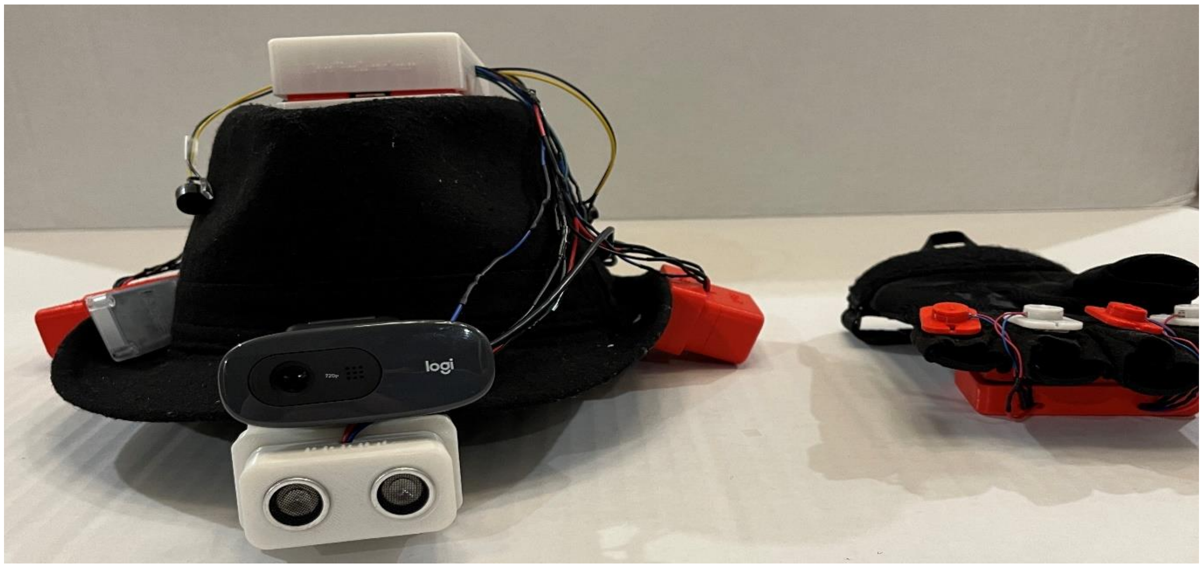


Table of Contents

Abstract

According to the World Health Organization, an estimated 2.2 million people globally are suffering from a visual impairment or total blindness. Although millions have an inability to see, there is a lack of a practical product on the market which is both cost effective and efficient in assisting the visually impaired population. The Object Detection Universal System Plus (O.D.U.S.+) was designed in order to be that solution, allowing and providing the wearer with freedom and safety in movement, as well as extending their capabilities with Computer Vision Artificial Intelligence. The device was programmed with the programming language Python, and utilized two Raspberry Pi microcomputers, 4 ultrasonic sensors, 4 vibration motors, and a camera. O.D.U.S.+ works by collecting distance data using ultrasonic sensors and parsing it to provide alerts if an object is within 3-feet of the wearer, through 3 methods. These alerts are vibrations created by vibration motors on the glove, with data being received over BLE, or Bluetooth Low Energy, a series of beeps from a buzzer, and a computer-generated voice produced through a text-to-speech program. Furthermore, by using the TensorFlow LITE database, coupled with the programming library, OpenCV, the device can accurately describe objects in front of the wearer, providing an added sense of awareness. Performance tests were done on both the ultrasonic sensors and OpenCV to determine their accuracy. More research and development is needed to expand the functionality of the device, including multiple modes for different applications of the Computer Vision AI for different situations.

Introduction

According to a study conducted by the World Health Organization, an estimated 2.2 million people globally are suffering from a visual impairment or total blindness. Although hundreds of millions have the inability to see, there is a lack of a practical product on the market which is both cost effective and efficient in assisting the visually impaired population. Devices such as Ray Electronic Mobility Aid for the Blind act as a cane replacement, but it does not allow full mobility for the user, as their hand is still occupied with the device since the user still has to use the device by moving it from side to side.

The goal of the Object Detection Universal System Plus (O.D.U.S.+) was to design and develop a wearable device to aid the visually impaired community in safety of movement, reducing or eliminating the need for a cane and additional assistance using sensor-based technology, easy to understand and user-friendly alerts, and with integrated Computer Vision Artificial Intelligence, for added safety and usability. The objective was to make a wearable device that can detect what an object is, how far away it is, and asses how the user should avoid it, making it easier to know where an object is and how to avoid it.

What was hoped to be achieved for the Object Detection Universal System Plus was to create a cost effective and efficient device, which provides 360-degree protection for the wearer, real-world object detection, allowing the wearer full mobility along with user friendly alerts. O.D.U.S.+ has four ultrasonic sensors, positioned in the four directions of front, back, left, and right. With their overlapping ranges, the sensors were able to provide protection all

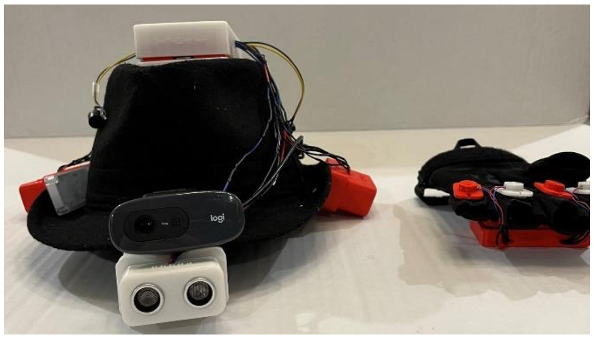


Figure 1: Object Detection Universal System Plus

around the wearer. The device also has a USB camera mounted onto the front, so Computer Vision can be used. There are multiple alerts that are produced for the user to comprehend and circumvent around objects. Two of the alerts can be heard from the hat, a spoken voice detailing what an object is and how far away it is, and a buzz from a buzzer. Another type of alert is a vibration from vibration motors found on the glove. The vibration motors get sent information to vibrate wirelessly over Bluetooth Low Energy (BLE), which allows for full mobility with their user's hand being completely free to use.

Materials and Methods

Goals and Requirements

The goal of the Object Detection Universal System (O.D.U.S.) was to design and develop a wearable device to aid the visually impaired community in safety of movement, reducing or eliminating the need for a cane and additional assistance using sensor-based technology, Computer Vision AI, and easy to understand and user-friendly alerts. The hat, having four ultrasonic sensors, covering all directions for a 360-degree protection system, which gives the user protection while they are moving around. It also has a camera mounted on the front, real-world objects could be detected, adding another measure of safety and independence. The wearer will receive alerts that they can understand easily, a spoken voice, which states what the object is, the direction, and how far away it is. There is another alert made from the buzzer, which changes the frequency of the beeps based on the distance of an object relative to the wearer. The last type of alert is a vibration produced by one of the vibration motors on the glove, where each of the motors represent one of the directions, front, back, left, and right. One of the motors will vibrate based off the distance of the closest object.

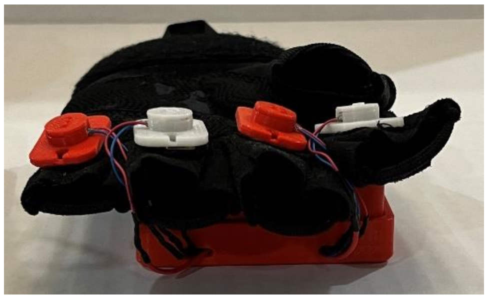


Figure 2: O.D.U.S.+ Glove with Vibration Motors and the Raspberry Pi Zero W

Materials

- HC-SR04 Ultrasonic Sensors
- Raspberry Pi 4 B+
- Raspberry Pi Zero W
- Coral Edge TPU

- USB Camera
- Vibration Motors
- Power Banks
- Jumper Cables
- Momentary Switches (Push Buttons)
- Hat (Fedora)
- Fingerless Glove
- 3d Printer
- Soldering Iron and Solder
- Wire Stripper and Cutter
- Electrical tape and Heat Shrink
- Needle and thread
- Zip ties

Methods

The initial design of the prototype was drawn out on a scrap piece of paper, a rough sketch of the positioning of the sensors and the Raspberry Pi's. Later, the sketches progressed into 3-D models designed in Autodesk Fusion 360. There were 7 iterations of sensor holdings, 3-iterations of vibration motor housings, 2 iterations of the Raspberry Pi case, and 1 iteration of the camera mount. In order to fit all these

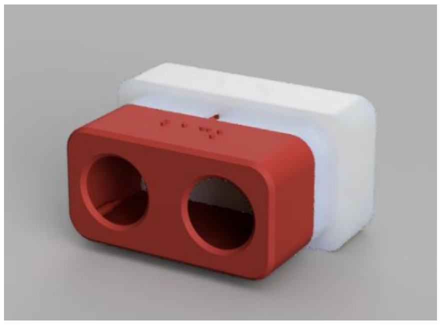


Figure 4: 3-D Render of the Back Ultrasonic Sensor Case

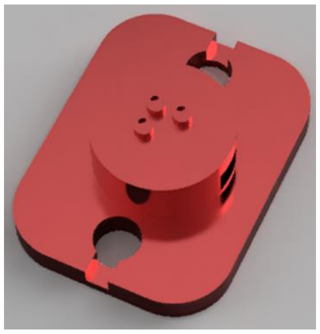


Figure 3: 3-D Render of the Front Vibration Motor Case

pieces effectively on a hat, the best type of hat would be a fedora, because of its wide brim and indentation on the top. Each of the ultrasonic sensors are mounted onto the hat with custom 3-part 3-D printed housings, the case body, the back of the case body, and the hat mount. The hat mount was sewn directly onto the hat. Two 3 x 5 mm neodymium magnets were embedded in both the back of the case body and the hat mount, which allows for the sensors to be removable, making it easier to service the

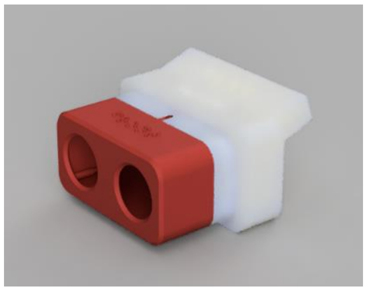


Figure 5: Front Ultrasonic Sensor Mount

unit, as well as being able to wash the hat as needed. But, on a fedora, only the sides and back have a raised, angled brim, so the front sensor required a new mount, which included a 20-degree extension with 4 holes to be sewn on. The case body and the hat mount have matching braille lettering to ensure the correct placement of the sensors after removal. This is the same for the Raspberry Pi case and the Coral Edge TPU Case. Then there were the wiring diagrams, which showed how each device was wired to the Raspberry Pi's. On the hat, the Power and Ground pins of the 5-Volt ultrasonic sensors were connected in a parallel circuit. The Trigger and Echo Pins were connected to unique pins, to be referenced in the code.

Figure 5: Raspberry PI 4 Case

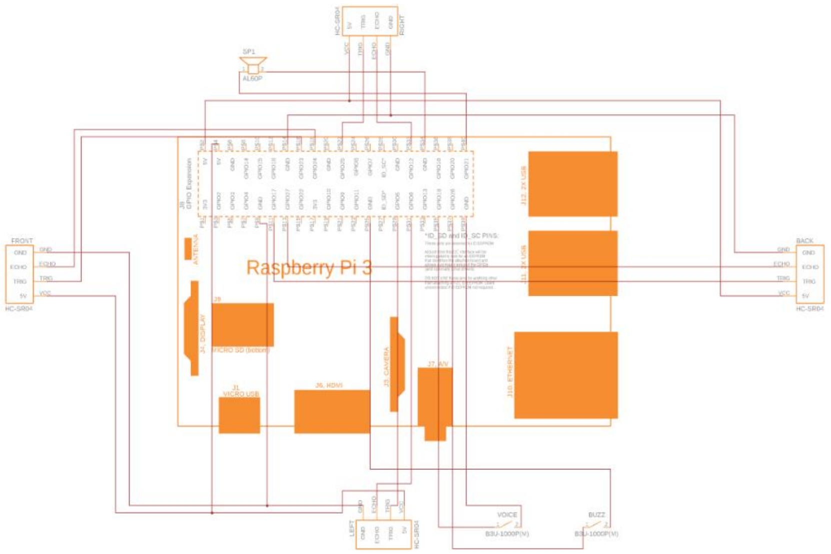


Figure 7: Ultrasonic Sensor Module Schematic

Peripherals were also added, such as two momentary switches (push buttons) and a buzzer, each wired to unique GPIO Pins. On the glove, four vibration motors were soldered and wired to unique GPIO pins on the Raspberry Pi Zero W, one for grounding the motor, and one for interfacing with the motor through the code. Each motor was embedded into a 3-D printed vibration motor case, with braille lettering to correspond with each direction that the Ultrasonic Sensors on the hat are facing. A 3-D Printed case was designed for the Raspberry Pi Zero W and was sewn onto the glove.

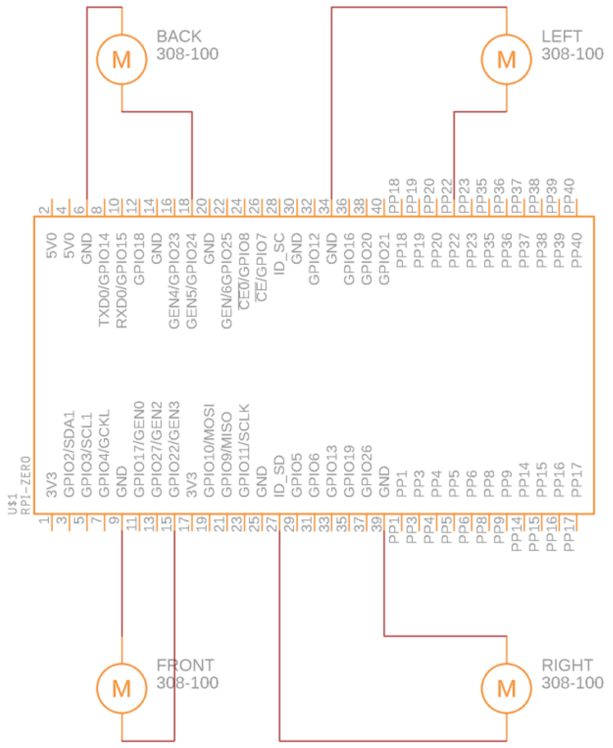
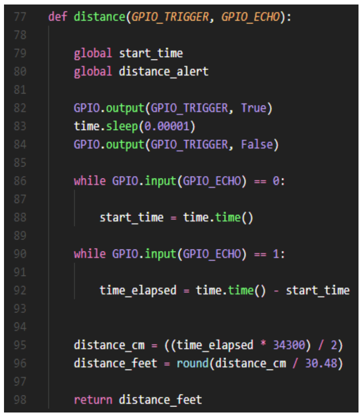


Figure 8: Vibration Motor Module Schematic

For the Raspberry Pi 4 on the hat, A custom Python script was created to continuously collect distance data every second from each sensor, parse it, and provide alerts if an object is within 3-feet of the wearer. Those outputs include a computer-generated voice from a text-to-speech library, a series of beeps generated from the buzzer, and vibrations generated by motors on the glove. Another function was created to transmit the distance data to the Raspberry Pi Zero W on the glove, over Bluetooth Low-Energy (BLE). This wireless connection ensures full mobility and freedom of movement for the wearer. The program on the Hat Transmitter Module also has Computer Vision capability, with the library, OpenCV. It

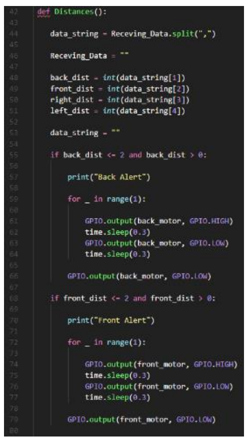


```
def distance(GPIO_TRIGGER, GPIO_ECHO):

    global start_time
    global distance_alert

    GPIO.output(GPIO_TRIGGER, True)
    time.sleep(0.00001)
    GPIO.output(GPIO_TRIGGER, False)

    while GPIO.input(GPIO_ECHO) == 0:

        start_time = time.time()

    while GPIO.input(GPIO_ECHO) == 1:

        time_elapsed = time.time() - start_time


    distance_cm = ((time_elapsed * 34300) / 2)
    distance_feet = round(distance_cm / 30.48)

    return distance_feet
```

Figure 9: Excerpt from the Ultrasonic Sensor program to continuously collect distance data

references a dataset compiled in the TensorFlow Lite database, which has the capability to detect and describe objects. The function takes the description of the object, and converts it into a String, or a data type that the Espeak Text to Speech Library text-to-speech function is able to utilize, allowing the computer-generated voice to describe the object that is in front of the wearer, as well as how far away it is from the wearer. In the code on the hat, other functions were in place for Bluetooth Connectivity using the Bluedot Comm(communication) API and changing the modes.

On the glove, the custom Python on the Raspberry Pi Zero W had two functions. One was to receive the distance data from the Hat Transmitter Module, and the other was to interpret each distance and turn on the corresponding vibration motor on the glove, based on the smallest value.



```
def Distances():

    data_string = Receving_Data.split(",")

    Receving_Data = ""

    back_dist = int(data_string[1])
    front_dist = int(data_string[2])
    right_dist = int(data_string[3])
    left_dist = int(data_string[4])

    data_string = ""

    if back_dist <= 2 and back_dist > 0:

        print("Back Alert")

        for _ in range(1):

            GPIO.output(back_motor, GPIO.HIGH)
            time.sleep(0.3)
            GPIO.output(back_motor, GPIO.LOW)
            time.sleep(0.3)

        GPIO.output(back_motor, GPIO.LOW)

    if front_dist <= 2 and front_dist > 0:

        print("Front Alert")

        for _ in range(1):

            GPIO.output(front_motor, GPIO.HIGH)
            time.sleep(0.3)
            GPIO.output(front_motor, GPIO.LOW)
            time.sleep(0.3)

        GPIO.output(front_motor, GPIO.LOW)
```

Figure 10: Excerpt from Glove Receiver function to continuously receive distance data

Some edits that were made were the continuous loop of information that was being sent from Raspberry Pi to Raspberry Pi. The data had to be a string, not an integer, as specified from the Bluedot Documentation. There was an issue with Espeak, and it was an internal issue with Raspbian Buster, the operating system of the Raspberry Pi, because the Text to Speech Library was not properly binded to the ALSA Utils library, or the audio synthesizer for the Raspberry Pi. The library needed to be uninstalled, then reinstalled and then the text file of ALSA Utils had to be edited to accommodate for the Espeak Library.

Results

To ensure that every component of the O.D.U.S.+ was working properly, extensive testing was completed. The Ultrasonic Sensors are the most important part of the design, and it’s important to test that each of them are working, as well as the accuracy of each sensor. This was done by collecting distance data in one program and creating a distance over time graph based on the data using the Matplotlib Python Library. This was done because it detailed the accuracy of how close an object was to the sensor. After each of the sensors were tested, it was determined that they all worked, and they were accurate.

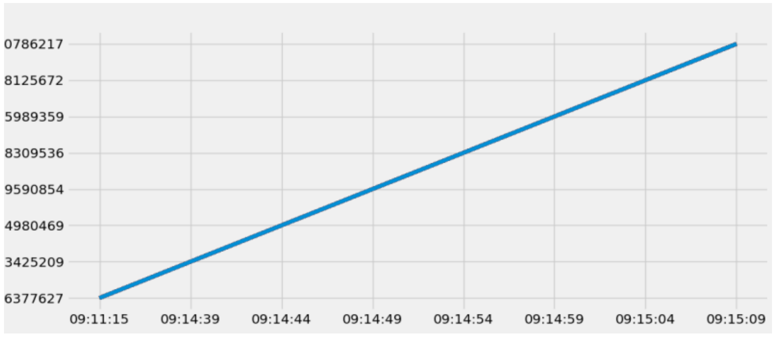
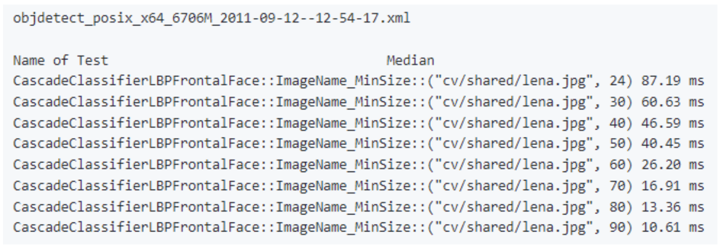


Figure 11: Distance vs. Time Graph for the Front Ultrasonic sensor

Programming the Computer Vision AI was the most complicated part of this project because there were multiple parts to consider. The first test was making sure that the OpenCV commands from the library were executed fast enough to be effective, to help the visually impaired to identify objects quickly. This was done by completing a median-time performance test for the library, generating a chart of times, in milliseconds, that the commands were executed. After this experiment, it was determined that OpenCV was running fast enough, with the longest command taking 87.19 milliseconds. Finally, the last thing to test was the combination of OpenCV and the TensorFlow Lite Database, to test the framerate that the Raspberry Pi 4 would be able to detect objects. This was done by turning on the framerate overlay in the object detection preview window. After this experiment, it was



```
objdetect_posix_x64_6706M_2011-09-12--12-54-17.xml

Name of Test                                          Median
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 24) 87.19 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 30) 60.63 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 40) 46.59 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 50) 40.45 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 60) 26.20 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 70) 16.91 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 80) 13.36 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 90) 10.61 ms
```

Figure 12: The OpenCV Median-Time Performance Test

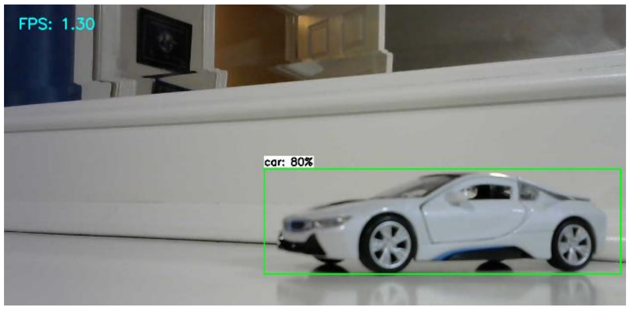


Figure 13: Screenshot of the first test with OpenCV and TensorFlow Lite

determined that the framerate was much too low for Computer Vision to be effectively used, averaging at about 2 frames per second. In order to solve this problem, a Coral Edge TPU was utilized, as an Edge TPU is a device meant specifically to run artificially intelligent models. This drastically increased the frame rate by 4 times the original, at about 8 frames per second.

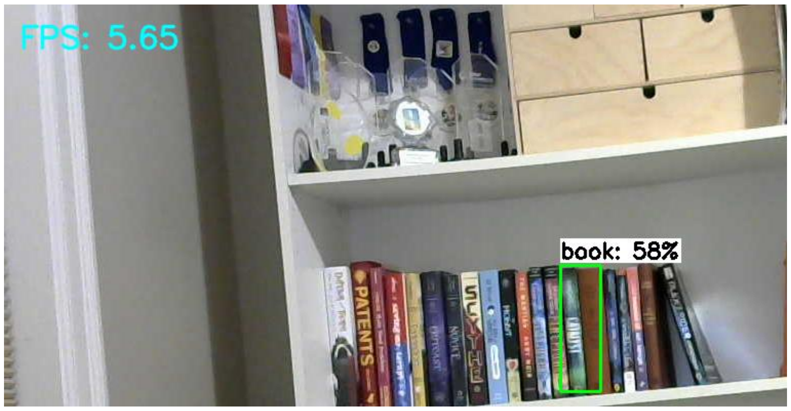


Figure 14: Screenshot of the second test with OpenCV and TensorFlow Lite

Discussion

The Object Detection Universal System Plus is a device with the hopes of being a viable, comfortable, and functioning aid for the visually impaired. The four ultrasonic distance sensors, placed in the front, back, left, and right, provide a 360-degree protection radius for the wearer. The camera mounted on the front allowed for the Computer Vision AI to actively detect real-world objects, creating an added measure of safety and autonomy for the wearer. The wearer can be alerted about an object that is less than 3 feet away from them by three alerts, a vibration, a series of beeps, and a computer-generated voice. These alerts are user friendly, with the computer-generated voice stating what the object is, what direction the object is in and how far away it is. With the buzzer mode, the frequency of beeps change based upon how close the object is, and the vibration mode vibrates one of the four motors, representing front, back, left, and right for the closest object that is near the user. The vibration is continuous, and the data is sent from the Raspberry Pi 4 connected to the sensors to the Raspberry Pi Zero W, connected to the vibration motors. The data is being sent over the Bluetooth Low Energy link, and is acted upon by a custom Python program, on both the Raspberry Pi 4 and the Raspberry Pi Zero W. On the Raspberry Pi 4, A custom program was written in the Python programming language that collects the data, and parses it to provide outputs, or alerts, based off of the closest distance. A continuous loop function was created to collect data every five seconds. Moreover, in the Script, a function was made so that the user can change modes to buzzer and audio mode. A separate thread needed to be made, so that the audio and buzzer mode would be able to run in the background, while the sensors are still collecting data. The Python Library used for text to speech, called Espeak, did not initially work on the Raspberry Pi 4. The Espeak library did not bind properly to the ALSA Utils audio synthesizer of the Raspberry Pi. Changes were needed to be made to the ALSA Utils text file, to bind to the Espeak Library and create a synthesized voice. Another program needed to be created on the Raspberry Pi Zero W, for the vibration motors needed to vibrate based on the distances collected. As the Raspberry Pi Zero W was positioned on the glove, and the goal of the project was complete mobility for the user, a wireless connection needed to be made. Radio frequency communication, or rfcomm, was tried, and when tried, a Serial Port Profile needed to be created and set up. It was created, but when it was time to set it up, the software used did not open on the Raspberry Pis. A question was posed on the Raspberry Pi forums about Bluetooth communication, and there were many Bluetooth Socket Communication tutorials created. The Bluedot Communication API was among them, and it provided a function for sending a string of syntax from Raspberry Pi to Raspberry Pi. All that needed to be done was pairing the two Pis over Bluetooth, then the data could be sent over. On the Raspberry Pi A+, as all the distances were collected, they were converted into strings. Then, each of the strings made were concatenated, and assigned to a single variable. That variable was able to be sent over, every five seconds, to the Raspberry Pi Zero W, using indexing, the Raspberry Pi Zero W's Python Program was able to break apart the string, and vibrate to motors accordingly. This completed the goal of providing a comfortable device that gives complete mobility through wireless communication, with Computer Vision AI to detect real-world objects.

Conclusion

With the empathy and compassion for the visually impaired community, the Object Detection Universal System Plus prototype was created to maintain safety and security. O.D.U.S.+ provides the much-needed independence for the visually impaired community, as it reduces or eliminates the need for canes and other assistive technologies. After many days and nights, cable stripping, programming, sewing, and connecting, a working prototype was made. A patent for O.D.U.S.+ is in progress. With more enhancements and testing to the prototype, the hope is that O.D.U.S.+ will be manufactured and sold at a larger scale, as a cost effective and functional product to aid the visually impaired demographic.

Acknowledgements

I would like to express my undying gratitude to my parents, who always helped me when I got stuck, and provided me with the work ethic and drive to create and complete this project, Ms. Barker, the Science Department leader at Cheshire High School, for making this years’ CT Science and Engineering Fair possible, my sister for helping me express my findings in an eloquent and scholarly fashion.

References

Barela, Anne. “Speech Synthesis on the Raspberry Pi.” *Adafruit Learning System*, learn.adafruit.com/speech-synthesis-on-the-raspberry-pi/programs.

“Blue Dot¶.” *Blue Dot - Bluedot 1.3.2 Documentation*, bluedot.readthedocs.io/en/latest/index.html.

Bluetooth Pairing of Two Raspberry Pis - Raspberry Pi Forums, www.raspberrypi.org/forums/viewtopic.php?t=182681.

“Can We Connect Ultrasonic Sensors Wirelessly to Raspberry Pi? If Yes, How?” *Quora*, www.quora.com/Can-we-connect-ultrasonic-sensors-wirelessly-to-Raspberry-Pi-If-yes-how.

Espeak Alsa Issue - Raspberry Pi Forums, www.raspberrypi.org/forums/viewtopic.php?t=242842.

Espeak TTS ALSA Unknown PCM Cards - Raspberry Pi Forums, www.raspberrypi.org/forums/viewtopic.php?t=136974.

“Festival.” *Festvox*, www.festvox.org/festival/index.html.

“Five Ways to Run a Program On Your Raspberry Pi At Startup.” *Dexter Industries*, www.dexterindustries.com/howto/run-a-program-on-your-raspberry-pi-at-startup/.

“Gpiozero¶.” *Gpiozero*, gpiozero.readthedocs.io/en/stable/index.html.

How to Connect 2 Raspberry Pi's via Bluetooth - Raspberry Pi Forums, www.raspberrypi.org/forums/viewtopic.php?f=36&t=247422&p=1512040#p1512040.

How to Use Pyserial on Raspberry Pi - Raspberry Pi Forums, www.raspberrypi.org/forums/viewtopic.php?f=32&t=247822&p=1513691#p1513691.

Linuxize. “How to Install Xrdp Server (Remote Desktop) on Raspberry Pi.” *Linuxize*, Linuxize, 5 Dec. 2022, linuxize.com/post/how-to-install-xrdp-on-raspberry-pi/.

MarcelJason, Jason, and Bluetooth SIG. “Why the Future Is Blue.” *Bluetooth® Technology Website*, 12 Feb. 2022, www.Bluetooth.com/blog/why-the-future-is-blue/.

Real Python. “Python ‘While’ Loops (Indefinite Iteration).” *Real Python*, Real Python, 20 June 2022, realpython.com/python-while-loop/.

Use App to Connect to Pi via Bluetooth. - Raspberry Pi Forums, www.raspberrypi.org/forums/viewtopic.php?p=947185#p947185.

B, Andy, et al. “Fusion 360 Snap Fit Cases: 3D-Printable Raspberry Pi Case.” *Product Design Online*, 10 Jan. 2022, https://productdesignonline.com/fusion-360-tutorials/fusion-360-snap-fit-cases-3d-printable-raspberry-pi-case/.

EdjeElectronics. “Tensorflow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/raspberry_pi_guide.MD at Master · EdjeElectronics/Tensorflow-Lite-Object-Detection-on-Android-and-Raspberry-Pi.” *GitHub*, 13 Dec. 2020, https://github.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/blob/master/Raspberry_Pi_Guide.md.

Funincomplete, and Author funincomplete. “How to Make Raspberry Pi Speak.” *Fun Incomplete*, 19 Apr. 2020, https://www.funincomplete.com/how-to-make-raspberry-pi-speak/.

Opencv. “HowToUsePerfTests · Opencv/Opencv Wiki.” *GitHub*, https://github.com/opencv/opencv/wiki/HowToUsePerfTests.

Opencv. “Opencv/Opencv: Open Source Computer Vision Library.” *GitHub*, https://github.com/opencv/opencv.