VOICE-CONTROLLED SIGNAL LIGHTS VEST FOR CYCLISTS FOR ENHANCED SAFETY AND COMMUNICATION ON THE ROAD

Arjay T. Añoso

Arjay S. Beligañio

John Rafael H. Brigildo

Southern Luzon State University

College of Engineering

May 2024

ABSTRACT

Title: Voice-Controlled Signal Lights Vest for Cyclist for Enhanced Safety and Communication on the Road

Authors: Arjay T. Añoso, Arjay S. Beligañio, and John Rafael H. Brigildo

Adviser: Engr. Madonna D. Castro

Cycling safety is an important concern, particularly in low-light conditions when cyclists are more prone to accidents because other road users are less aware of their presence. In order to address the problems of signaling while cycling. A voice-controlled signal lights vest that can be used by the cyclist to enhanced safety and communication on the road. The study introduced a voice-controlled signal lights vest for cyclist which can detect specific voice command from the user and generate signal lights display such as turn, slow down, brake signal and off command to turn off the current signal lights display. Machine learning was used to develop a model that recognizes the specific command that communicates with the INMP441 MEMS Microphone for capturing voice input so that the microcontroller could detect the spoken command. The LED Matrix was programmed in the C++ programming language to display the signal lights. The study's research method is applied to address the problem of visibility and use of hand signals. For the evaluation, the device is efficiently responded on the voice command of the user using their normal voice or high voice with the distance of 2-inch from the microphone. The proposed study will enhance the safety and communication on the road.

Keywords: Cycling Safety, Machine Learning, Microcontroller, Signal Lights Vest, Voice Command

Chapter 1

INTRODUCTION

Bicycles are recognized vehicles under the law, they share the road with other vehicles, just like motorists. Cyclists have the same rights and responsibilities as drivers when it comes to traffic laws and guidelines. Nowadays, many individuals use it for recreational activities and also for modes of transportation.

Bicycle use in the Philippines has been on the rise in recent years, with a 2023 poll by the Social Weather Stations (SWS) finding that 36% of households in the whole country now had at least one person who cycles. This indicates a significant increase from the 29% of families that reported riding bicycles in 2022. On the other hand, cyclists must be aware that riding on a bicycle might provide risks in their lives. 33 of the 2,397 cyclists involved in accidents in Metro Manila were dead, according to the 2021 MMDA report on bicycle-related traffic crash data. The data thus emphasizes the need of the cyclist's visibility on the road.

Cycling has various health and environmental benefits, however, there are major issues of concern for cyclists, one is lack of visibility particularly in low-light situations and weather conditions. Being visible on the road is essential for cyclists because it avoids incidents for cyclists and enhances communication with other road users. Another one is using hand signals that are more vulnerable to incidents.

Providing a means for proper visibility and safety for cyclists can make the road a better place for all road users. With this in mind, the researchers decided to develop a voice-controlled signal light vest for cyclists to enhance the safety and communication on the road.

Background of the Study

For cyclists on the road, visibility to other drivers is essential. Bicycles run a higher chance of not being easily seen by cars in low-light conditions like night, morning, or twilight. Accidents are more likely when there are risky circumstances brought on by this lack of visibility. Similarly, visibility may be severely limited in bad weather, such as heavy rain and fog, which makes it considerably more difficult for cars and cyclists to see one another on the road. Cyclist rules often mandate that cyclists riding at night or in low-visibility conditions carry reflectors, a red rear light, and a white front light. Following these laws is necessary to guarantee safety and lower the chance of accidents.

Putting on reflective gear and appropriate lighting is an essential first step in increasing cyclist visibility on the road, especially in low-light conditions. But it's crucial to understand that, particularly when bicycles are turning, relying just on illuminated clothing won't be sufficient to prevent accidents. In such cases, further safety measures like hand signals are even more prone to result in accidents since the hand is left on the handlebar. It is impossible to guarantee that using hand signals and bright clothes will reduce the chance of cycling traffic incidents.

The developers came up with an innovative way to address the problem by developing a voice-controlled signal light vest for cyclist. This vest will feature a microphone and a signal light system, enhanced with machine learning capabilities to recognize the user's voice commands. It stands out as an instance of innovation in the area of road safety. Cyclists, who are typically vulnerable in congested areas, benefit immensely from new technologies that improve their visibility and communication with other road users. This device, intended to be a useful addition to a cyclist's equipment, provides an innovative approach to solving safety concerns while promoting a sense of security and confidence on two wheels. This article delves into the development process of this revolutionary cycling accessory, shedding light on the path to safer and more connected cycling experiences. The objective is to address the problem of low visibility during cycling and improve communication with other road users. Not only will this approach satisfy the current problem, but it also brings up new possibilities for future researchers to look at more developments. The possibility for adjustments and discoveries in this subject is endless, and this study can serve as a basis for more thorough research in the future.

Objectives of the Study

To develop a signal lights vest that uses voice commands to address the problem of low visibility, and the use of hand signals while cycling. Specifically, the study aims to:

1. To design a signal light vest considering the materials and plan design of the system.

2. To create a circuit for the signal lights vest that responds to the voice commands of the user.

3. To develop a program that performs the signal lights system depending on the cyclist's voice command.

4. To evaluate the efficiency of the voice-controlled signal lights vest.

Significance of the Study

The typical method of using lighting and reflective gear to be seen on the road and using the proper hand signal. However, this is not enough to minimize the risk of accidents due to various causes including low light, lack of communication and weather conditions, and poor safety of hand signals. Therefore, the findings from this study would be beneficial to cyclists, other road users, developers, and future developers.

For cyclists, the system will assist them to be properly visible on the road by generating a signal light using voice recognition commands.

For other road users, the system will allow them to see and anticipate bicycle movements, reducing the risk of accidents.

For future developers, this study may be applied as a reference for the development of further research and study.

Scopes and Limitations

The study focused on developing a working prototype that is able to generate a signal light system using a voice command based on the user and the device can be detachable to the vest, so that the vest will be washable. This phase pertains to the creation of the design of hardware and software components of the device using sensors, LED's, microcontrollers, and connectivity features to evaluate the usability of the system.

The system will limit only three types of signal lights, such as left and right signal lights, stop signal lights, slowing down signal lights, and have an off-feature command to turn off the signal light display. Also, the study will not extend to the advanced factors identifying any other types of voice commands/words from the user.

Definition of Terms

For a better understanding and interpretation of this study, the following terms are operationally defined:

Bicycle is a vehicle having two wheels, one behind the other and driven by pedals, steering with handlebars attached to the front wheel.

Cyclist refers to those who ride bicycles, whether for recreational or modes of transportation.

LED matrix or LED display is a large-format, low-resolution dot-matrix display that finds application in hobbyist human-machine interfaces as well as industrial and commercial information displays. Its components are a 2-D diode matrix with their anodes joined in columns and their cathodes combined in rows (or vice versa).

Machine Learning is described as the field of artificial intelligence (AI) that gives computers the capacity to forecast and find patterns automatically from data and past experiences with little human involvement.

MEMS Microphones are a kind of microphone that translate sound waves into electrical impulses by use of a small MEMS sensor. Small in size, low power consumption, and excellent audio recording are hallmarks of these microphones.

Microcontroller is a small computer on a single integrated circuit that regulates human interactions and lighting patterns to operate the Signal Light Vest.
Signal Light Vest refers to a specialized wearable garment equipped with integrated lighting components such as LEDs, fiber optics, or electroluminescent panels intended to improve the wearer's visibility and safety—often a cyclist.

Voice Command is a method of interaction where the user provides verbal instructions or commands to control and operate the functions of the Signal Light Vest. This can include actions like turning lights on or off, changing lighting patterns, or activating safety alerts using spoken words.

Chapter II

REVIEW OF RELATED LITERATURE AND RELATED STUDIES

This chapter reviews the related literature and studies of different published books and research, as well as internet sources, to provide a comprehensive understanding of this study. Furthermore, the gathered information will serve as the basis for the conceptual framework.

The literature review is an integral part of every research work. It enables researchers to get a thorough grasp of their area of interest, uncover gaps in the literature, and contextualize their research within the larger body of current knowledge. By conducting a thorough literature review, researchers can ensure that their study is well-informed and that it makes a significant contribution to the field. (Martell, 2019, p. 1).

The review of related literature and study serves several important purposes. First, it helps the researcher to develop a deep understanding of the research topic. Second, it helps the researcher to identify any gaps in the existing knowledge that their study can address. Third, it helps the researcher to develop a theoretical framework for their study. Fourth, it helps the researcher to position their study within the broader context of the field. This means that the researcher should not simply summarize the existing literature, but should also evaluate its strengths and weaknesses. The researcher should identify any areas where the existing literature is incomplete, contradictory, or outdated. The researcher should also identify any opportunities to extend or refine the existing knowledge. (Creswell, 2018).

ESP32-S3-DevKitC-1

Complete Wi-Fi and Bluetooth Low Energy functionalities are integrated into the general-purpose Wi-Fi + Bluetooth® Low Energy MCU module, ESP32-S3-WROOM-1, ESP32-S3-WROOM-1U, or ESP32-S3-WROOM-2, which is included into the entry-level development board ESP32-S3-DevKitC-1.

For simple interface, the majority of the I/O pins on the module are split out to the pin headers on both sides of this board. ESP32-S3-DevKitC-1 may be mounted on a breadboard or peripherals can be connected using jumper wires.

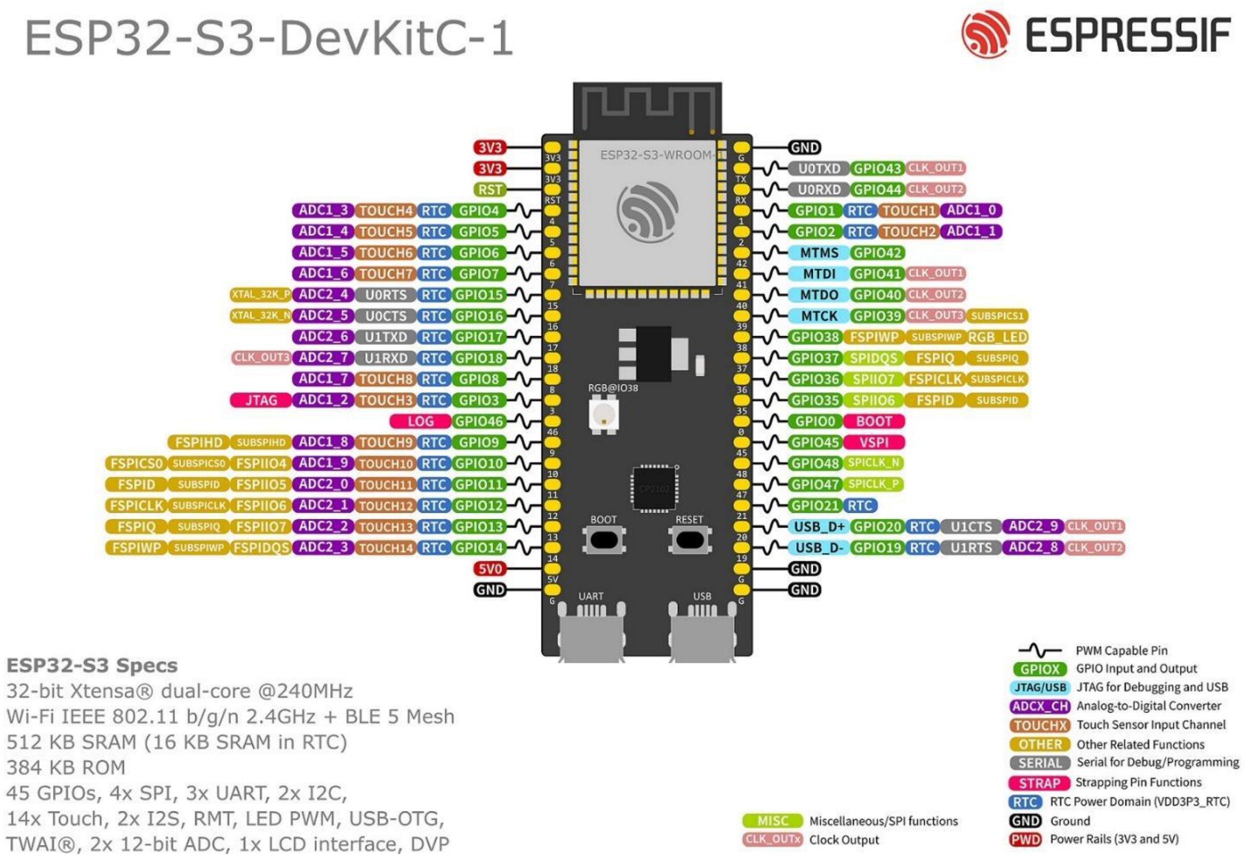

ESP32-S3-DevKitC-1
ESPRESSIF
ESP32-S3-WROOM-1
3V3
3V3
RST
ADC1_3 TOUCH4 RTC GPIO4
ADC1_4 TOUCH5 RTC GPIO5
ADC1_5 TOUCH6 RTC GPIO6
ADC1_6 TOUCH7 RTC GPIO7
XTAL_32K_P ADC2_4 U0RTS RTC GPIO15
XTAL_32K_N ADC2_5 U0CTS RTC GPIO16
ADC2_6 U1TXD RTC GPIO17
CLK_OUT3 ADC2_7 U1RXD RTC GPIO18
ADC1_7 TOUCH8 RTC GPIO8
JTAG ADC1_2 TOUCH3 RTC GPIO3
LOG GPIO46
FSPIHD SUBSPIHD ADC1_8 TOUCH9 RTC GPIO9
FSPICS0 SUBSPICS0 FSPIIO4 ADC1_9 TOUCH10 RTC GPIO10
FSPID SUBSPID FSPIIO5 ADC2_0 TOUCH11 RTC GPIO11
FSPICLK SUBSPICLK FSPIIO6 ADC2_1 TOUCH12 RTC GPIO12
FSPIQ SUBSPIQ FSPIIO7 ADC2_2 TOUCH13 RTC GPIO13
FSPIWP SUBSPIWP FSPIDQS ADC2_3 TOUCH14 RTC GPIO14
5V0
GND
GND
U0TXD GPIO43 CLK_OUT1
U0RXD GPIO44 CLK_OUT2
GPIO1 RTC TOUCH1 ADC1_0
GPIO2 RTC TOUCH2 ADC1_1
MTMS GPIO42
MTDI GPIO41 CLK_OUT1
MTDO GPIO40 CLK_OUT2
MTCK GPIO39 CLK_OUT3 SUBSPICS1
GPIO38 FSPIWP SUBSPIWP RGB_LED
GPIO37 SPIDQS FSPIQ SUBSPIQ
GPIO36 SPIIO7 FSPICLK SUBSPICLK
GPIO35 SPIIO6 FSPID SUBSPID
GPIO0 BOOT
GPIO45 VSPI
GPIO48 SPICLK_N
GPIO47 SPICLK_P
GPIO21 RTC
USB_D+ GPIO20 RTC U1CTS ADC2_9 CLK_OUT1
USB_D- GPIO19 RTC U1RTS ADC2_8 CLK_OUT2
GND
GND
RGB@IO38
BOOT
RESET
UART
USB
ESP32-S3 Specs
32-bit Xtensa® dual-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz + BLE 5 Mesh
512 KB SRAM (16 KB SRAM in RTC)
384 KB ROM
45 GPIOs, 4x SPI, 3x UART, 2x I2C,
14x Touch, 2x I2S, RMT, LED PWM, USB-OTG,
TWAI®, 2x 12-bit ADC, 1x LCD interface, DVP
MISC Miscellaneous/SPI functions
CLK_OUTx Clock Output
PWM Capable Pin
GPIOX GPIO Input and Output
JTAG/USB JTAG for Debugging and USB
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
STRAP Strapping Pin Functions
RTC RTC Power Domain (VDD3P3_RTC)
GND Ground
PWD Power Rails (3V3 and 5V)

Fig. 1 ESP32-S3-DevKitC-1 pinout

INMP441 MEMS Microphone

The INMP441 is a bottom-ported, digital-output, omnidirectional, high-performance, low-power MEMS microphone. A MEMS sensor, signal conditioning, an analog-to-digital converter, power management, anti-aliasing filters, and an industry-standard 24-bit I²S interface make up the whole INMP441 system. Without requiring an audio codec inside the system, the INMP441 can interface directly to digital processors like DSPs and microcontrollers thanks to the I²S interface. Near field applications will find the INMP441 to be a great option because of its high SNR. A flat wideband frequency response of the INMP441 produces very intelligible natural sound.

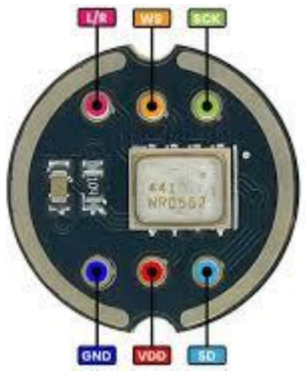


Fig. 2 INMP441 MEMS Microphone

LED Matrix

An LED matrix is a display device that is made up of a grid of light-emitting diodes (LEDs). These LEDs are arranged in rows and columns, forming a matrix structure. Each LED can be individually controlled to emit light, allowing the matrix to display patterns, text, or images.

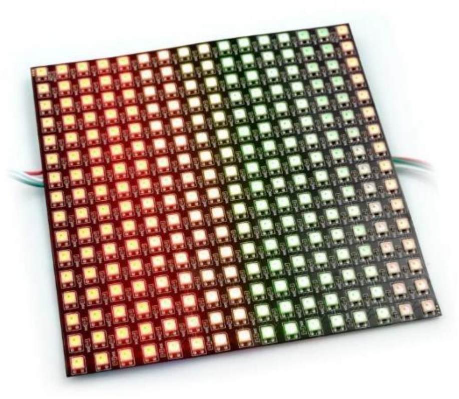


Fig.3 LED Matrix

Cyclist Vest

A cyclist vest is a versatile and practical piece of cycling clothing that can help cyclists to stay warm and comfortable in cool or windy weather. It is typically made of a breathable fabric with a fitted design, and can be worn over a variety of other cycling clothing. Cyclist vests are also often available in reflective or fluorescent colors to help cyclists be seen by motorists in low-light conditions.

Fig.4 Cyclist Vest

Visibility

Increased visibility for pedestrians and cyclists is crucial. One approach to make oneself more noticeable is using lighting. While street lighting may assist motorists in identifying these vulnerable road users, regions of low contrast, glare, and black patches can result from uneven illumination.

High-Visibility Colors

Strongly visible colors are very helpful in raising safety. The colors neon orange, yellow, and green facilitate easy identification of both persons and things. Accidents may be less and awareness raised as result.

Uniforms in neon orange, green, and yellow provide the most possible visibility in dimly lit areas. These hues are readily recognizable under different lighting situations and show out well against backgrounds.

Every material, fluorescent or retroreflective, has benefits. Fluorescent colours visually stand out more, yet retroreflective material works better in low light.

Backpack weight

When considering how much weight to carry for bikepacking, especially for a day trip, it's recommended to keep your backpack load between 2.5 to 3 pounds. This guideline allows for flexibility depending on the duration and nature of your ride. For shorter rides, you can opt to carry less, focusing on essential items like food and other necessities. The key takeaway is that keeping your backpack light enhances your performance during the ride due to reduced strain on both you and your bike.

Traffic Signals

Placed along, alongside, or above a roadway, traffic control signals direct, warn, and control the flow of traffic, which includes motor vehicles, motorcycles, bicycles, pedestrians, and other road users. You may perform a protected turn by using the green arrow that points right or left; as long as the green arrow is lighted, approaching cars, cyclists and pedestrians are halted by a red light. Just like a stop sign, a flashing red signal light indicates to halt! Once stopped, go forward when it is safe and follow the laws of right-of-way. A yellow warning light flashing indicates to proceed carefully. React slowly and with particular attention.

Machine Learning

Within artificial intelligence (AI), machine learning is the use of algorithms trained on data sets to produce self-learning models that can categorize information and forecast results without human assistance. These days, a multitude of business uses for machine learning include text translation, stock market prediction, and product recommendations based on prior purchases.

Because machine learning is so prevalent for AI applications in today's environment, the phrases "machine learning" and "artificial intelligence" are often used synonymously. The two words are, however, essentially different. Machine learning, in particular, is the application of algorithms and data sets to the overall endeavor to build computers with cognitive capacities similar to those of humans.

Spectrogram

A spectrogram is a graphic representation of a signal's "loudness," or intensity, across time at different frequencies included in a certain waveform. Not only is there more or less energy at, say, 2 Hz versus 10 Hz visible, but energy levels also change with time. Spectrograms are widely used in different disciplines to show frequencies of sound waves generated by people, machines, animals, whales, aircraft, etc., as captured by microphones.. The frequency content of continuous signals collected by individual or groups of seismometers is being examined more and more in the seismic world using spectrograms to assist identify and classify various kinds of earthquakes and other ground movements.

RELATED LITERATURE

VeloCity: Using Voice Assistants for Cyclist to Provide Traffic Reports

G. Salvino et al. (2021) design, development, and evaluation of VeloCity, an application for documenting bicycle-related traffic occurrences and infrastructure. Participants preferred to utilize the operating system's voice assistant since it was the least obtrusive of the three input modalities (touch, in-app speech recognition, and the voice assistant) compared by the authors. Participants favored brief instructions over conversational sentences, they also discovered. Using their findings, the writers offered five recommendations for creating voice user interfaces for bikers.

Rangan et al. (2018), titled "Voice Controlled Smart Helmet," presents a comprehensive exploration of integrating voice control technology into motorcycle helmets. The user has a lot more alternatives to operate the car than to stray from driving thanks to the speech module that controls the visor, turn indicator, lamps, horn, and ignition system. The travel will be spent awake by the user since they utilize voice commands to complete the tasks.

Nordmark Anton (2019), stated that traffic is a complex environment in which many actors take part; several new technologies bring promises of reducing this complexity. But thus far, these new developments—among them Cooperative Intelligent Traffic Systems (C-ITS) and its components of human-computer interaction—have somewhat ignored bikers, a particularly vulnerable road user group. This master's thesis in industrial design engineering uses a unique application of bone conduction headphones (BCH) via sensations of both sound and touch to offer five multimodal collision warning signals for cyclists—future ones in these projected C-ITS. The thesis study was carried out as an auxiliary component of the RISE Interactive-organized bigger research project "V2Cyclist." Creating a physical prototype in the shape of a cycling helmet and a matching human-computer interface, V2Cyclist set out to modify the wireless V2X-protocol for cyclists.

LifeLight: Wearable Active Hazard Detection System for Urban/Suburban Nighttime Cyclists

N. Hinson et al. (2019) use Arduino microcontrollers, which, in conjunction with HC-05 Bluetooth modules, facilitate the communication necessary for the system's operation. The HC-05 modules were configured in a master-slave setup, allowing one module to transmit signals while the other received them. One Arduino was equipped with a LiDAR and a logic converter. The TF-mini LiDAR sensor has a range of 12 meters in ideal conditions, but in reality, it works well up to 10 meters in the dusk and nighttime lighting conditions with less than 1% error. The LiDAR sensor works best when it is aimed at a surface that is at an angle of 60 degrees or less. To avoid detecting vehicles that are not a danger to the rider, the LiDAR is set to only detect vehicles that are directly behind the rider. The LiDAR has a very narrow beam, so it will not detect oncoming vehicles unless they are on a collision course with the rider. This system is to decrease the possibility of collision and visibility for cyclists and alerting them to vehicles approaching from behind.

LED Bike Safety Vest

FAHMIDDIN, A. W. Z. B. (2023) This study is intended to develop a safety vest for bikes using a gyroscope system. Wearable device that is designed to improve the visibility and safety of cyclists. The vest is equipped with a variety of LED lights that can be programmed to flash in different patterns, making the cyclist more visible to other road users. The LED in the vest automatically lights up depending on what the gyroscope system indicates, if you turn right the LED indicator shows your turning right and vice versa.

Maroma A. (2018) Development of Brake Lights and an Indicator for Motorcycles. The goal of the project was to design a brake light and motorbike indication system that would be included into a typical motorcyclists jacket. The idea of the gadget was to make the cyclist more visible, particularly at night. The gadget may be included into regular bikes with very little changes to the lighting system thanks to the way the system was designed. For convenience of maintenance, the gadget was also built utilizing materials that are easily accessible in the mainstream electronics industry.

Fadzil, A., Jalaludin, N. A., & Sadun, A. S. (2022) suggested the blindspot detection system. The system uses ultrasonic sensors to detect nearby obstacles in the cyclist's blind spot area. When an obstacle is detected within 10 meters, the system alerts the cyclist with an LED light. When an obstacle is detected within 2 meters, the system alerts the cyclist with a buzzer. This gives the cyclist more time to avoid a collision.

RELATED STUDY

Cycling may be a great way to get exercise and take in the environment, whether you're commuting to work or just having fun around the community. Riding on the road, however, is risky as you never know whether cyclists or pedestrians will be paying attention to your turn.

According to Dulo, J. et al (2022), the signal light would consist of an automatic voice turn and a manual switch that could be mounted on your bicycle. The system’s design is focused on using voice recognition technology to activate signal lights on a vest worn by the cyclist, allowing them to make turns without having to raise their hands. The system is designed by using an Elechouse V3 module to train their voice recognition AI and created an AI with a Google Text-to-Speech library. They also used an Arduino Pro Mini/UNO. The collected data sets to assess whether or not the signal lights will turn on after the voice has been recognized. They concluded that a voice-activated signal light system for bicycles is a feasible and effective solution for improving cyclist safety on the road. They found that the system was able to accurately recognize voice commands and activate the signal lights in most cases. They also found that the system was easy to use and control, and that it could be a valuable tool for reducing the risk of accidents involving cyclists. However, they also noted that there were some limitations to the system, such as the need for clear and consistent voice commands and the potential for interference from background noise.

IJSHRE, (2022), suggested a voice-based cycle direction indication. A wireless controller and a wearable gadget connected to an app that do away with the need to remove your hands from the handlebar. To always keep pedestrians informed, the wearable gadget is a foam panel that snaps together and has up to four LED signals: left, right, forward, and stop. Easy to use, the program is named Dabble. Together with automated lights and siren, the device helps bicycle riders indicate their direction. The wireless controller of the wearable gadget makes it unnecessary to remove your hands from the handlebar and is designed to keep pedestrians informed at all times. The research indicates that while their approach is useful for bike riders, it still has to be improved in many areas. After testing the idea on a bicycle, they discovered that it offers automation and spares the rider from having to worry about unneeded, automatable tasks. But they also discovered that the horn automation feature was a bit bothersome since it occasionally honks in traffic without permission.

Smart LED Bike Jacket proposed by Alsalman et al. (2021), the study is about the development and evaluation of a Smart LED Bike Jacket (S.L.B.J). The purpose of the study is to implement a wearable jacket for cyclists that has different LED colors used for turning signals, provides physical alarms through vibration signals when objects are detected near the cyclist, and includes features such as night vision, a display screen, and a portable power bank charging mechanism. The study aims to enhance cyclist safety on the road and reduce the risk of accidents. They also found that the system was easy to use and control, and that it could be a valuable tool for reducing the risk of accidents involving cyclists. However, they also noted that there were some limitations to the system, such as the need for longer life span of the power source for the reason of some high power consumption of LEDs they used in the system.

According to Harshith H., Dr. M L Anitha (2020), highlighting the potential of development and implementation of the Smart Cyclist Jacket to address the challenges faced by cyclists in urban environments. The jacket's innovative design, coupled with the integration of technology such as LED indicators and voice command functionality, aims to improve the cycling experience and reduce the risk of traffic accidents. The Smart Cyclist Jacket, features LED indicators for left, right, and stop signals, which can be controlled through voice commands via the accompanying Android application. The purpose of these features is to enhance the visibility and safety of cyclists, particularly during night time, by providing a hands-free method of signaling and navigation. They found the potential benefits of implementing the Smart Cyclist Jacket, including the reduction of traffic accidents in major cities and the promotion of cycling as a mode of transportation. The authors suggest that the jacket's innovative design and safety features may attract more people to cycling and improve overall safety for cyclists. Also, they found highlights the potential for regulations mandating the use of safety gear for cyclists, similar to the rules for motorbike and car users.

Conceptual Framework

This illustrates the conceptual framework that will guide the researchers on their study.

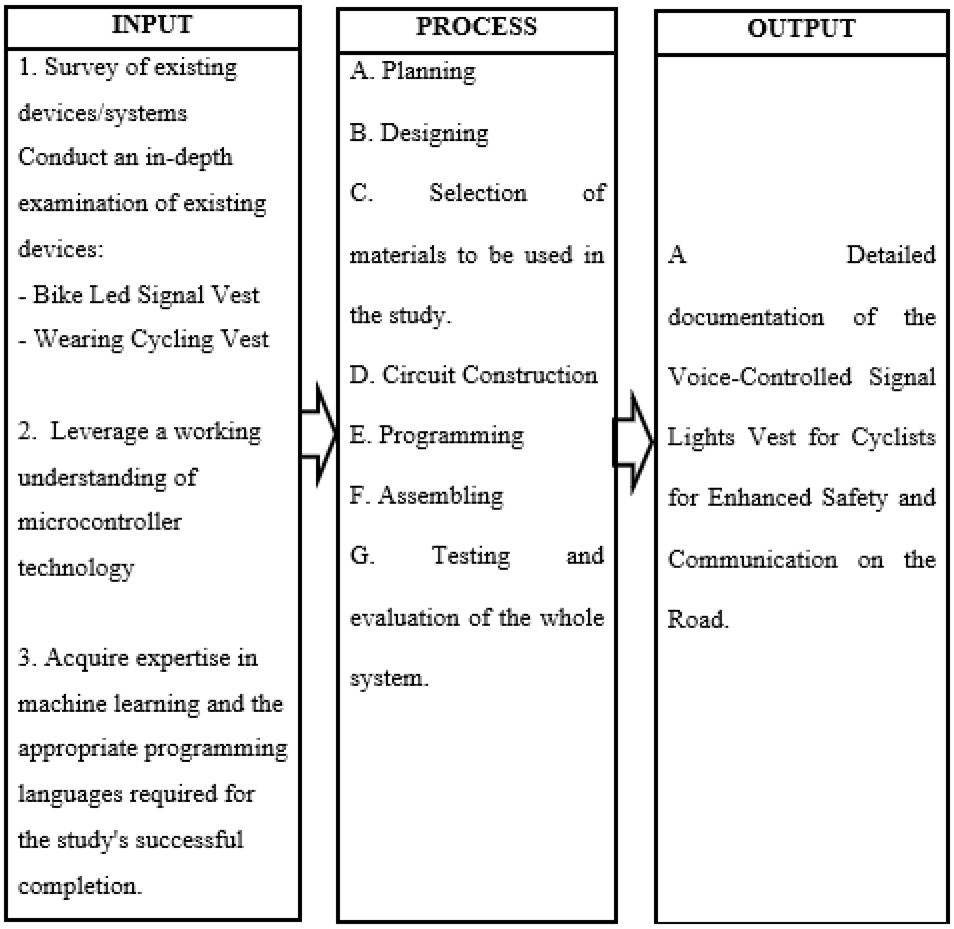

INPUT
1. Survey of existing devices/systems
Conduct an in-depth examination of existing devices:
- Bike Led Signal Vest
- Wearing Cycling Vest
2. Leverage a working understanding of microcontroller technology
3. Acquire expertise in machine learning and the appropriate programming languages required for the study's successful completion.
PROCESS
A. Planning
B. Designing
C. Selection of materials to be used in the study.
D. Circuit Construction
E. Programming
F. Assembling
G. Testing and evaluation of the whole system.
OUTPUT
A Detailed documentation of the Voice-Controlled Signal Lights Vest for Cyclists for Enhanced Safety and Communication on the Road.

Fig. 5 Research Paradigm of the Proposed Design of Signal Lights Vest for Cyclist using Voice Command.

The development of the Signal Lights Vest for Cyclists using Voice Command begins with the input section, where researchers conduct studies on existing devices and utilize their knowledge in microcontroller and programming. The process involves planning, design, material selection, circuit construction, programming, assembly, testing and evaluation of the whole system, with every step contributing to the creation of the system. The output section shows the finished product, the Signal Lights Vest for Cyclists, including a prototype and detailed documentation of the study.

Table 1. Synthesis Table of the Related Literatures and Studies and the Researcher's Statement

Theme of the Study	Author, Year & Title	Related Statements
Design, development, and evaluation of a voice user interface for cyclists	G. Salvino et al. (2021), *VeloCity: Using Voice Assistants for Cyclists to Provide Traffic Reports*	VeloCity is a voice user interface created by the authors for cyclists. Since the voice assistant of the operating system is the least distracting, bikers seem to like using it. Short directives also trump conversational ones.
Integration of voice control technology into motorcycle helmets	Rangan et al. (2018), *Voice Controlled Smart Helmet*	The voice-controlled smart helmet system allows riders to control their motorcycle's visor, turn signals, headlights, horn, and ignition with voice commands. This reduces distractions and helps riders stay focused on the road.
Development of a Wearable Active Hazard Detection System for Cyclists	N. Hinson et al. (2019), *LifeLight: Wearable Active Hazard Detection System for Urban/Suburban Nighttime Cyclists*	The LifeLight system uses a LiDAR sensor to detect vehicles approaching from behind and alerts cyclists with a signal to their smartphone. It has the potential to reduce the possibility of collisions and improve cyclist safety.
Development of a safety vest for bikes using a gyroscope system	FAHMIDDIN, A. W. Z. B. (2023), *LED Bike Safety Vest Using Gyroscope System*	The project is to create a wearable gadget intended to increase riders' visibility and safety. Because the vest has many LED lights that can be set to flash in various patterns, other drivers will see the cyclist more clearly. The gyroscope mechanism tells the vest to light up automatically; if you turn right, the

		LED signals that you are going right, and vice versa.
Development of a motorcycle indicator and brake light system integrated into a standard rider's jacket	Maroma A. (2018), *Development of Motorcycle Jacket with Modified Indicator and Brake Lights*	The objective of the project was to create a brake light and indication system for motorcycles that could be included into a regular rider's jacket to improve visibility, particularly at night. The idea behind the system was to employ easily accessible materials for maintenance and to need as little changes to the conventional motorbike lighting system.
Development of a blind spot detection system for cyclists using ultrasonic sensors	Fadzil, A., Jalaludin, N. A., & Sadun, A. S. (2022), *Blind Spot Detection System for Cyclists*	The system uses ultrasonic sensors to detect nearby obstacles in the cyclist's blind spot area. When an obstacle is detected within 10 meters, the system alerts the cyclist with an LED light. When an obstacle is detected within 2 meters, the system alerts the cyclist with a buzzer.
Signal Lights for Cyclist using Voice and Manual Switch	Dulo, J. et al (2022), *Signal Lights for Cyclist Through Voice Turn and Manual Switch*	Due to the COVID-19 pandemic, many people have switched to a much safer and healthier way of transportation, which is riding a bike or cycling. However, cyclists face the risk of accidents due to the lack of proper signaling when making turns. The study aims to introduce a solution to this problem by developing a product that utilizes voice recognition and manual switch activated bicycle turn signals.
Voice-based direction indicator for cycle	IJSHRE, (2022), *Voice-based direction indicator for cycle*	A voice-based direction indicator system for bicycles was proposed by IJSHRE (2022). The system consists of a wearable device with LED signals and a wireless control. It provides automatic headlight and horn, but the horn automation part needs improvement.

Development and evaluation of a Smart LED Bike Jacket (S.L.B.J)	Alsalman et al. (2021), *Smart LED Bike Jacket (S.L.B.J)*	The wearable Smart LED Bike Jacket (S.L.B.J.) was created to improve bicycle safety by offering night vision, turning signals, physical alerts, and a display screen. Though it used a lot of electricity, the system was simple to operate and manage.
Development and implementation of the Smart Cyclist Jacket	Harshith H., Dr. M L Anitha (2020), *iSmart Cyclist Jacket*	The Smart Cyclist Jacket is a wearable device that uses LED indicators and voice commands to improve the safety and convenience of cycling in urban environments. It has the potential to reduce traffic accidents and promote cycling as a mode of transportation.

Recent research has focused extensively on enhancing cyclist safety through the development of wearable devices and voice-controlled systems. One significant area of innovation is voice-controlled interfaces. G. Salvino et al. (2021) introduced VeloCity, a voice user interface that cyclists prefer due to its minimal distraction and the efficiency of short commands. Similarly, Rangan et al. (2018) developed a voice-controlled smart helmet that allows motorcyclists to manage various bike functions using voice commands, thus reducing distractions and improving road focus. Dulo, J. et al. (2022) addressed signaling risks exacerbated by the COVID-19 pandemic by creating voice-activated and manually switched signal lights for cyclists. Additionally, IJSHRE (2022) proposed a voice-based direction indicator system that includes LED signals and automatic headlight and horn control, although the horn automation requires further improvement.

Wearable safety devices have also seen significant advancements. N. Hinson et al. (2019) developed LifeLight, a wearable hazard detection system that uses LiDAR sensors to alert cyclists to approaching vehicles, potentially reducing collision risks.

FAHMIDDIN, A. W. Z. B. (2023) created an LED Bike Safety Vest equipped with a gyroscope system that automatically signals turns, thereby enhancing cyclist visibility. Maroma A. (2018) integrated indicator and brake lights into a standard motorcycle jacket to improve rider visibility, particularly at night. Alsalman et al. (2021) developed the Smart LED Bike Jacket (S.L.B.J), which features turning signals, physical alarms, night vision, and a display screen, although it has high power consumption. Harshith H. and Dr. M L Anitha (2020) implemented the iSmart Cyclist Jacket, which uses LED indicators and voice commands to improve safety in urban cycling environments.

Furthermore, blind spot and hazard detection systems are crucial for cyclist safety. Fadzil, A., Jalaludin, N. A., and Sadun, A. S. (2022) developed a blind spot detection system using ultrasonic sensors to alert cyclists of nearby obstacles, thus enhancing situational awareness. Overall, this collective research underscores the importance of integrating advanced technologies such as voice control, LED signaling, and sensor-based detection into wearable devices to promote cyclist safety, visibility, and convenience.

Chapter III

RESEARCH METHODOLOGY AND DESIGN

This chapter discusses the methods and procedures used to conduct the research and provides the readers with important information on how the research was made and how the data was obtained to come up with conclusions. The sources of information are also created as references for the study.

Research Locale

The research was conducted in Mauban, Quezon, specifically in Sadsaran (Poblacion), where the Seawall is located. This location was chosen due to its popularity among cyclists for recreational activities and its serene views, providing an ideal setting for conducting interview. The researchers aimed to develop a system that would enhance cyclists' experiences throughout their cycling journeys along the Quezon Highway in Mauban, Quezon.

Respondents of the study

The respondents for this study include cyclists from a variety of backgrounds, including recreational riders, and cycling enthusiasts. Engaging with this varied group of respondents through semi-structured interviews, where questions were raised and participants had the opportunity to elaborate on their significant thoughts and experiences that will provide a comprehensive understanding of cyclists’ practical demands and preferences regarding signal lights.

Research Design

The research design for this study is classified as applied research. This approach is applied to addressing practical issues, specifically focusing on automating the traditional hand signals used by cyclists through the development of a voice command signal light vest to enhance cycling safety in diverse environments. The device to be created will perform the different Signal Lights using the Voice Command System. It aimed to develop a signal lights vest that uses voice command to address the problem of low visibility and the use of hand signals while cycling and additional safety use was added with the use of technology.

Research Instrument

Internet Articles and Journals

Using internet articles and journals the researchers searched online to provide traditional study resources has brought real-time insights and various perspectives. Peer-reviewed publications, which contain in-depth analysis and empirical investigations connected to microcontrollers, have also provided an academic base. This dual approach ensures a deeper understanding by merging current perspectives with traditional research approaches.

Microcontroller Articles

The researchers explored and searched for more related articles about microcontrollers that they used in their devices. These furnished valuable insights into circuit construction and component integration to make it function according to their plans. They determined the selection of specific pins and connections for the successful development of the device.

Published and Unpublished

The researcher analyzed and studied published and unpublished theses related to the development of their device that served as a comprehensive guide, influencing component selection. Their insights extracted from existing devices, their functionalities, societal impact, and potential improvements significantly contributed to the innovative design of the device.

Semi-Structured Interview

The researchers conducted a semi-structured interview to gather information in designing their device. The responses obtained during these interviews are carefully considered, providing valuable inputs to refine and shape the development of the device.

Procedures

The researchers conducted preliminary research to obtain enough information about the different technologies and components that were employed in the Signal Light Vest using Voice Command. The researchers' major source of resource materials is the Internet, as most of the information regarding the topic being studied is available online. The research was done online to establish what programming languages are appropriate for the software part of the research and to assist in developing them in the development of the embedded system for the Signal Light Vest within the time frame specified. Other sources of information discovered in the internet include online documentation, and articles associated with the development of Voice-Controlled Signal Light Vest for cyclist

.

Based on the research's development cycle design, the following steps are followed:

a. Planning

The researcher started by evaluating all of the components that went into developing the "Voice-Controlled Signal Lights Vest for Cyclist". They collect data by analyzing different studies of currently existing devices. They analyzed the components and sensors utilized by other researchers in their study. They used it as a starting point for their research.

b. Designing

Designing our system involves understanding what materials will work best, planning out the layout, and ensuring it's comfortable to use. We start by asking people what they need and like. With valuable feedback, we strategically developed a durable, functional, and user-friendly plan.

c. Selection of materials to be used in the study

Based on thorough evaluation, the researchers prepared a list of possible components for the device. When developing a device, they analyze the cost, availability, and functionality of the components.

d. Circuit Construction

The researcher began constructing a connection of the different components. The components had been incorporated using the microcontroller and each was configured on their pins based on their connections to the other.

e. Programming

The researcher then started working on the research program. They designed a program for a system that would automatically generate a signal light system depending on the voice user's command. The components are connected together and will carry out their functions in an integrated way.

f. Assembling the Prototype

The researchers created the system’s hardware components and converted it into a prototype. They upload the code to the microcontroller that combines the components for the circuit design. They placed the components in the appropriate arrangement and assembled them.

g. Testing and Evaluation of the whole system

After the assembly of the prototype, they tested it to check that it was operating according to its objectives. They apply the testing and debug procedure. They reconstruct the code and assembly once the device does not work, then re-evaluate it. They proceeded with the process involves integrating LED lights into the vest that respond to spoken commands that works through the use of microphone and microphone to interpret and execute the commands which it provides that the whole system works upon on its functionality to generate signal lights.

Chapter IV

RESULTS AND DISCUSSION

This chapter summarized the results of the finished system and the data that was collected. This includes the overview of the system created i.e., the description of the device, flowchart, block diagram, hardware components, and important materials and parts that will be used in the development of the device.

System Overview

Cycling can be risky, specifically when the cyclist is hard to see by other road users or understands their signals by using their hands. That's why it is important to have a device to improve the safety of the cyclist. The researchers developed the Signal Lights Vest using voice command which is a device that will help the cyclist to communicate with other road users without using hand signals, improve visibility on the road, and avoid the risk of accidents.

The device will use a MEMS microphone to be able to get the voice command of the user. The researchers used machine learning to create a model for voice command recognition and convert it into C++ code. These voice commands will be limited only to the cyclist's signals such as turn signal lights, brake lights, slowing down lights, and turn off command. The signal lights will activate when the user starts a command on the microphone. The LED signal lights are incorporated to the vest and it can be detached. When the user says "left" the LED lights will display a blinking left signal and indicate that the user will go left, same as when it says “right” it will display a blinking right signal, “stop” will display a full red as brake signal, “slow” will display a full yellow as slow down signal, and an “off” command to turn off the current signal display.

The device may be used by the cyclist to be able to communicate with other road users without using hand signals and improve visibility on the road. Also, it can be used both during the day and at night, and it is detachable from the vest, allowing the vest to be washed.

Technical Description

This section focuses on the technical details of the system as well as the researchers' activities, in line with the research approach described in the previous chapter. This section also covers the system designs and components that helped the researchers in achieving the objectives of the research.

Planning and Designing

The researchers completed these tasks as part of the system planning and design. The researchers collected reliable data that explained the system, including its capabilities and limitations. They planned to create a system where the user starts the specific commands then, the commands have equivalent signals such as left, right, stop, slow down signals and an off command.

The table below outlines the result of the basis of conversion from traditional way of signaling that were used by the cyclist to automated signal light system.

Table 2. Results of Conversion from Traditional to Automated Signal Light System

Traditional	**Automated**
Hand signal turning left	Voice command Left, blinking left arrowed LED display (Green LED)
Hand signal turning left	Voice command Right, blinking right arrowed LED display (Green LED)
Hand signal stop	Voice command Stop, Round LED display (Red LED)
Hand signal slowing down	Voice command Slow, Round LED display (Yellow LED)

The researchers gathered information on the traditional signaling methods used by cyclists. Cyclists typically use a switch or press a button, which is then converted into a voice command input. Additionally, cyclists use hand signals, which are transformed into LED signal light displays.

The table below shows the results of the interview on the respondents to consider the basis of the design of the device. There was a set of questions asked regarding the design of the device.

Table 3. Results of Interviews on the Respondents for the Design of the Device

Basis	**Solution**

The device is lightweight	The weight of the components and materials is considered that the device weighs less than 1 kilogram
The vest is flexible to any body size	The vest has adjustable straps and garter straps, and also the microphone built in to the adjustable headset which is flexible to wear and to adjust the position.
The vest is comfortable for the body	The vest has padding at the back
The device is designed to be used in light rain	The material used in the vest is a water-repellent fabric
The device has readable display both in size and distance.	The 16x16 LED matrix is used to display signal light pattern
The display color of the device is high-visible	The color for the turning signals is green, slow down signal is yellow, and brake signal is red.

The data gathered by the researchers is shown in Table 3. The researchers decided to consider it as the basis for designing the device and the vest. The data gathered is based on the interview of different cyclists. The researchers focused on several key bases when designing a device for cyclists. First, they ensured that the device was lightweight, taking into account the weight of components and aiming for a weight similar to that of a backpack for a day trip. To make the device suitable for cyclists of all sizes, they made it adjustable with straps and garter straps. Comfort was also a priority, so they added padding to the back of the vest. The researchers decided on a 16x16 LED matrix to provide clear display and guarantee readable signal light patterns. Finally, the researchers chose high-visibility colors like green, yellow, and red for the signals to enhance visibility on the road.

Block Diagram

The block diagram below illustrates how the major components process the audio input signal that send on the microcontroller and display the output.

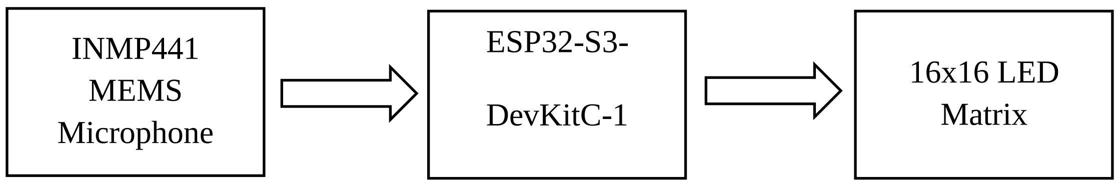


Fig. 6 Block Diagram

As shown in the figure 6, the system starts on the INMP441 MEMS Microphone will serve as an input device to get the command from the user. The microcontroller will process the user's input and check if the command is valid or match in the trained model. If the command is invalid there is no display on the LED matrix. If the command is valid, the microcontroller will get the command id and display the assigned signal pattern for that command on the LED matrix. The LED matrix will serve as an output device of the system.

System Flowchart

The flowchart below illustrates how the device operates within the system. As soon as the switch is turned on, the device starts the process.

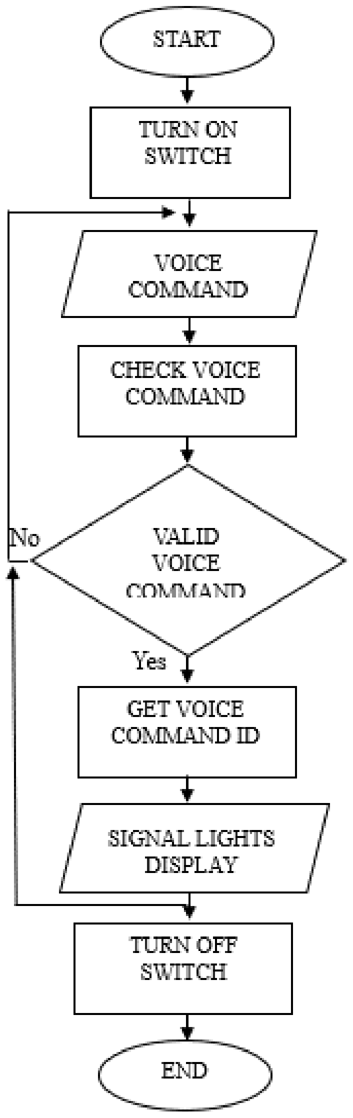


Fig. 7 System Flowchart

The system begins with the user turning on the device switch. Subsequently, the device waits to detect a voice command. Once a voice command is detected, the system checks whether the command is stored within the system. If the command is not stored or invalid, the system fails to recognize the command and there will be no signal light display and it will wait to detect a voice command again. If the system recognizes the command, it retrieves the corresponding voice command ID and the system displays the signal lights. If the user turns off the switch, it will end the process.

Materials Used

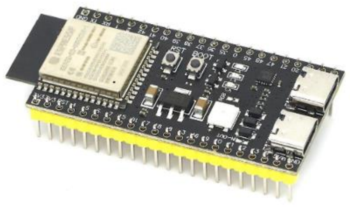
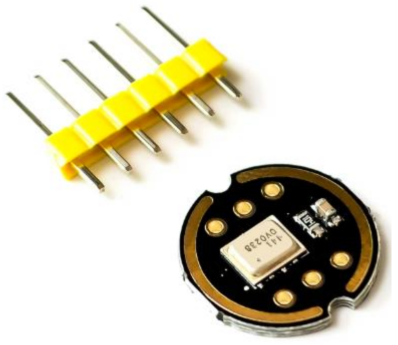
Based on thorough research, the necessary materials have been carefully identified and examined. The researchers have listed the materials suitable for constructing the system. Key factors they considered include functionality, quantity, weight, and power consumption.

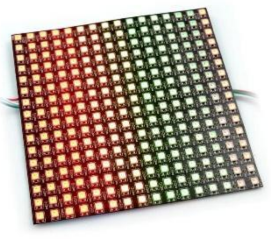
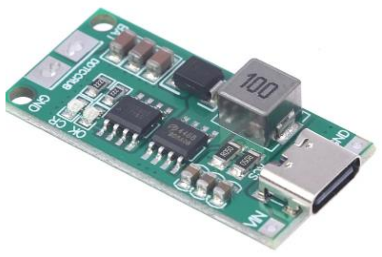
Hardware Implementation

The table below outlines the various components used in developing the device. It provides details on the device's specifications. The third column describes the description of use for the system of each component.

Table 4. Hardware components

Name	Specification	Description of Use for the System

ESP32-S3-DevKitC-1 	Up to 240MHz is the frequency, or clock speed. SPI flash: 8MB (64Mbit) 384 KB is ROM. Ram: 512 KB 2 MB is SPI PSRAM. RTC's SRAM is 16 KB Interactive Interfaces: 45 GPIOs, 4 SPI, 3 UART, 1 USB OTG, 2 x I2C, 2 x I2S Ports: Two USB connections type C Voltage of Supply: 5V via USB Circuit Voltage: 3.0–3.6V Circuit Function	This will act as the system's microcontroller, to which the uploaded program will be sent.
INMP441 MEMS Microphone 	High-Precision 24-Bit Data Interface with Digital IS Broad SNR of 61 dBA Extremely Sensitive at -26 dBFS 60 Hz to 15 kHz Flat Frequency Response Minimal 1.4 mA Current Consumption -75 dBFS high PSR Tiny Surface-Mount Package, 4.72 x 3.76 x 1 mm Power range is from 1.8 to 3.3 V.	This Microphone serve as the device used to gathered the data or input from voice commands of the user.

16x16 LED Matrix 	Type of Light: SMD 5050 RGB WS2812B IC Model 256 LEDs in all. DC 5V Working Voltage 160x160 mm in size	In this device, this LED matrix displays the output which is the left, right, slow down blinking arrows, and the stop full LED displays in terms of the form of the LED lights on display and in terms of what color is specified.
Lithium polymer Battery (2pcs)	Lithium polymer battery type 5000 mAh capacity. Three-seven volts Rating of Wat-Hou: 18.5Wh Weigh: about. 100 gramme NTC (thermistor): 10000 Dimensions: 48 mm x 9.5 mm	This battery is used as the power supply for the LED Matrix and also the other one is for the microcontroller.
Type C 2s Lithium battery charger board 	Two-Cell (2S) 8.4V Version DC 3-6V input voltage (DC 3.7V-5V recommended) One A (1A Version); two A (2A Version); four A (4A Version) input current 8.4V charging voltage Current charge: 0.55A (1A Version); 1.1A (2A Version); 2.2A (4A Version)	This charger board serve as the charging port that use for charging the battery of the device.

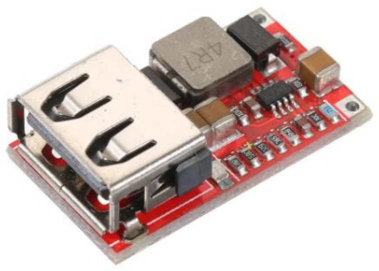
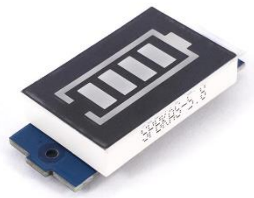
Step down-USB, power supply module 	DC 6–24V input voltage Voutage output: 5.1–5.2V At most, 3A of output current 50.000Hz is the switching frequency.	This module is used as to transfer the power supply from the battery to the microcontroller.
Battery Level Indicator 	Lithium Ion Batteries are Compatible Compatible Configuration of Li-Ion Cell: 2S Valid Battery Voltage Range: 6.6V–8.4V Show Colours: Green Battery Level Segments, Red Battery Outline There are no illuminated segments; voltage is less than 6.6V Voltage > 6.6V, One Segment Illuminated Illuminated Two Segments: Voltage ≥ 7.0V Three Lighted Segments: Voltage ≥ 7.4V Voltage ≥ 7.8V: Four Segments Lighted Show Dimensions: (31.4 x 20 mm) Dimensions of the module: 43.8 x 20 x 8.7 mm. ~2.0mm is the mounting hold diameter. Center to Center spacing of the mounting holes is around 37mm.	The battery level indicator serves as a display to check the battery level of the device. It is useful as the device is being charged, to monitor the battery's status.

Table 4 discusses the major components of the voice-controlled signal lights vest. The ESP32-S3-DevKitC-1 served as the microcontroller of the device where the program was uploaded. This INMP441 MEM Microphone served as the device used to gather the data or input from voice commands of the user. The LED matrix displays the output which is the left, right, slow-down blinking arrows, and the stop full LED displays in terms of the form of the LED lights on display and in terms of what color is specified. The lithium polymer battery is used as the power supply for the LED Matrix and also the other one is for the microcontroller. Type C 2s Lithium battery charger board serves as the port and handles the functionality of charging the battery of the device and the Step down-USB power supply module, one of the components used to transfer the power supply from the battery to the microcontroller. Lastly, is the battery level indicator that will display the status of the battery life of the device specially in charging mode.

To calculate the number of hours that the battery can be use, we can use the formula:

$$\textbf{Battery Life} = \frac{Capacity\ (mAh)}{Consumption\ (mA)}$$

$$\textbf{Battery Life} = \frac{5000\ mAh}{70\ mA + 60\ mA + 20\ mA + 2.2\ mA + 3\ mA}$$

$$\textbf{Battery Life = 32.22 hrs}$$

The two lithium batteries are connected in series so that the capacity is still 5000 mAh and the normal load current according to datasheet of the microcontroller is 70 mA, LED matrix is 60 mA, power LED is 20 mA, microphone is 2.2 mA, and for the battery level indicator is 3 mA, so the total battery life in hours is 32.22.

The table below outlines the software and programming language used for creating the voice command recognition model as well as developing a program to get the voice input command and display the LED patterns of each command signal.

Table 5. Software Implementation

Name	**Specification**	**Description of the System's Usage**
Python	Python Version: 3.10	The researchers used the python programming language to program the voice command recognition model by training and testing the model.
PlatformIO IDE	PlatformIO IDE Version: 3.4.4	The researchers used PlatformIO IDE in this study to create firmware specifically for the ESP32-S3 microcontroller, the PlatformIO IDE used as a critical tool in their development workflow. It is used to manage the many processes involved in firmware development. This involved managing dependencies, structuring the code, and optimizing

		the uploading and compilation procedures.
C++ Language	C++ Version: C++20	The researchers used C++ language for the study to programmed to display each signal light symbol and its corresponding function using code written in the C++ programming language. They were able to use C++'s powerful features and object-oriented capabilities to write effective, modular code by selecting it for ESP32-S3 microcontroller.

The table 5 is the programming software platform that was used for creating a model, writing, compiling, and uploading code to the microcontroller of the system. It includes the Google Colab to create a model for voice command recognition using python programming language, and Platformio IDE to program the microcontroller using C++ programming language.

Schematic Diagram

To be able to create a voice-controlled signal light system, the researchers constructed the circuit plan by connecting all the main components to their appropriate pins.

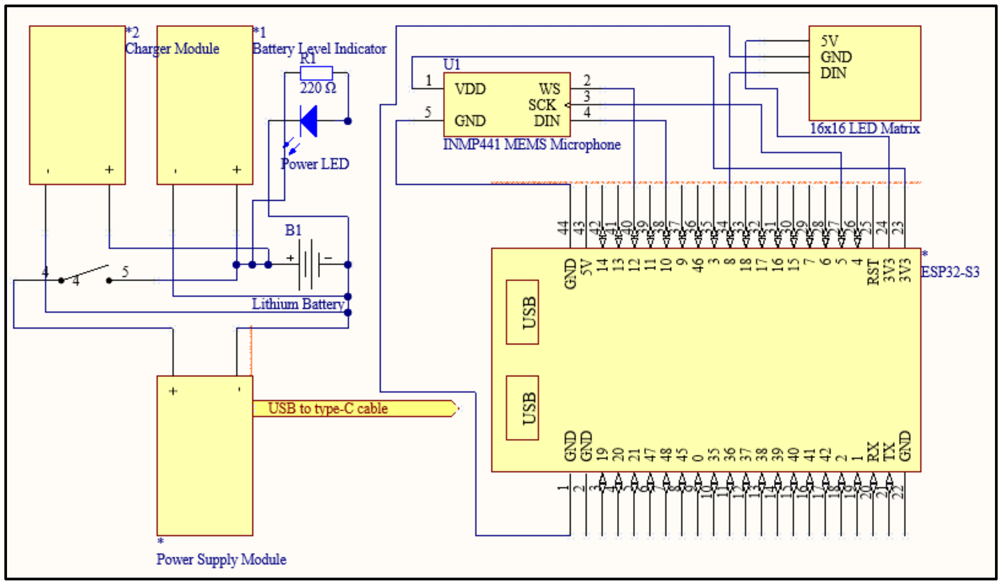
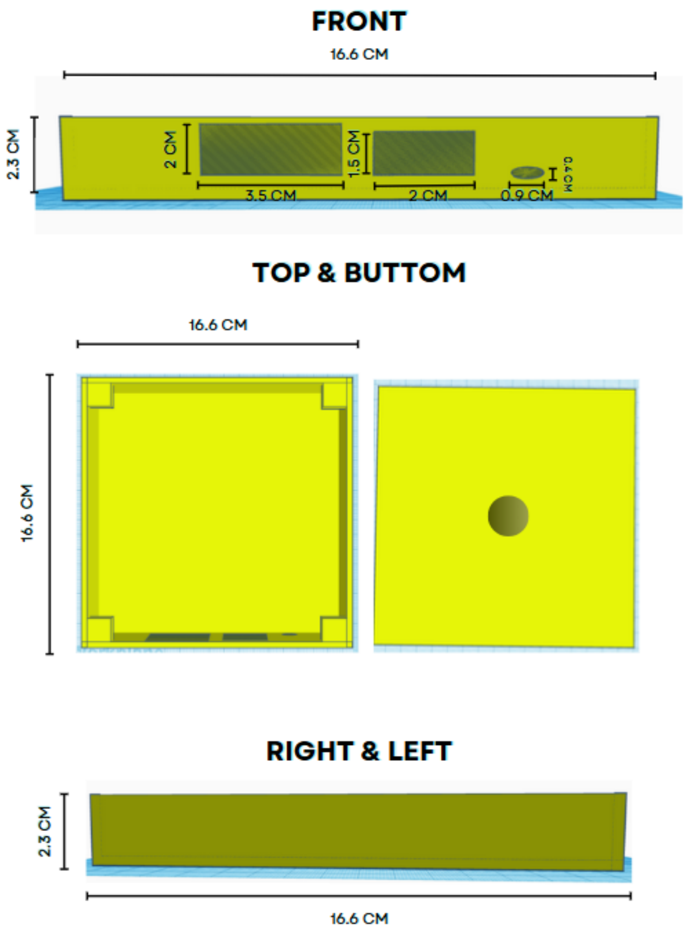


Fig. 8 Device Schematic Diagram

The figure 8 illustrates the connections between the various components and modules used in the system. The ESP32-S3-DevKitC-1 as the microcontroller, with the 16x16 LED matrix connected to pins GPIO_NUM_8 for DIN and INMP441 MEMS Microphone connected to pins GPIO_NUM_10 for DIN, GPIO_NUM_12 for WS, GPIO_NUM_5 for SCK, GND to GND and VDD to 3V3. The power source of the LED Matrix is connected on the microcontroller and microcontroller is connected to the Lithium battery using USB power module and Type-C cable. The lithium battery is connected on the charger module to be able to charge and battery level indicator to determine the battery life. The system had a switch to turn on and off the power and power LED indicator.

Design of Case of the Device

The voice-controlled signal light vest prototype is illustrated in front, top, bottom, right, left, and back views in the following figure 9 and the top, bottom, and front view of the main cover of the device in the following figure 10. The design illustrates the position of holes where plan to put the designated components.



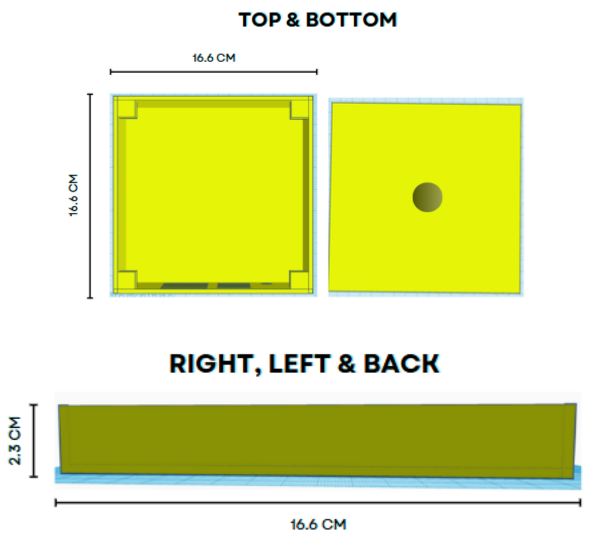


Fig. 9 Design of the Case of Device

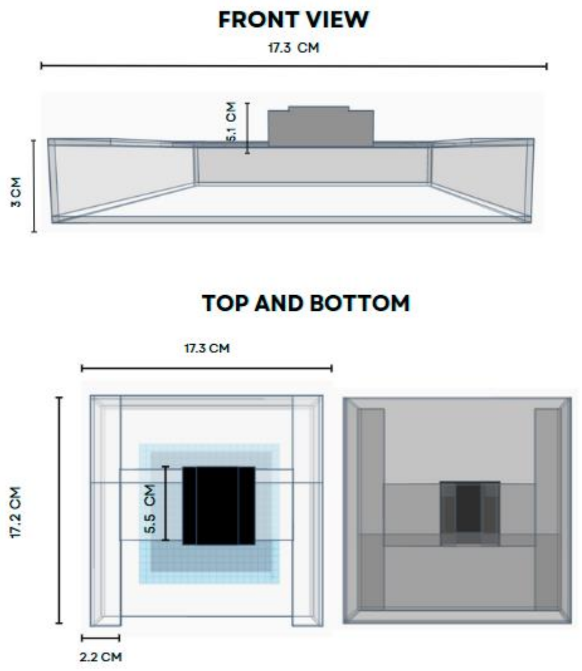


Fig. 10 Design of Main Cover of the Device

Figure 9 and 10 shows the plan design of the prototype of the case of the device in different views. The design of the case shows the position of every hole for the planned position of the battery level indicator, switch, and charging port, and also holes for the connections of the microphone. Moreover, figure 9 shows each measurement of the casing in centimeters.

Design of the vest of the device

The voice-controlled signal light vest prototype is illustrated in front and rear views in the following figure. The design of the vest takes several factors into account.

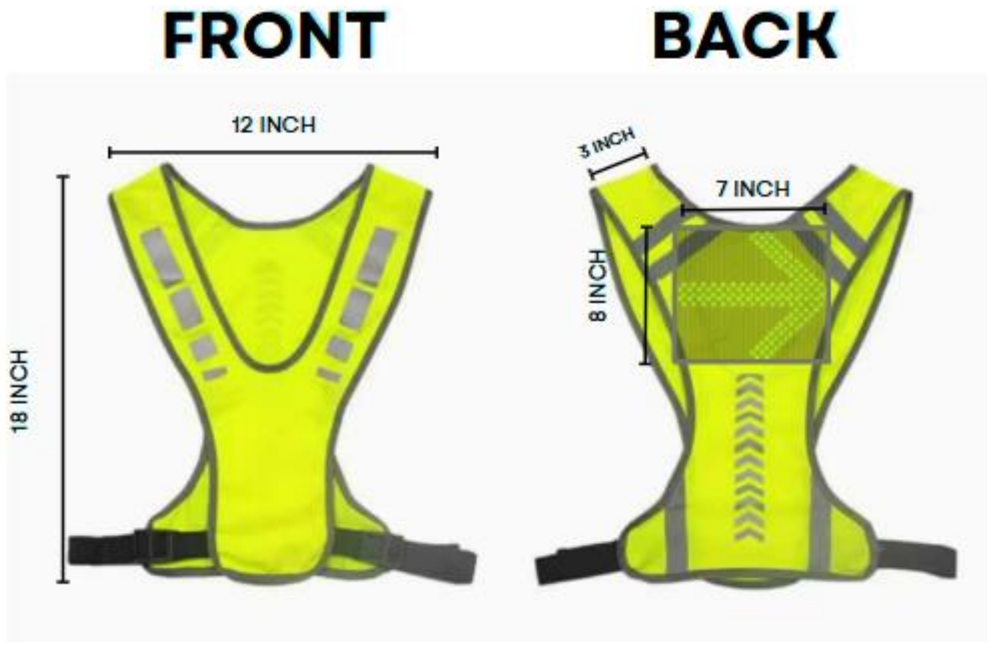


Fig. 11 Design of the Signal Light Vest

The figure 11 shows the plan design of the prototype of vest of the device in front view and rear view and also the pocket and position of the pocket where the main device will put on. The plan measurement of the vest in height is 18 inch long, 12 inch in width, 3 inch for the shoulder strap and for the pocket of the vest, the height is 8 inch and the width is 7 inch. The researchers considered the standard measurements of the backpack within its height and width through searching.

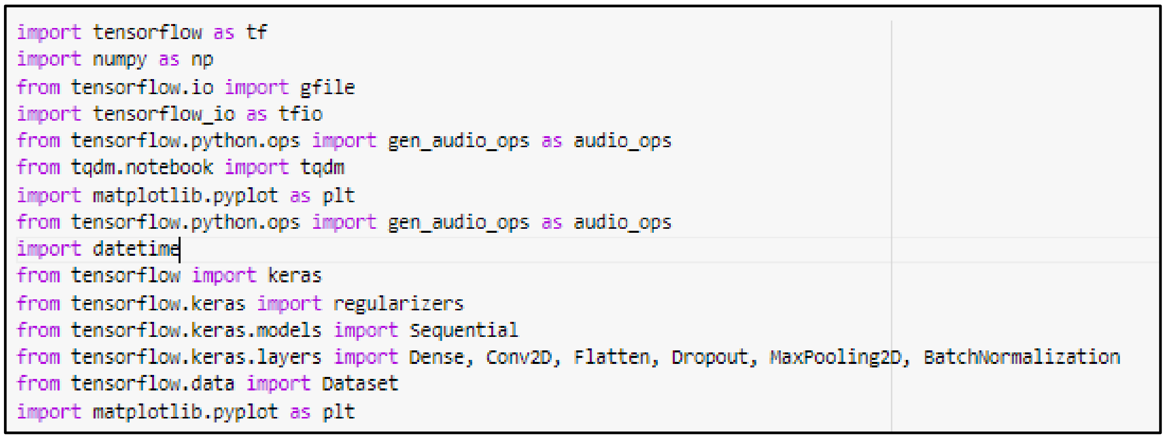
Structure and Organization

This section discusses how the Voice-Controlled Signal Lights Vest was developed. The researcher had planned ahead and used that knowledge to build the system. The process involved developing the system in accordance with previously gathered information about planning. The researcher was guided by the insight gained from prior preparation when able to develop the system. This part specifies the system programming, assembly, testing, and evaluation of the system.

Programming

The researchers develop a model for recognizing the input command of the user using machine learning and write program to display signal pattern based on the command. This program enables user to indicate their directional intentions and presence to other road users through simple voice commands and display the signal through LED patterns.

The figure below shows how to import necessary libraries and modules for the voice command recognition model.



```
import tensorflow as tf
import numpy as np
from tensorflow.io import gfile
import tensorflow_io as tfio
from tensorflow.python.ops import gen_audio_ops as audio_ops
from tqdm.notebook import tqdm
import matplotlib.pyplot as plt
from tensorflow.python.ops import gen_audio_ops as audio_ops
import datetime
from tensorflow import keras
from tensorflow.keras import regularizers
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Conv2D, Flatten, Dropout, MaxPooling2D, BatchNormalization
from tensorflow.data import Dataset
import matplotlib.pyplot as plt
```

Fig. 12 Importing libraries

The code involved imports all the libraries and modules needed for working with audio data, visualizing it, and developing and training neural network models.

In the directory of the files, all of the voice command will get the voice position, length, and if there’s enough audio signal will be checked and measured. Thorough checking and validation of the audio of voice in the files, it will be transform an audio signal into a spectrogram, which is a visual representation of the frequency spectrum of the audio as it varies over time. The spectogram will be the dataset which is for the training, testing and validation dataset. Model will be trained, tested and evaluated through that process.

The figure below shows the functions on how to retrieve all the audio files within a directory and determine the position of the voice within the audio clip base on the specified noise floor.

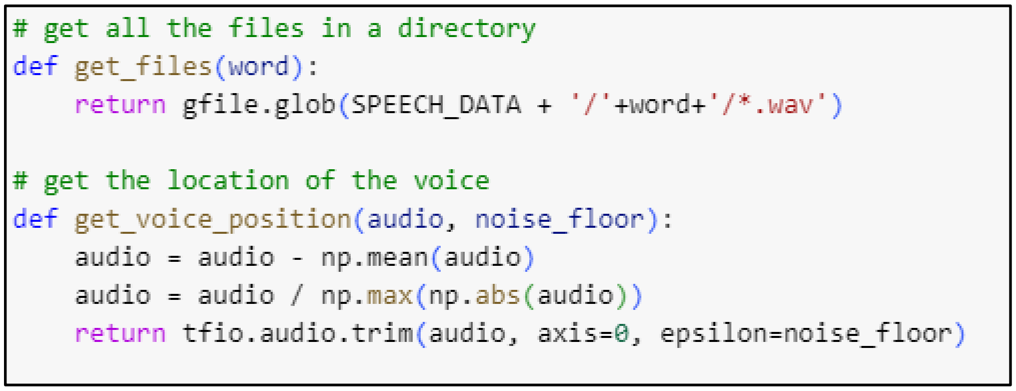

```python
# get all the files in a directory
def get_files(word):
    return gfile.glob(SPEECH_DATA + '/'+word+'/*.wav')

# get the location of the voice
def get_voice_position(audio, noise_floor):
    audio = audio - np.mean(audio)
    audio = audio / np.max(np.abs(audio))
    return tfio.audio.trim(audio, axis=0, epsilon=noise_floor)
```

Fig. 13 Get files and voice position functions

These functions are part of a program for processing and analyzing audio data. The *get_files(word)* helps in gathering relevant audio files, while *get_voice_position(audio, noise_floor)* aids in identifying the position of voice within those audio clips.

The figure below shows how to calculate the duration of the voice in an audio clip by finding the start and end position where the voice is present, checks if the voice exceeds a required length and verifies if the audio clip’s length matches the expected number of samples.

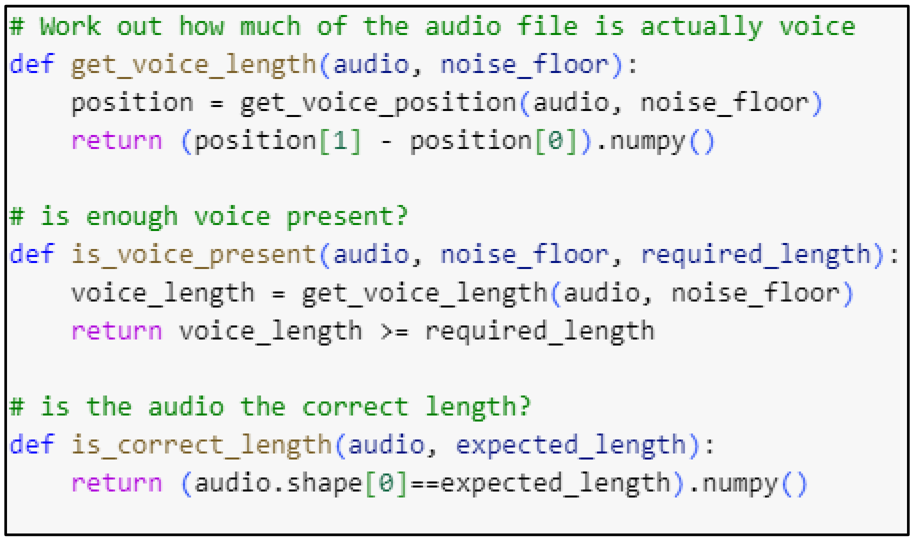
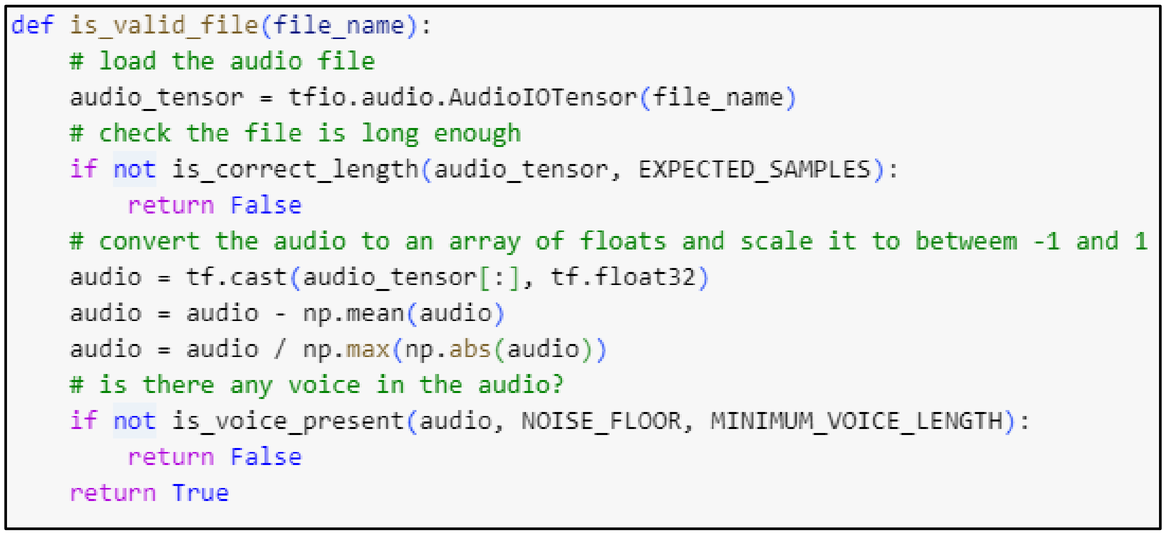

```python
# Work out how much of the audio file is actually voice
def get_voice_length(audio, noise_floor):
    position = get_voice_position(audio, noise_floor)
    return (position[1] - position[0]).numpy()

# is enough voice present?
def is_voice_present(audio, noise_floor, required_length):
    voice_length = get_voice_length(audio, noise_floor)
    return voice_length >= required_length

# is the audio the correct length?
def is_correct_length(audio, expected_length):
    return (audio.shape[0]==expected_length).numpy()
```

Fig. 14 Functions for validating and measuring voice presence

By determining the start and end positions where the voice signal is above a given noise floor, the function *get_voice_length* determines the duration of the voice in an audio clip. This duration is used by the *is_voice_present* function to determine if the voice length in the audio clip either meets or exceeds a required threshold. Confirming the overall length of the audio files, the *is_correct_length* function makes sure the length of the audio clip corresponds to the expected number of samples.

The figure below shows the function on how to check if an audio file is valid by ensuring it has the expected length and contains enough voice content, normalizing the audio data in the process.

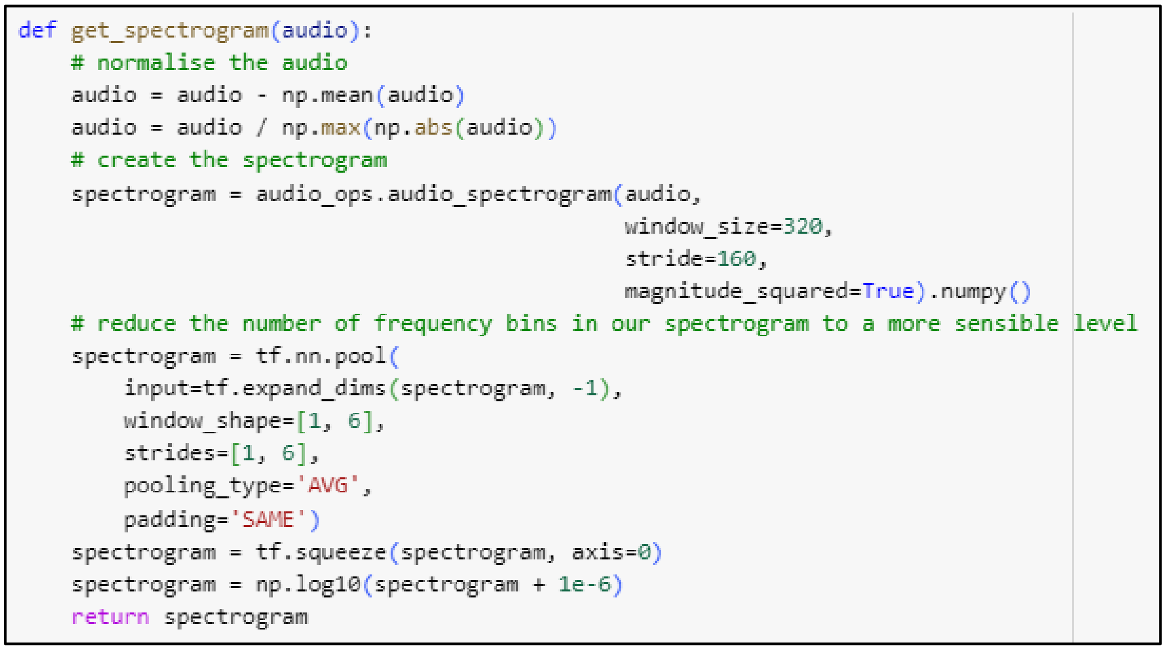


```
def is_valid_file(file_name):
    # load the audio file
    audio_tensor = tfio.audio.AudioIOTensor(file_name)
    # check the file is long enough
    if not is_correct_length(audio_tensor, EXPECTED_SAMPLES):
        return False
    # convert the audio to an array of floats and scale it to betweem -1 and 1
    audio = tf.cast(audio_tensor[:], tf.float32)
    audio = audio - np.mean(audio)
    audio = audio / np.max(np.abs(audio))
    # is there any voice in the audio?
    if not is_voice_present(audio, NOISE_FLOOR, MINIMUM_VOICE_LENGTH):
        return False
    return True
```

Fig. 15 Function to validate audio files

The *is_valid_file* function loads an audio file and first determines whether the number of samples it contains is as expected. Subtracting the mean and scaling the audio data between -1 and 1 normalizes it after that. Finally, it tests if there is enough voice content in the audio above the specified noise floor; if both checks pass, it returns True; if not, it returns False.

The figure below shows the function to transform an audio signal into a spectrogram, which is a visual representation of the frequency spectrum of the audio as it



```
def get_spectrogram(audio):
    # normalise the audio
    audio = audio - np.mean(audio)
    audio = audio / np.max(np.abs(audio))
    # create the spectrogram
    spectrogram = audio_ops.audio_spectrogram(audio,
                                              window_size=320,
                                              stride=160,
                                              magnitude_squared=True).numpy()
    # reduce the number of frequency bins in our spectrogram to a more sensible level
    spectrogram = tf.nn.pool(
        input=tf.expand_dims(spectrogram, -1),
        window_shape=[1, 6],
        strides=[1, 6],
        pooling_type='AVG',
        padding='SAME')
    spectrogram = tf.squeeze(spectrogram, axis=0)
    spectrogram = np.log10(spectrogram + 1e-6)
    return spectrogram
```

varies over time.

Fig. 16 Converting audio signal into spectrogram

The *get_spectrogram* function normalizes an audio input, computes its spectrogram using a Short-Time Fourier Transform (STFT), and, in order to make the spectrogram much simpler to use, averaging pooled the frequency bins. The processed spectrogram is then returned after the dynamic range of the spectrogram data is compressed by a logarithmic transformation for better visualization.

The figure below shows the function that generates a list of tuples with a processed audio file (e.g., converted to spectrogram) and its matching label is shown in the figure below. It repeats each filename a certain number of times.

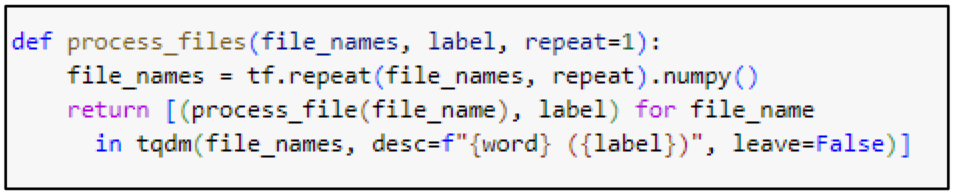
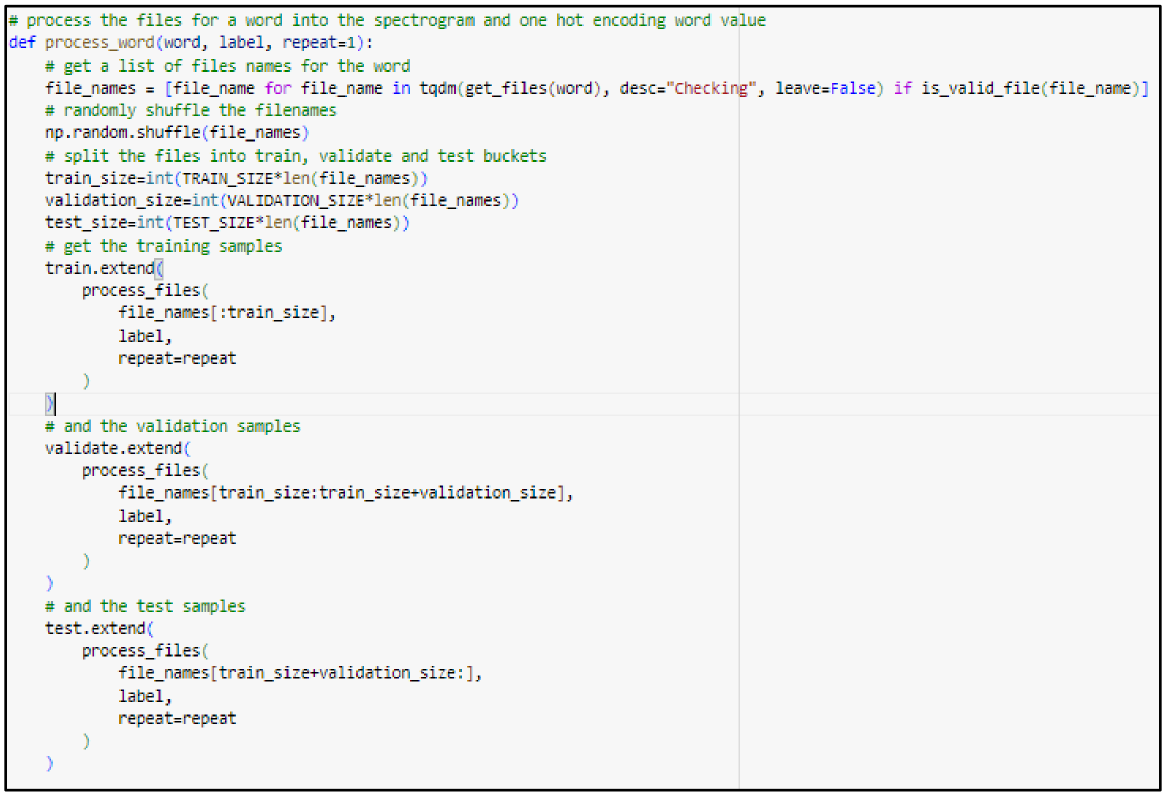

```python
def process_files(file_names, label, repeat=1):
    file_names = tf.repeat(file_names, repeat).numpy()
    return [(process_file(file_name), label) for file_name
      in tqdm(file_names, desc=f"{word} ({label})", leave=False)]
```

Fig. 17 Process audio files function

This program uses TensorFlow's repeat function to repeat each audio file name a certain number of times given a list of them and a label. Each repeated file name is then processed into a spectrogram (or any other processing specified in *process_file* function and matched with the provided label to produce a list of tuples. It also tracks processing status through a progress bar that uses tqdm.

The figure below shows the function on how to organize audio files associated with a specific word into training, validation, and test datasets.

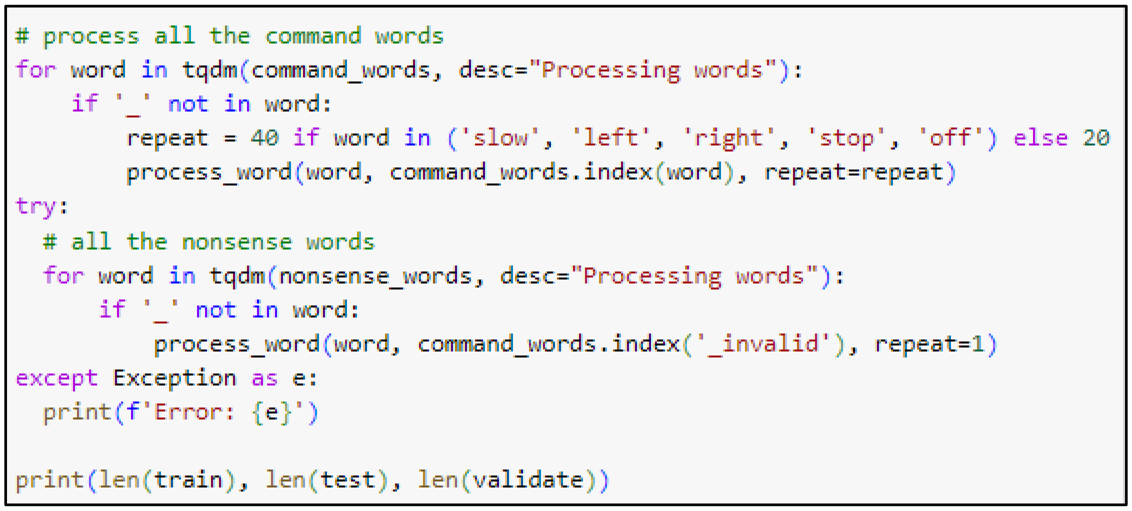


```python
# process the files for a word into the spectrogram and one hot encoding word value
def process_word(word, label, repeat=1):
    # get a list of files names for the word
    file_names = [file_name for file_name in tqdm(get_files(word), desc="Checking", leave=False) if is_valid_file(file_name)]
    # randomly shuffle the filenames
    np.random.shuffle(file_names)
    # split the files into train, validate and test buckets
    train_size=int(TRAIN_SIZE*len(file_names))
    validation_size=int(VALIDATION_SIZE*len(file_names))
    test_size=int(TEST_SIZE*len(file_names))
    # get the training samples
    train.extend(
        process_files(
            file_names[:train_size],
            label,
            repeat=repeat
        )
    )
    # and the validation samples
    validate.extend(
        process_files(
            file_names[train_size:train_size+validation_size],
            label,
            repeat=repeat
        )
    )
    # and the test samples
    test.extend(
        process_files(
            file_names[train_size+validation_size:],
            label,
            repeat=repeat
        )
    )
```

Fig. 18 Function to process word audio files and dataset partitioning

Processing of audio files associated with a specified word is handled by this function, *process_word*. It first gets valid audio file names for the given term, shuffles them at random, and then, using predefined proportions, separates them into training, validation, and test sets. Through the extension of global lists (train, validate, and test) with processed data, the function ensures that the datasets are arranged correctly for machine learning tasks that follow. It ensures that valid files are processed and distributed across the datasets according to predefined sizes, with each file being repeated a specified number of times to augment the dataset.

The figure below shows how to iteratively process audio files associated with command words and nonsense words, organizing them into training, validation, and test datasets.

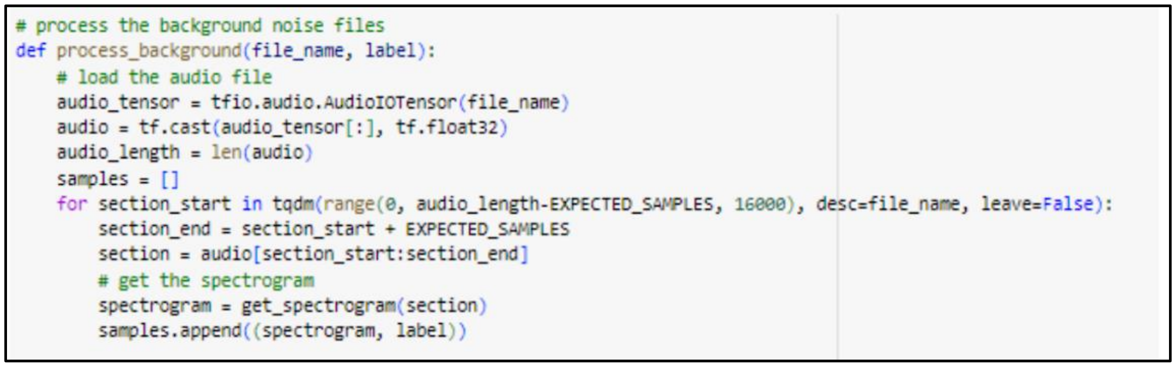


```
# process all the command words
for word in tqdm(command_words, desc="Processing words"):
    if '_' not in word:
        repeat = 40 if word in ('slow', 'left', 'right', 'stop', 'off') else 20
        process_word(word, command_words.index(word), repeat=repeat)
try:
  # all the nonsense words
  for word in tqdm(nonsense_words, desc="Processing words"):
      if '_' not in word:
          process_word(word, command_words.index('_invalid'), repeat=1)
except Exception as e:
  print(f'Error: {e}')

print(len(train), len(test), len(validate))
```

Fig. 19 Generating command and invalid command datasets

This program sorts lists of nonsense and command words into training, validation, and test datasets by spectrogram-processing the corresponding audio files. To prepare for differences in the dataset, it gives some command words varying repetition factors, and it manages possible exceptions when processing the files. The lengths of the training, validation, with and test datasets that result are then printed.

The figure below shows the code on how to process audio files with background noise involves splitting the audio into expected-length segments, creating spectrograms for each segment, and labeling each segment with a background noise indicator.

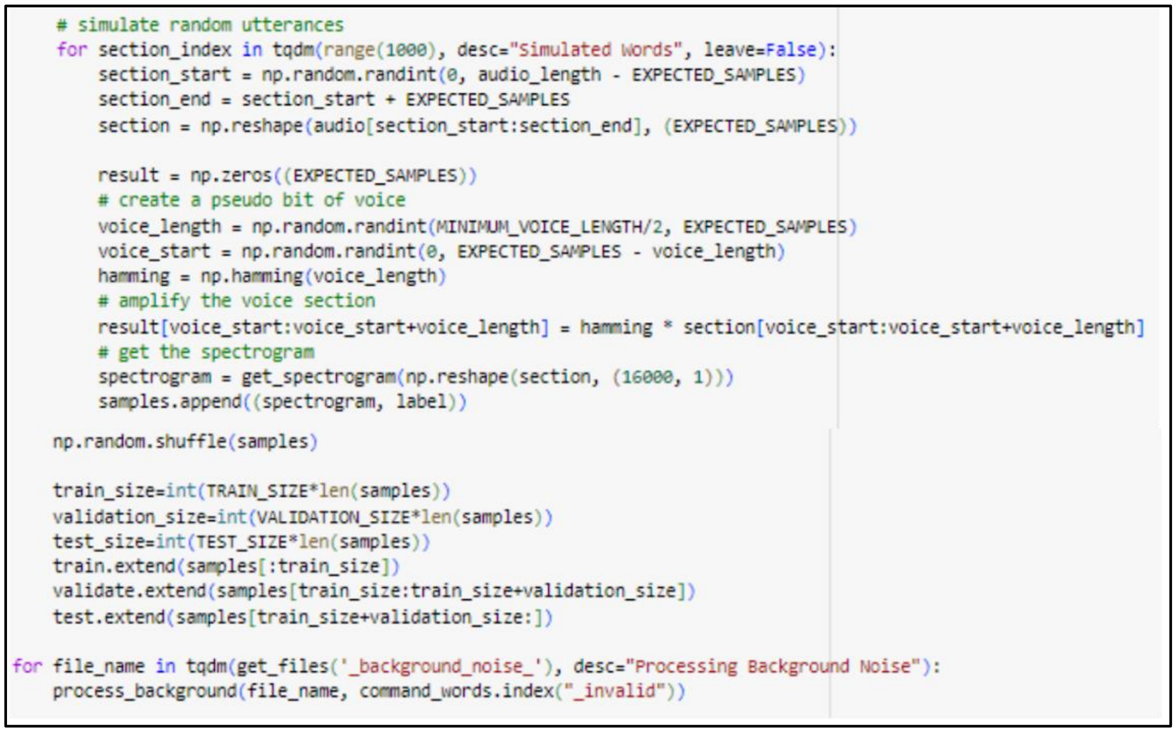


```python
# process the background noise files
def process_background(file_name, label):
    # load the audio file
    audio_tensor = tfio.audio.AudioIOTensor(file_name)
    audio = tf.cast(audio_tensor[:], tf.float32)
    audio_length = len(audio)
    samples = []
    for section_start in tqdm(range(0, audio_length-EXPECTED_SAMPLES, 16000), desc=file_name, leave=False):
        section_end = section_start + EXPECTED_SAMPLES
        section = audio[section_start:section_end]
        # get the spectrogram
        spectrogram = get_spectrogram(section)
        samples.append((spectrogram, label))
```

Fig. 20 Processing background noise files

Spectrogram samples for training a command word recognition model are produced by this function processing background noise audio files. Extracting portions of the audio, it creates spectrograms from simulated speech segments.

The figure below shows the simulated random utterances within the background noise to enhance the dataset and separation of program of the data into training, validation, and test sets.

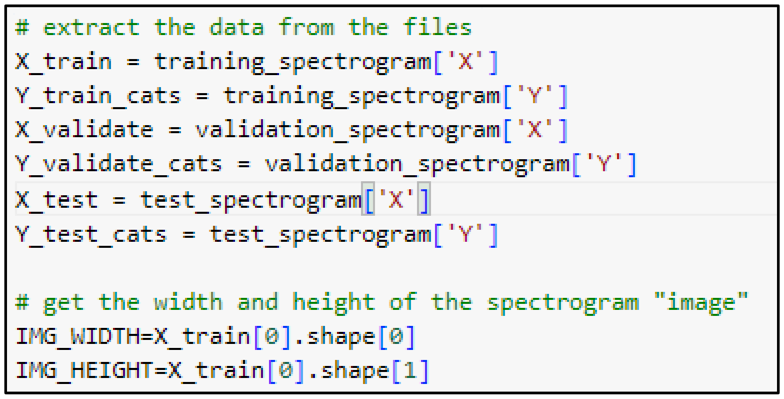


```python
    # simulate random utterances
    for section_index in tqdm(range(1000), desc="Simulated Words", leave=False):
        section_start = np.random.randint(0, audio_length - EXPECTED_SAMPLES)
        section_end = section_start + EXPECTED_SAMPLES
        section = np.reshape(audio[section_start:section_end], (EXPECTED_SAMPLES))

        result = np.zeros((EXPECTED_SAMPLES))
        # create a pseudo bit of voice
        voice_length = np.random.randint(MINIMUM_VOICE_LENGTH/2, EXPECTED_SAMPLES)
        voice_start = np.random.randint(0, EXPECTED_SAMPLES - voice_length)
        hamming = np.hamming(voice_length)
        # amplify the voice section
        result[voice_start:voice_start+voice_length] = hamming * section[voice_start:voice_start+voice_length]
        # get the spectrogram
        spectrogram = get_spectrogram(np.reshape(section, (16000, 1)))
        samples.append((spectrogram, label))

    np.random.shuffle(samples)

    train_size=int(TRAIN_SIZE*len(samples))
    validation_size=int(VALIDATION_SIZE*len(samples))
    test_size=int(TEST_SIZE*len(samples))
    train.extend(samples[:train_size])
    validate.extend(samples[train_size:train_size+validation_size])
    test.extend(samples[train_size+validation_size:])

for file_name in tqdm(get_files('_background_noise_'), desc="Processing Background Noise"):
    process_background(file_name, command_words.index("_invalid"))
```

Fig. 21 Simulate Random Utterances

Extracting portions of the audio, randomly shuffles and separates the segments into training, validation, and test datasets, and also by getting the converted spectogram. This ensures that a range of background noise conditions are trained into the model.

The figure below shows how to extract spectrogram data and their corresponding categorical labels from separate dictionaries for training, validation, and testing datasets, while also determining the width and height of the spectrogram "image".

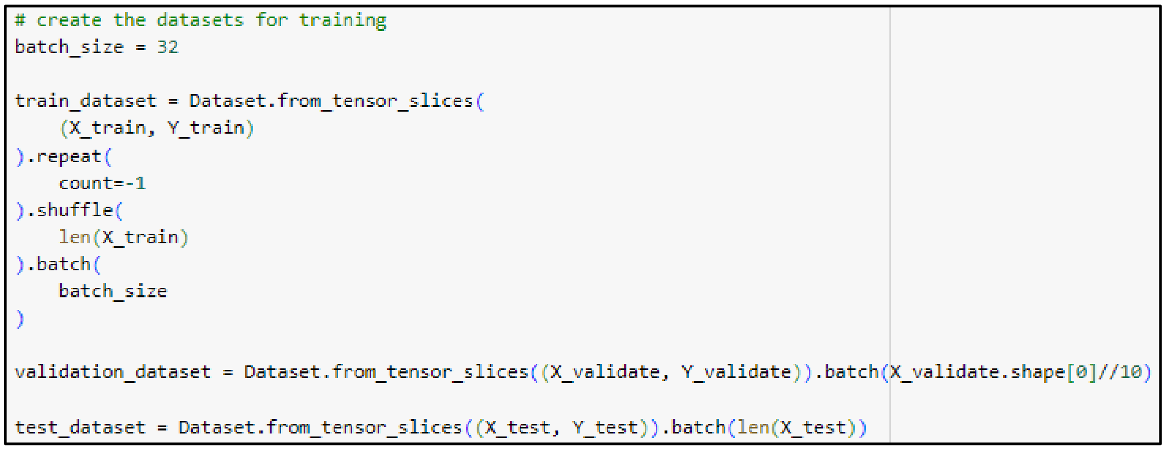


```
# extract the data from the files
X_train = training_spectrogram['X']
Y_train_cats = training_spectrogram['Y']
X_validate = validation_spectrogram['X']
Y_validate_cats = validation_spectrogram['Y']
X_test = test_spectrogram['X']
Y_test_cats = test_spectrogram['Y']

# get the width and height of the spectrogram "image"
IMG_WIDTH=X_train[0].shape[0]
IMG_HEIGHT=X_train[0].shape[1]
```

Fig. 22 Extracting spectrogram dataset

From dictionaries storing training, validation, and testing datasets, this code retrieves categorical labels and spectrogram data. In particular, it splits up the spectrogram data and category labels for every dataset into independent variables and, by looking at the first spectrogram in the training dataset, determines the spectrogram images' width and height.

The figure below shows how to prepare datasets for training, validation, and testing by creating TensorFlow datasets from spectrogram and label data arrays. It configures batch sizes and sets up shuffling for the training dataset to facilitate efficient model training.

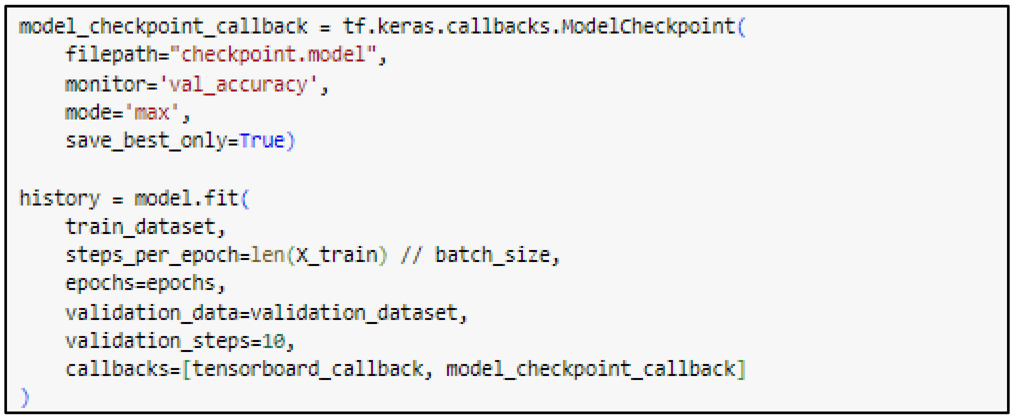


```
# create the datasets for training
batch_size = 32

train_dataset = Dataset.from_tensor_slices(
    (X_train, Y_train)
).repeat(
    count=-1
).shuffle(
    len(X_train)
).batch(
    batch_size
)

validation_dataset = Dataset.from_tensor_slices((X_validate, Y_validate)).batch(X_validate.shape[0]//10)

test_dataset = Dataset.from_tensor_slices((X_test, Y_test)).batch(len(X_test))
```

Fig. 23 Creating training datasets

By slicing the spectrogram and label data arrays and batching them with specified batch sizes, this code segment generates TensorFlow datasets for training, validation and testing. To improve model learning, the training dataset has its data shuffled and repeated endlessly (count=-1). In addition, all of the samples in the testing dataset are batched, whereas the validation dataset is batched to a portion of its size.

The figure below shows how to train the machine learning model using TensorFlow's Keras framework.

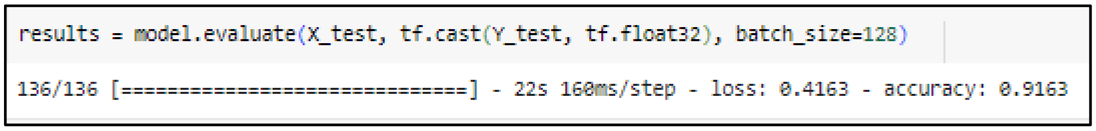


```
model_checkpoint_callback = tf.keras.callbacks.ModelCheckpoint(
    filepath="checkpoint.model",
    monitor='val_accuracy',
    mode='max',
    save_best_only=True)

history = model.fit(
    train_dataset,
    steps_per_epoch=len(X_train) // batch_size,
    epochs=epochs,
    validation_data=validation_dataset,
    validation_steps=10,
    callbacks=[tensorboard_callback, model_checkpoint_callback]
)
```

Fig. 24 Training the model

It set up a ModelCheckpoint callback to save the best-performing model based on validation accuracy. Additionally, it uses TensorBoard to visualize the training process. This setup ensures efficient training and allows us to monitor and save the optimal model for future use.

The figure below shows how to test the created voice command recognition model.

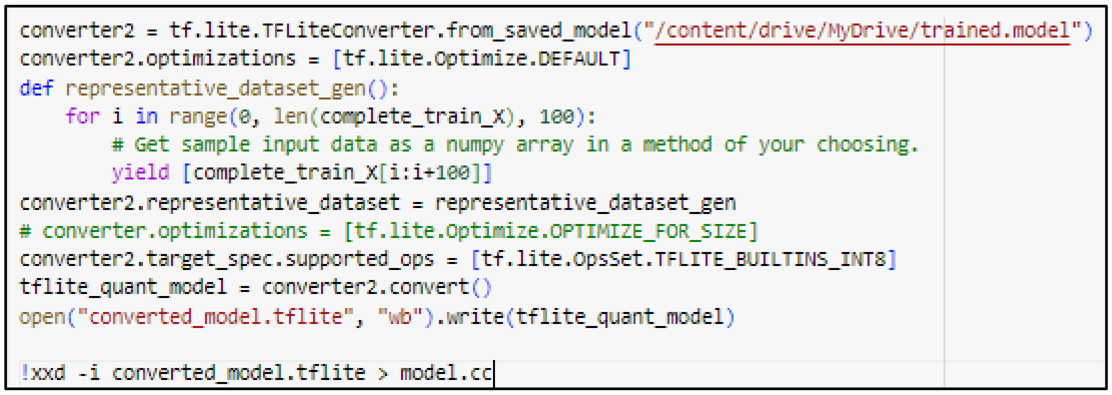


```
results = model.evaluate(X_test, tf.cast(Y_test, tf.float32), batch_size=128)

136/136 [==============================] - 22s 160ms/step - loss: 0.4163 - accuracy: 0.9163
```

Fig. 25 Testing the model

After training the model, the researchers evaluate the model with the accuracy of 91.63%.

The figure below shows the conversion of TensorFlow SavedModel to a TensorFlow Lite model for deployment.

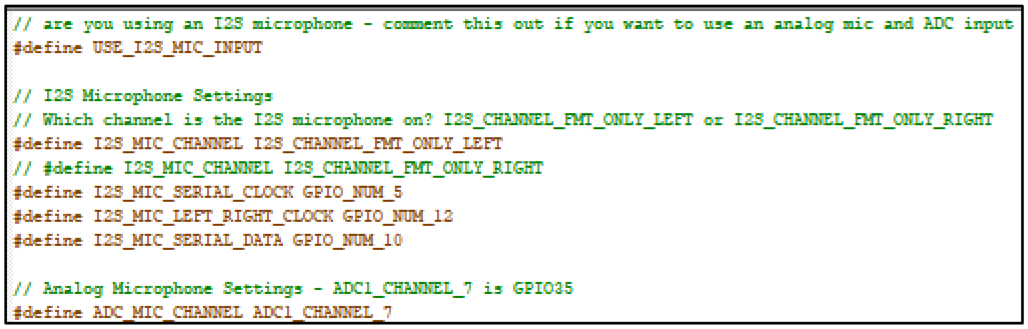


```
converter2 = tf.lite.TFLiteConverter.from_saved_model("/content/drive/MyDrive/trained.model")
converter2.optimizations = [tf.lite.Optimize.DEFAULT]
def representative_dataset_gen():
    for i in range(0, len(complete_train_X), 100):
        # Get sample input data as a numpy array in a method of your choosing.
        yield [complete_train_X[i:i+100]]
converter2.representative_dataset = representative_dataset_gen
# converter.optimizations = [tf.lite.Optimize.OPTIMIZE_FOR_SIZE]
converter2.target_spec.supported_ops = [tf.lite.OpsSet.TFLITE_BUILTINS_INT8]
tflite_quant_model = converter2.convert()
open("converted_model.tflite", "wb").write(tflite_quant_model)

!xxd -i converted_model.tflite > model.cc
```

Fig. 26 Converting the test model into C++ source code

It saves the converted TensorFlow Lite model to a file named "converted_model.tflite". Finally, it converts the file into C++ binary code that will be used to instruct the microcontroller.

The figure below shows the configuration of pins of the INMP441 MEMS microphone.

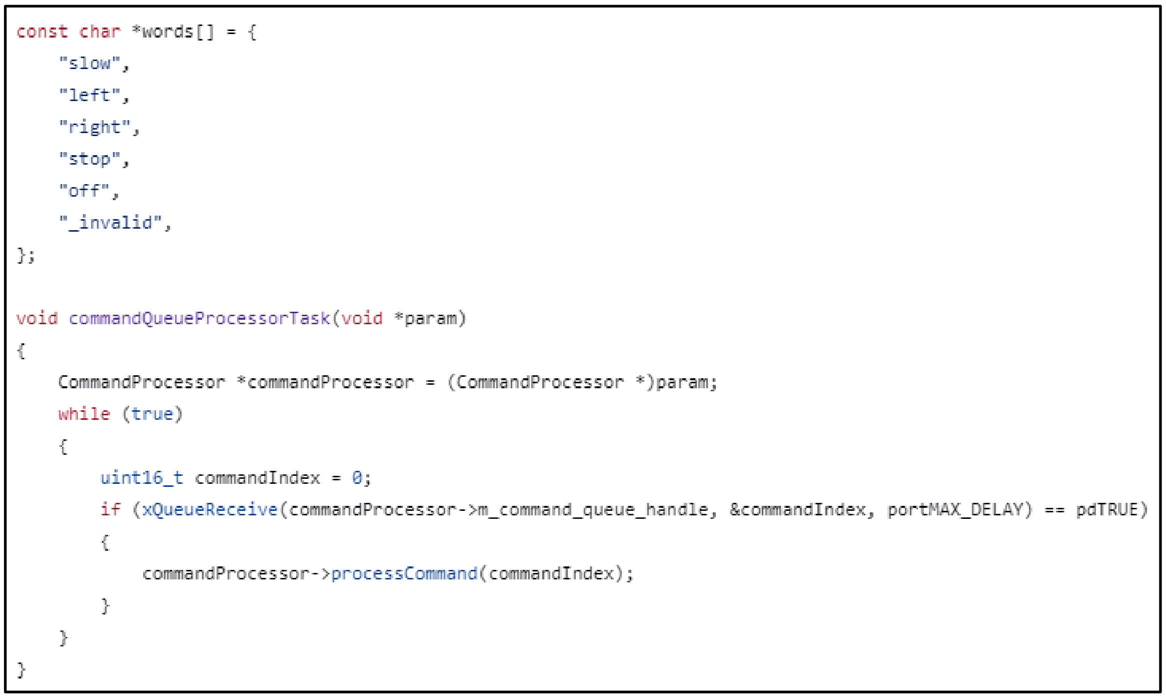


```
// are you using an I2S microphone - comment this out if you want to use an analog mic and ADC input
#define USE_I2S_MIC_INPUT

// I2S Microphone Settings
// Which channel is the I2S microphone on? I2S_CHANNEL_FMT_ONLY_LEFT or I2S_CHANNEL_FMT_ONLY_RIGHT
#define I2S_MIC_CHANNEL I2S_CHANNEL_FMT_ONLY_LEFT
// #define I2S_MIC_CHANNEL I2S_CHANNEL_FMT_ONLY_RIGHT
#define I2S_MIC_SERIAL_CLOCK GPIO_NUM_5
#define I2S_MIC_LEFT_RIGHT_CLOCK GPIO_NUM_12
#define I2S_MIC_SERIAL_DATA GPIO_NUM_10

// Analog Microphone Settings - ADC1_CHANNEL_7 is GPIO35
#define ADC_MIC_CHANNEL ADC1_CHANNEL_7
```

Fig. 27 config.h file

The pins are connected to GPIO_NUM_5 for SCK, GPIO_NUM_12 for WS, and GPIO_NUM_10 for DIN.

The figure below shows declaring specific commands such as ‘slow’, ‘left’, ‘right’, ‘slow’ and ‘off’.

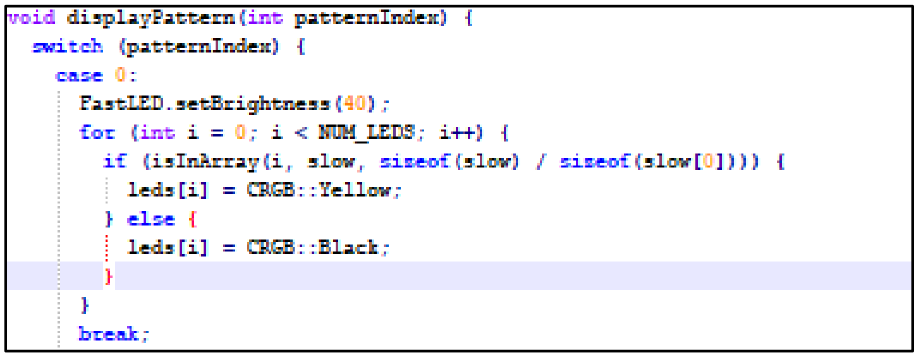


```
const char *words[] = {
    "slow",
    "left",
    "right",
    "stop",
    "off",
    "_invalid",
};

void commandQueueProcessorTask(void *param)
{
    CommandProcessor *commandProcessor = (CommandProcessor *)param;
    while (true)
    {
        uint16_t commandIndex = 0;
        if (xQueueReceive(commandProcessor->m_command_queue_handle, &commandIndex, portMAX_DELAY) == pdTRUE)
        {
            commandProcessor->processCommand(commandIndex);
        }
    }
}
```

Fig. 28 Command words

The code also defines a task function that is responsible for receiving commands from a queue and executes them by invoking the processCommand method in the CommandProcessor object. The program is cyclic and runs endlessly if there are commands to deal with. They are processed one at a time.

The figure below shows the function designed to switched display predefined LED patterns and colors for slow signal light.

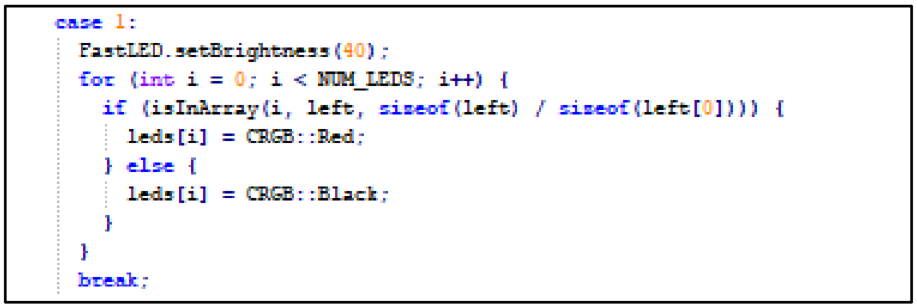


```
void displayPattern(int patternIndex) {
  switch (patternIndex) {
    case 0:
      FastLED.setBrightness(40);
      for (int i = 0; i < NUM_LEDS; i++) {
        if (isInArray(i, slow, sizeof(slow) / sizeof(slow[0]))) {
          leds[i] = CRGB::Yellow;
        } else {
          leds[i] = CRGB::Black;
        }
      }
      break;
```

Fig. 29 Slow signal light display pattern function

It operates based on a variable called patternIndex, which determines the current LED pattern to display. The function switches between different predefined LED patterns and updates the LEDs accordingly for slow signal light.

The figure below shows the function designed to switched display predefined LED patterns and colors for left signal light.

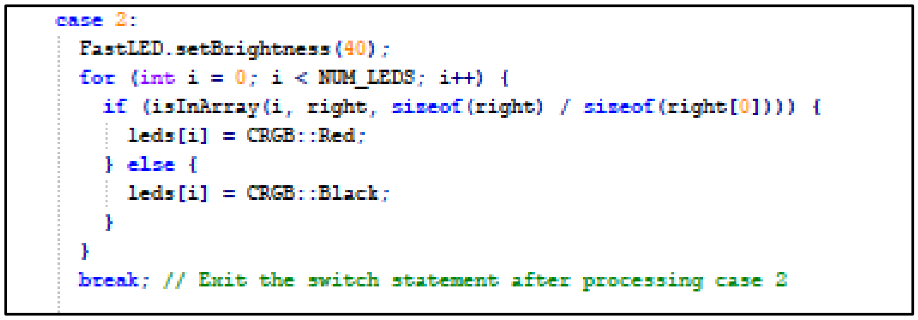


```
    case 1:
      FastLED.setBrightness(40);
      for (int i = 0; i < NUM_LEDS; i++) {
        if (isInArray(i, left, sizeof(left) / sizeof(left[0]))) {
          leds[i] = CRGB::Red;
        } else {
          leds[i] = CRGB::Black;
        }
      }
      break;
```

Fig. 30 Left signal light display pattern function

It operates based on a variable called patternIndex, which determines the current LED pattern to display. The function switches between different predefined LED patterns and updates the LEDs accordingly for left signal light.

The figure below shows the function designed to switched display predefined LED patterns and colors for right signal light.

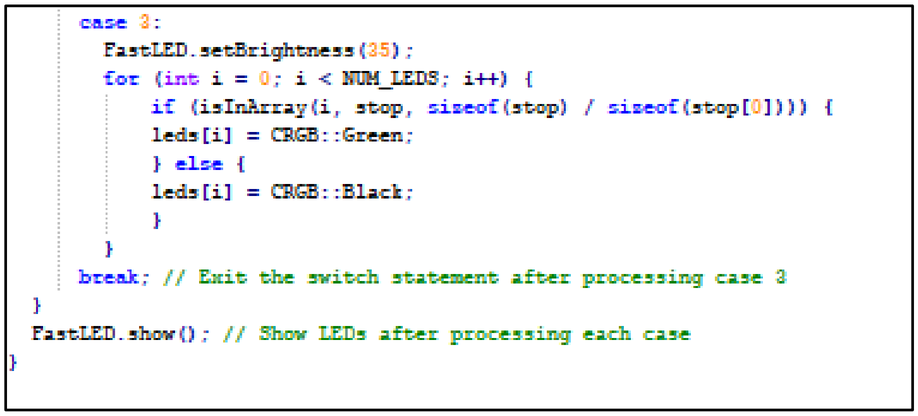


```
case 2:
  FastLED.setBrightness(40);
  for (int i = 0; i < NUM_LEDS; i++) {
    if (isInArray(i, right, sizeof(right) / sizeof(right[0]))) {
      leds[i] = CRGB::Red;
    } else {
      leds[i] = CRGB::Black;
    }
  }
  break; // Exit the switch statement after processing case 2
```

Fig. 31 Right signal light display pattern function

It operates based on a variable called patternIndex, which determines the current LED pattern to display. The function switches between different predefined LED patterns and updates the LEDs accordingly for right signal light.

The figure below shows the function designed to switched display predefined LED patterns and colors for right signal light.

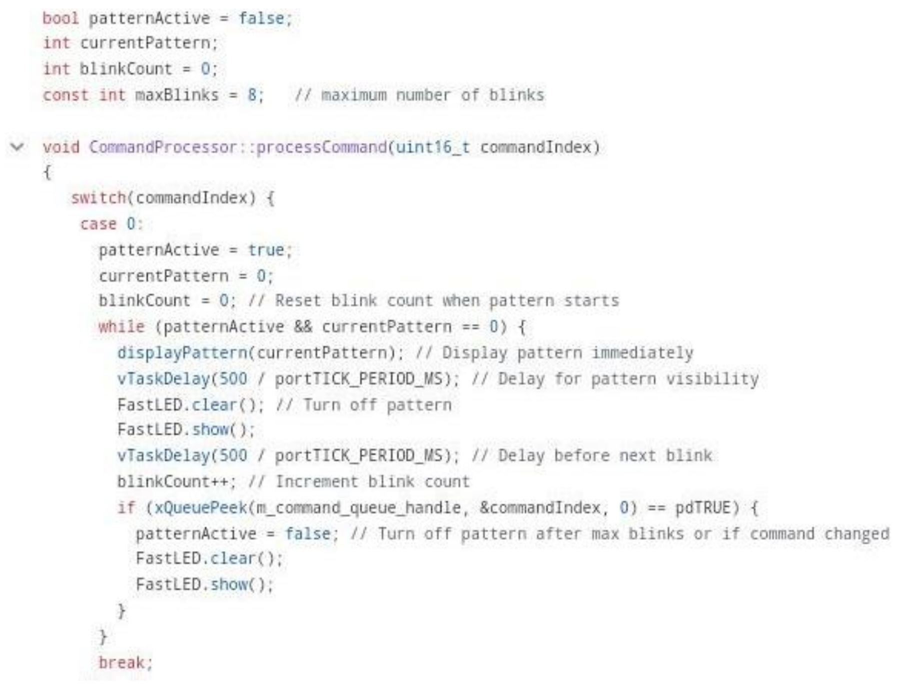


```
    case 3:
      FastLED.setBrightness(35);
      for (int i = 0; i < NUM_LEDS; i++) {
          if (isInArray(i, stop, sizeof(stop) / sizeof(stop[0]))) {
          leds[i] = CRGB::Green;
          } else {
          leds[i] = CRGB::Black;
          }
      }
    break; // Exit the switch statement after processing case 3
  }
  FastLED.show(); // Show LEDs after processing each case
}
```

Fig. 32 Stop signal light display pattern function

It operates based on a variable called patternIndex, which determines the current LED pattern to display. The function switches between different predefined LED patterns and updates the LEDs accordingly for stop signal light.

The figure below shows a function that offers a detailed control system for LED patterns for left signal light display using different command indices.

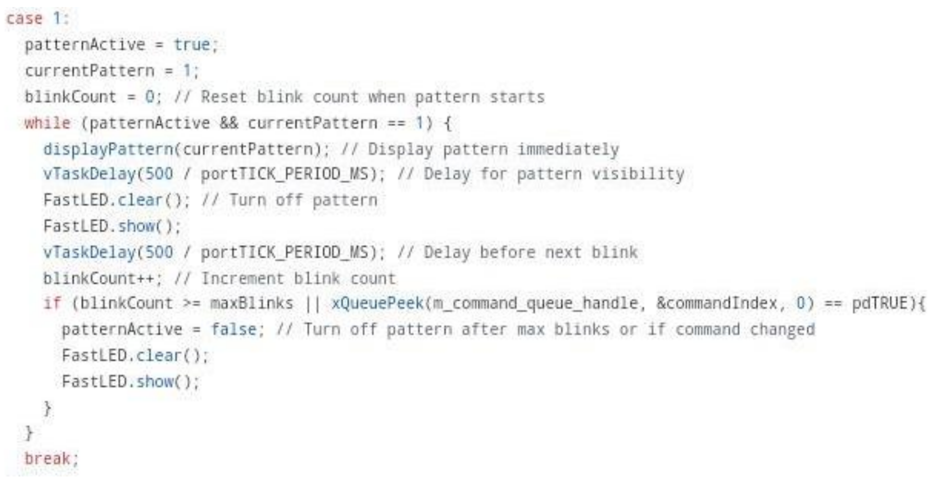


```
bool patternActive = false;
int currentPattern;
int blinkCount = 0;
const int maxBlinks = 8;   // maximum number of blinks

void CommandProcessor::processCommand(uint16_t commandIndex)
{
   switch(commandIndex) {
    case 0:
      patternActive = true;
      currentPattern = 0;
      blinkCount = 0; // Reset blink count when pattern starts
      while (patternActive && currentPattern == 0) {
        displayPattern(currentPattern); // Display pattern immediately
        vTaskDelay(500 / portTICK_PERIOD_MS); // Delay for pattern visibility
        FastLED.clear(); // Turn off pattern
        FastLED.show();
        vTaskDelay(500 / portTICK_PERIOD_MS); // Delay before next blink
        blinkCount++; // Increment blink count
        if (xQueuePeek(m_command_queue_handle, &commandIndex, 0) == pdTRUE) {
          patternActive = false; // Turn off pattern after max blinks or if command changed
          FastLED.clear();
          FastLED.show();
        }
      }
      break;
```

Fig. 33 Function to display pattern of left signal light

This code specifies how to handle different LED pattern display. It controls cases such limited blink counts and timed patterns, activates patterns according to the command index, and delays for left signal light display. When needed, it disables patterns and responds to a command to turn off all LED light.

The figure below shows a function that offers a detailed control system for LED patterns for right signal light display using different command indices.

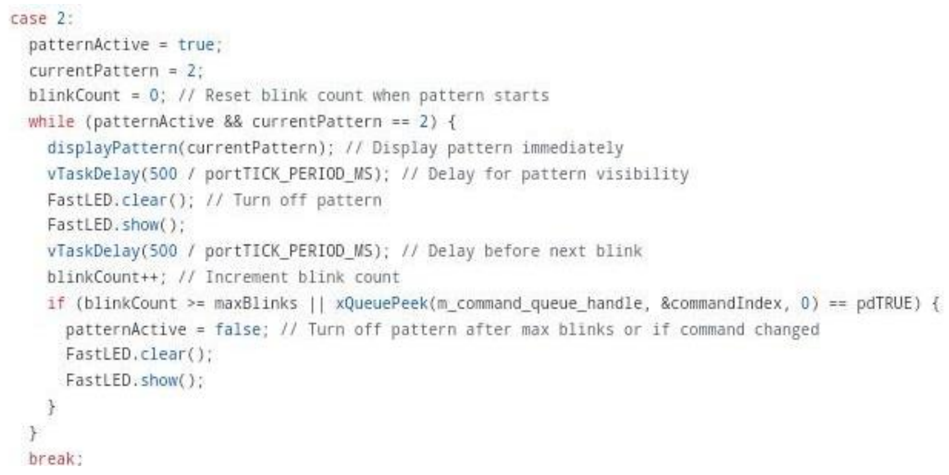


```
case 1:
  patternActive = true;
  currentPattern = 1;
  blinkCount = 0; // Reset blink count when pattern starts
  while (patternActive && currentPattern == 1) {
    displayPattern(currentPattern); // Display pattern immediately
    vTaskDelay(500 / portTICK_PERIOD_MS); // Delay for pattern visibility
    FastLED.clear(); // Turn off pattern
    FastLED.show();
    vTaskDelay(500 / portTICK_PERIOD_MS); // Delay before next blink
    blinkCount++; // Increment blink count
    if (blinkCount >= maxBlinks || xQueuePeek(m_command_queue_handle, &commandIndex, 0) == pdTRUE){
      patternActive = false; // Turn off pattern after max blinks or if command changed
      FastLED.clear();
      FastLED.show();
    }
  }
  break;
```

Fig. 34 Function to display pattern of right signal light

This code specifies how to handle different LED pattern display. It controls cases such limited blink counts and timed patterns, activates patterns according to the command index, and delays for right signal light display. When needed, it disables patterns and responds to a command to turn off all LED light.

The figure below shows a function that offers a detailed control system for LED patterns for stop signal light display using different command indices.

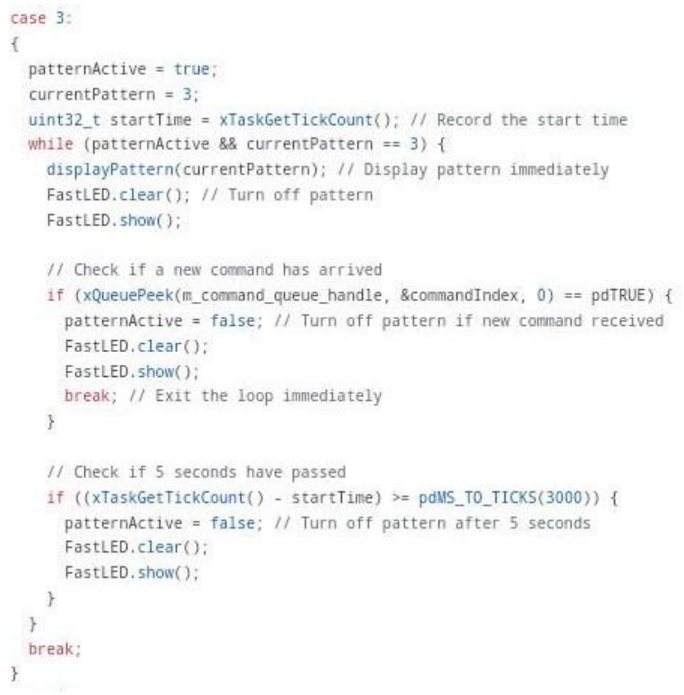


```
case 2:
  patternActive = true;
  currentPattern = 2;
  blinkCount = 0; // Reset blink count when pattern starts
  while (patternActive && currentPattern == 2) {
    displayPattern(currentPattern); // Display pattern immediately
    vTaskDelay(500 / portTICK_PERIOD_MS); // Delay for pattern visibility
    FastLED.clear(); // Turn off pattern
    FastLED.show();
    vTaskDelay(500 / portTICK_PERIOD_MS); // Delay before next blink
    blinkCount++; // Increment blink count
    if (blinkCount >= maxBlinks || xQueuePeek(m_command_queue_handle, &commandIndex, 0) == pdTRUE) {
      patternActive = false; // Turn off pattern after max blinks or if command changed
      FastLED.clear();
      FastLED.show();
    }
  }
  break;
```

Fig. 35 Function to display pattern of stop signal light

This code specifies how to handle different LED pattern display. It controls cases such limited blink counts and timed patterns, activates patterns according to the command index, and delays for stop signal light display. When needed, it disables patterns and responds to a command to turn off all LED light.

The figure below shows a function that offers a detailed control system for LED patterns for slow signal light display using different command indices.

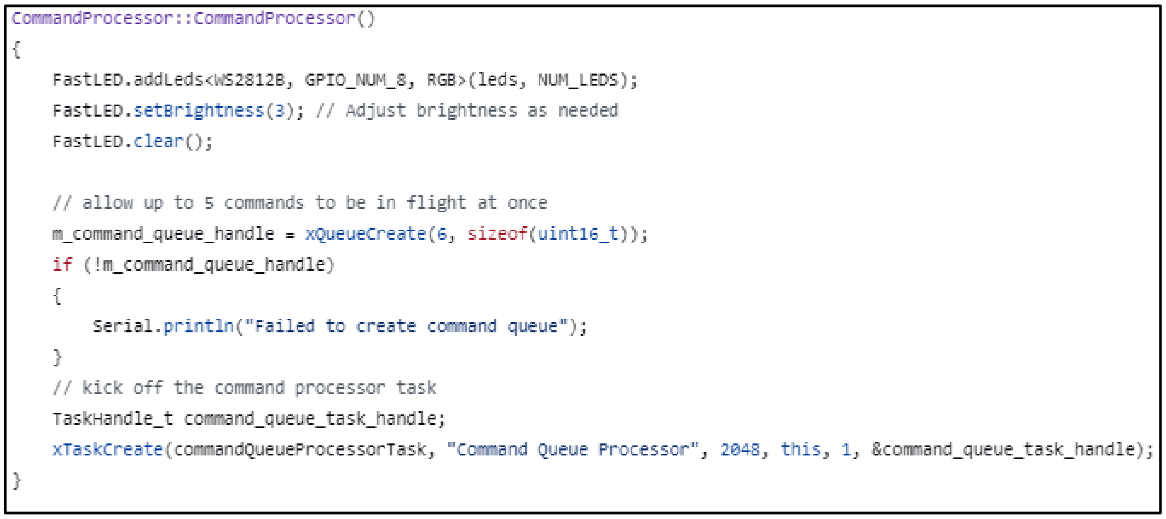


```
case 3:
{
  patternActive = true;
  currentPattern = 3;
  uint32_t startTime = xTaskGetTickCount(); // Record the start time
  while (patternActive && currentPattern == 3) {
    displayPattern(currentPattern); // Display pattern immediately
    FastLED.clear(); // Turn off pattern
    FastLED.show();

    // Check if a new command has arrived
    if (xQueuePeek(m_command_queue_handle, &commandIndex, 0) == pdTRUE) {
      patternActive = false; // Turn off pattern if new command received
      FastLED.clear();
      FastLED.show();
      break; // Exit the loop immediately
    }

    // Check if 5 seconds have passed
    if ((xTaskGetTickCount() - startTime) >= pdMS_TO_TICKS(3000)) {
      patternActive = false; // Turn off pattern after 5 seconds
      FastLED.clear();
      FastLED.show();
    }
  }
  break;
}
```

Fig. 36 Function to display pattern of slow signal light

This code specifies how to handle different LED pattern display. It controls cases such limited blink counts and timed patterns, activates patterns according to the command index, and delays for slow signal light display. When needed, it disables patterns and responds to a command to turn off all LED light.

The figure below shows the setup of a CommandProcessor object, initializing an LED strip and creating a command queue for handling commands.



```
CommandProcessor::CommandProcessor()
{
    FastLED.addLeds<WS2812B, GPIO_NUM_8, RGB>(leds, NUM_LEDS);
    FastLED.setBrightness(3); // Adjust brightness as needed
    FastLED.clear();

    // allow up to 5 commands to be in flight at once
    m_command_queue_handle = xQueueCreate(6, sizeof(uint16_t));
    if (!m_command_queue_handle)
    {
        Serial.println("Failed to create command queue");
    }
    // kick off the command processor task
    TaskHandle_t command_queue_task_handle;
    xTaskCreate(commandQueueProcessorTask, "Command Queue Processor", 2048, this, 1, &command_queue_task_handle);
}
```

Fig. 37 Pin setup for LED Matrix

This code sets up a CommandProcessor object by initializing the LED strip, creating a command queue, and starting a task to process commands asynchronously. If the command queue creation fails, it prints an error message.

Assembly

This part of the study shows the process of the assembly of the connections of the prototype until it became a fully functional system.

This figure below shows the connection between the main prototype and the microphone of it.

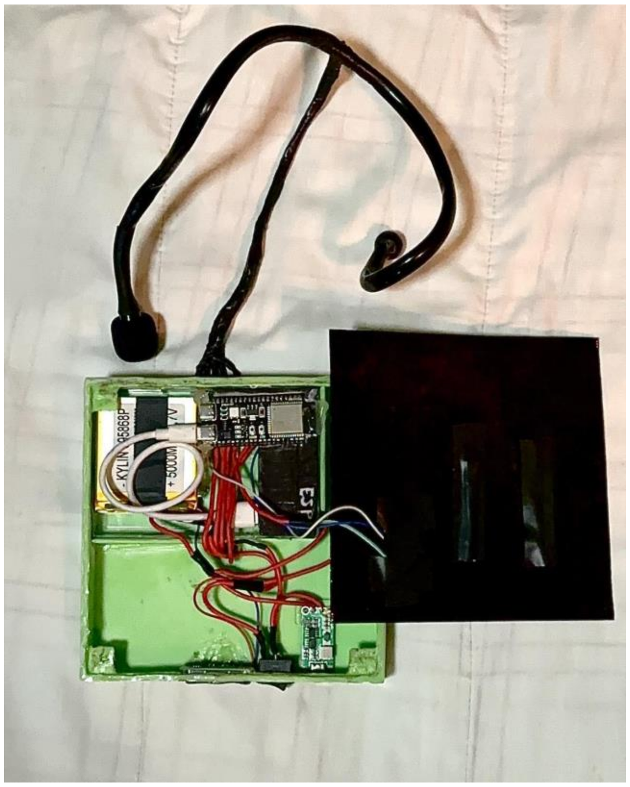


Fig. 38 Connection of the whole system

The components are placed compactly inside the case. The LED matrix is designed in front of the device so that the display design of the device is more visible and applicable for more effective communication along the road. The switch, battery level indicator, and the charging port of the charger module are located at the lower part and compactly close to each other so that the user can easily operate the device as needed. The ESP-32 Microcontroller, USB module, and battery are located in the upper part which is in separate positions as it is the main components of the device. The connection of the wire of the microphone is in the center near the main components so that it is organized and connected to it. The microphone is an input that is responsible for the indication of the input command of the user that triggers the main function of the device. This figure illustrates the connection to the main device of the microphone to detect the command signals whether it is right, left, slow, off, or stop and the display will be in the main device.

The figure below shows the prototype placing on the vest and the proper position in the vest.

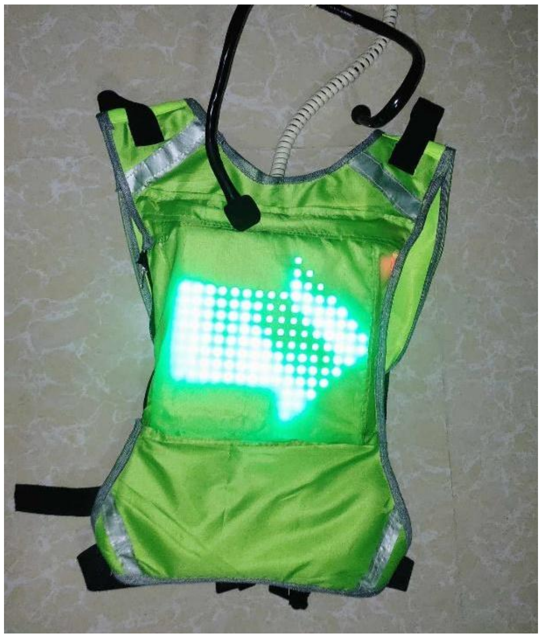


Fig. 39 Placed the device on the cycling vest

The prototype is assembled to the vest, placing it to the appropriate position of it in the pocket of the vest at back, and also the connection of the microphone has separate placed to put it to be able to remove the prototype in the vest completely as to be able to washed the vest.

Project Evaluation

The result of the various tests and evaluations the researcher conducted in order to achieve the study's objectives is shown below.

Table 6 shows the testing of the device in terms of time response, expected and actual response of detected command. Each command undergoes testing, during which its time response is recorded alongside the expected and actual commands detected. Also the testing of background noise and invalid commands test its response.

Table 6. Result of Response and Response Time of the Signal Light Displayed based on the command.

Command	**Trial**	**Expected Response (Command Detected)**	**Actual Response (Command Detected)**	**Response Time (Signal Light Displayed)**
Left	1	Left	Left	0.38 second
	2	Left	Left	0.39 second
	3	Left	Left	0.39 second
	4	Left	Left	0.38 second
	5	Left	Left	0.38 second
	6	Left	Left	0.37 second
	7	Left	Left	0.39 second
	8	Left	Left	0.38 second
	9	Left	Left	0.36 second
	10	Left	Left	0.37 second
Right	1	Right	Right	0.37 second
	2	Right	Right	0.37 second

	3	Right	Right	0.38 second
	4	Right	Right	0.35 second
	5	Right	Right	0.38 second
	6	Right	Right	0.36 second
	7	Right	Right	0.35 second
	8	Right	Right	0.39 second
	9	Right	Right	0.37 second
	10	Right	Right	0.38 second
Slow	1	Slow	Slow	0.38 second
	2	Slow	Slow	0.37 second
	3	Slow	Slow	0.39 second
	4	Slow	Slow	0.38 second
	5	Slow	Slow	0.36 second
	6	Slow	Slow	0.36 second
	7	Slow	Slow	0.37 second
	8	Slow	Slow	0.35 second
	9	Slow	Slow	0.38 second
	10	Slow	Slow	0.39 second
Stop	1	Stop	Stop	0.38 second
	2	Stop	Stop	0.37 second
	3	Stop	Stop	0.37 second
	4	Stop	Stop	0.39 second
	5	Stop	Stop	0.38 second
	6	Stop	Stop	0.38 second
	7	Stop	Stop	0.36 second
	8	Stop	Stop	0.40 second

	9	Stop	Stop	0.39 second
	10	Stop	Stop	0.37 second
Off	1	Off	Off	0.37 second
	2	Off	Off	0.36 second
	3	Off	Off	0.38 second
	4	Off	Off	0.36 second
	5	Off	Off	0.36 second
	6	Off	Off	0.38 second
	7	Off	Off	0.37 second
	8	Off	Off	0.37 second
	9	Off	Off	0.36 second
	10	Off	Off	0.38 second
Noise/Invalid	1	Invalid	Invalid	Invalid
	2	Invalid	Invalid	Invalid
	3	Invalid	Invalid	Invalid
	4	Invalid	Invalid	Invalid
	5	Invalid	Invalid	Invalid
	6	Invalid	Invalid	Invalid
	7	Invalid	Invalid	Invalid
	8	Invalid	Invalid	Invalid
	9	Invalid	Invalid	Invalid
	10	Invalid	Invalid	Invalid

Based on the result above, each command has 10 trials and shows the actual command detected and the time when the signal light is displayed. According to the result, the prototype is efficiently responding to the voice command of the user with 10 out of 10 trials each command. The average response time for left and stop command is 0.38 second, for right, slow and off command is 0.37 second. For the noise and the invalid command, the trials used traffic/road noise and the device was proven to be efficient even with background noise.

The testing of the response of input command and response time of the prototype based on the command voice level is shown in table 7, the test is to determine on what voice level the device is more efficient in response to the command.

Table 7. Result of Response of the Signal Light Displayed based on Voice Level along with Background Noise

Voice Level	Trial	Input Command	Signal Light Display
Low Voice	1	Left	The Left Signal Light Displayed
	2	Right	Right Signal Light Displayed
	3	Slow	No Responded Signal Light Display
	4	Stop	Stop Signal Light Displayed
	5	Off	No Response in Signal Light Displayed Off
	6	Left	No Responded Signal Light Display
	7	Right	Right Signal Light Displayed
	8	Slow	Slow Signal Light Displayed
	9	Stop	Stop Signal Light Displayed

	10	Off	Current Signal Light Displayed Off
Normal Voice	1	Left	Left Signal Light Displayed
	2	Right	Right Signal Light Displayed
	3	Slow	Slow Signal Light Displayed
	4	Stop	Stop Signal Light Displayed
	5	Off	Signal Light Displayed Off
	6	Left	Left Signal Light Displayed
	7	Right	No Responded Signal Light Display
	8	Slow	No Responded Signal Light Display
	9	Stop	Stop Signal Light Displayed
	10	Off	Signal Light Displayed Off
High Voice	1	Left	Left Signal Light Displayed
	2	Right	Right Signal Light Displayed
	3	Slow	Slow Signal Light Displayed
	4	Stop	Stop Signal Light Displayed
	5	Off	Signal Light Displayed Off
	6	Left	Left Signal Light Displayed
	7	Right	Right Signal Light Displayed
	8	Slow	Slow Signal Light Displayed
	9	Stop	Stop Signal Light Displayed
	10	Off	Signal Light Displayed Off

The result of this test indicates that background noise has been evaluated along with the voice level for every command. The device responds to the input command in 7 out of 10 trials for low voice, 8 out of 10 trials for normal voice, and 10 out of 10 trials for high voice.

Table 8 shows the results of the tests on the efficiency of the response based on the distance measured in inches through numerous trials and variations of distance.

Table 8. Result of Response of the Signal Light Displayed based on the Distance in centimeter.

Distance	**Trial**	**Input Command**	**Response (Signal Light Display)**
2 inches	1	Left	Left Signal Light Displayed
	2	Right	Right Signal Light Displayed
	3	Slow	Slow Signal Light Displayed
	4	Stop	Stop Signal Light Displayed
	5	Off	Signal Light Displayed Off
	6	Left	Left Signal Light Displayed
	7	Right	Right Signal Light Displayed
	8	Slow	Slow Signal Light Displayed
	9	Stop	Stop Signal Light Displayed
	10	Off	Signal Light Displayed Off
10 inches	1	Left	Left Signal Light Displayed
	2	Right	Right Signal Light Displayed

	3	Slow	No Responded Signal Light Displayed
	4	Stop	Stop Signal Light Displayed
	5	Off	Signal Light Displayed Off
	6	Left	Left Signal Light Displayed
	7	Right	Right Signal Light Displayed
	8	Slow	Right Signal Light Displayed
	9	Stop	Stop Signal Light Displayed
	10	Off	Signal Light Displayed Off
> 30 inches	1	Left	Left Signal Light Displayed
	2	Right	No Responded Signal Light Displayed
	3	Slow	No Responded Signal Light Displayed
	4	Stop	No Responded Signal Light Displayed
	5	Off	Signal Light Displayed Off
	6	Left	No Responded Signal Light Displayed
	7	Right	Right Signal Light Displayed
	8	Slow	No Responded Signal Light Displayed
	9	Stop	Stop Signal Light Displayed
	10	Off	Signal Light Displayed Off

The results of the test, the response of the signal light displayed shows based on the distance where the voice command came from between the microphone. According to the test data above, the prototype is more efficient in detecting and receiving the voice command at a distance of 2 inches away from the microphone in 10 out of 10 trials, and in the 10 inches away in trial 3 the slow command didn’t detect the microphone. In the distance above 30 inches away, the voice command didn’t properly receive the commands.

Table 9 shows the results of the tests on the efficiency of the response based on the sound similarities of the command. The test run through various words that sound like the command.

Table 9. Result of Response of the Signal Light Displayed based on sound similarities of the command words.

Command	Similar Sound Words	Response (Signal Light Displayed)
Left	Lift	Left Signal Light Displayed
	Laughed	Left Signal Light Displayed
	Leafed	Left Signal Light Displayed
	Lefty	Left Signal Light Displayed
	Loft	No Signal Light Displayed
	Luft	Left Signal Light Displayed
	Cleft	Left Signal Light Displayed
	Leff	Left Signal Light Displayed
	Let	Left Signal Light Displayed
	Lafitte	Left Signal Light Displayed

Right	Rat	Right Signal Light Displayed
	Rate	Right Signal Light Displayed
	Rait	Right Signal Light Displayed
	Rit	No Signal Light Displayed
	Ret	Left Signal Light Displayed
	Ride	Right Signal Light Displayed
	Riot	Right Signal Light Displayed
	Write	Right Signal Light Displayed
	Rite	Right Signal Light Displayed
	Aright	Right Signal Light Displayed
Slow	Sloe	Stop Signal Light Displayed
	Sallow	Stop Signal Light Displayed
	Silo	No Signal Light Displayed
	Slay	No Signal Light Displayed
	Sloth	Stop Signal Light Displayed
	Slew	No Signal Light Displayed
	Slue	No Signal Light Displayed
	Slaw	Stop Signal Light Displayed
	Sloat	Stop Signal Light Displayed
	Sleigh	No Signal Light Displayed
Stop	Steep	No Signal Light Displayed
	Step	Stop Signal Light Displayed
	Steppe	Stop Signal Light Displayed
	Stipe	No Signal Light Displayed
	Stomp	Stop Signal Light Displayed
	Strop	Stop Signal Light Displayed

	Setup	No Signal Light Displayed
	Stab	No Signal Light Displayed
	stay up	No Signal Light Displayed
	Sop	Stop Signal Light Displayed
Off	Oft	Off the Signal Light Displayed
	Orf	No Responded
	Offer	No Responded
	Iff	No Responded
	Oaf	Off the Signal Light Displayed
	Ooph	Off the Signal Light Displayed
	Auth	No Responded
	Awe	Off the Signal Light Displayed
	Awful	Off the Signal Light Displayed
	Cough	No Responded

Based on the result above, each command has 10 trials of similar sound words of the exact voice command to test the response of the signal light displayed on the prototype. After testing all trials, the "left" command tests the efficiency in the sound of "t" at last pronounced of the command word, and the "right" command is efficient in the sound of " Ra " and "Rey" or in the sound of "R" in first pronounced of the command word, in the "slow" and "stop" command its efficiency close to each other that is in the sound of "s" and "o" at the pronunciation of the word, and for the "off" command it is efficient in the sound of "o" and "f" at the beginning of saying the command.

Limitations and capabilities

The researchers find out the limitations and capabilities of the device after the evaluation through numerous tests.

The detection of the voice command is based on the level of voice, indication of words and distance from the microphone. The higher the background noise the more you need to higher the level of your voice and near distance to the microphone. The device is only efficient within the 2-inch distance from the microphone, that’s why when the user is not within the distance the device does not respond as it should. The device is unable to differentiate between the exact command words and similar sound words.

The device was capable of detecting voice commands to control the signal light. The device shows a display for each specified command in the signal light system. Each of the signal light displays can be displayed in its specified mode. The device can handle the display of another command if the user wants to change the currently displayed command in a moment of time. The display of every signal light will be terminated using the off command.

Chapter V

SUMMARY, CONCLUSIONS AND RECOMMENDATION

This section of this study provides a brief summary of the project and the overall development of the device. Recommendations are generated based on the summary and conclusions.

Summary

The research emphasizes the creation of a voice-controlled signal light vest, which controls the signal light if the system receives a voice command (left) the LED matrix displays a green blinking left arrow signal light if the system receives a voice command (right) the display will be a green blinking right arrow signal light, if the system received a voice command (slow) the display would be a full bright yellow signal light, if the system received a voice command (stop) the display would be a full bright red signal light and the system has an off- feature to turn off the current signal displayed.

The visibility on the road is crucial for the cyclist in order to enhance safety and to communicate with other road users, the cyclist needs to use hand signals but it is more likely to cause an accident due to failing to properly signal intentions while riding on the road and leaving their hand on the handlebar. As a result, the researchers decided to create an innovative way to solve the problem. The primary goal of the research is to develop a Voice-Controlled Signal Light Vest for Cyclists.

The study used an applied research method in that it involves developing a device to test or evaluate proposed solutions. The Voice-Controlled Signal Light Vest was eventually developed by the researchers. The proposed study aims to create a device that can enhance safety and communication on the road.

The researchers used machine learning to train the data and create a model to recognize the user's command. The microcontroller used by the researchers was ESP32-S3-DevKitC-1 and it is in C++ programming language. The microcontroller was programmed to get the input from the MEMS microphone and control the signal light display on the LED matrix based on the user’s voice command.

Findings

The researchers carefully analyzed and selected the appropriate materials and components for this study. The color, size of the display, and additional reflective strips are better for the visibility of the vest. In selecting the materials used in the device and the vest it should be lightweight to be able to be used for a long period of time, the water-resistant fabric used to hold the device so the device cannot be damaged by water, and also the vest has the adjustable strap for more flexible use in fitting. The capacity of the battery of the device should be held for longer use of the device and through the operation of the display of the device it should have modes of display as the functions, so it is better to apply the mode of display of the signal light in a mode of blinking signal light and should have the enough applied brightness on it. For ergonomics, the vest should be comfortable enough by applying extra padding and at the same time it should be better for flexible wearing as it can be adjustable.

The circuit was constructed by the researchers, who also connected those important components. In order to respond to the user's voice command, the system's connection was properly pinned to process the incoming audio signals, extract specific command data, and activate the vest's signal lights to light up properly.

When the device was being programmed, the researchers used machine learning to develop a model for voice command recognition. The audio and its spectrogram served as the training data. The model predicted the spectrogram of the audio input, sent the audio signal to the microcontroller's I2S interface, checked the voice command ID, and finally displayed the corresponding signal light pattern.

Regarding the voice level, the device demonstrates efficiency in responding effectively to both normal and high voice along with background noise. Furthermore, it efficiently detects commands within a range of 2 inches from the microphone. However, the device responds to words that sound similar to the command word as well as to that exact command word. This indicates that the voice recognition system on the device is not finely tuned to distinguish between similar sound words and specific commands.

Conclusion

The voice - controlled signal lights vest is successfully developed to incorporate high-visibility LEDs and voice-activated technology, creating an reliable signaling method for cyclists. The materials and design of the vest can ensure durability, visibility, and comfort for the user. Through various trials and testing, the system can provide vision to other road users in terms of signal lights. The system is able to generate a signal light display based on the user’s voice command without using the hand signals. The system also works as a safety device for cyclists avoiding miscommunication to other road users.

Recommendations

The researchers thought that this study may be improved even more. The researchers further recommend that more advanced features be included for additional developments. These consist as follows:

1. Future researchers could use a wireless connection for the microphone of the device wherein it can be in Bluetooth or Wi-Fi connections.
2. Future researchers could use other algorithms for voice recognition system models that implement more advanced methods that distinguish the difference between the specific command.
3. Future researchers could add features to the device that can communicate with another cyclist.
4. Future researchers could have an indicator that the cyclists see or know whenever it commands.

References Cited

Websites

ESP32-S3-DevKitC-1 v1.1 - ESP32-S3 - — ESP-IDF Programming Guide v5.2.2 documentation. (n.d.). https://docs.espressif.com/projects/esp-idf/en/stable/esp32s3/hw-reference/esp32s3/user-guide-devkitc-1.html

What is a Spectrogram? | Pacific Northwest Seismic Network. (n.d.). Pacific Northwest Seismic Network. https://www.pnsn.org/spectrograms/what-is-a-spectrogram

Uniforms, A. (2024, February 6). A Complete Guide to High-Visibility Colors | Alsco. https://alsco.com/resources/a-complete-guide-to-high-visibility-colors/

11 Hand Signals Every Cyclist Must Know on the Road. (2024, March 27). Www.bikelegalfirm.com. https://www.bikelegalfirm.com/bike-hand-signals#:~:text=In%20this%20article%2C%20we%27ll%20cover%3A%201%20The%20top

The case for using a backpack for cycling commuting. (2017, April 14). Craft Cadence. https://craftcadence.com/blogs/backpacks/waterproof-backpack-for-cycling-guide

Admin. (2023, April 24). Standard Backpack Dimensions - Complete Guide - MeasuringHow. MeasuringHow. https://www.measuringhow.com/standard-backpack-dimensions-guide/?fbclid=IwZXh0bgNhZW0CMTAAAR32biBgjDbi9jYb0hAZ6jgccTvcF4JmD hs5vWoxtrmQZihybITGFeBnIBo_aem_AZbSfpdHJPfJuctfSLVDdUbFPLiVmmRXrHd4-Na2gBpasImkeNDbppqiTqcxq4DSWxAnlrRoyHXhvpJ0hZlWi77M

Helpful Driving Info | Traffic Signals. (n.d.). DriversEd.com. https://driversed.com/driving-information/signs-signals-and-markings/traffic-signals/?fbclid=IwAR0XuzxsYH2RQETofD6Ap6PIihETzi8CpQ3-vMTfDdxK5v1V1DVfzaxmOHc

Staff, C. (2024, March 27). What Is Machine Learning? Definition, Types, and Examples. Coursera. https://www.coursera.org/articles/what-is-machine-learning

Journals

Creswell, J. W. (2018). Research design: Qualitative, quantitative, and mixed methods approaches (5th ed.). SAGE Publications.

VoiceTurn is a voice-controlled turn signal system for safer bike rides. (2021, July 19). Arduino Blog.https://blog.arduino.cc/2021/07/19/voiceturn-is-a-voice-controlled-turn-signal-system-for-safer-bike-rides/

Dulo, J. F. (n.d.). Signal Lights for Cyclist Through Voice Turn and Manual Switch. Www.academia.edu. Retrieved May 16, 2024, from https://www.academia.edu/82281820/Signal_Lights_for_Cyclist_Through_Voice_Turn_and_Manual_Switch

Savino, G.-L., Jessé Moraes Braga, & Johannes Schöning. (2021). VeloCity: Using Voice Assistants for Cyclists to Provide Traffic Reports. Alexandria (UniSG) (University of St.Gallen). https://doi.org/10.1145/3474085.3475509

Hinson, N. (2019). LifeLight: Wearable Active Hazard Detection System for Urban/Suburban Nighttime Cyclists. Handle.net. http://hdl.handle.net/11714/6594

Wafri, A., & Bin, Z. (n.d.). LED BIKE SAFETY VEST (HIKARI VEST) NAME REGISTRATION NO. http://repository.psa.edu.my/bitstream/123456789/4311/1/LED%20BIKE%20SAFETY%20VEST%20%28HIKARI%20VEST%29%2008DEP20F1052%20AHMAD%20WAFRI%20ZUHDI%20BIN%20FAHMIDDIN%20FINAL%20REPORT%20%281%29.pdf

The Smart Jacket for Cyclist. (n.d.). Scholar.google.com. Retrieved May 16, 2024, from https://scholar.google.com/citations?view_op=view_citation&hl=en&user=orz83Z4AAAAJ&citation_for_view=orz83Z4AAAAJ:mVmsd5A6BfQC

Nordmark, A. (2019). Designing Multimodal Warning Signals for Cyclists of the Future. In ltu.diva-portal.org. https://urn.kb.se/resolve?urn=urn:nbn:se:ltu:diva-74884

Rangan, R. P., Sangameshwaran, M., Poovendan, V., Pavanpranesh, G., & Naveen, C. (2018). Voice Controlled Smart Helmet. Asian Review of Mechanical Engineering, 7(2), 1. https://www.academia.edu/67762042/Voice_Controlled_Smart_Helmet

Maroma, A. (2018, July 11). Development of Motorcycle Jacket with Modified Indicator and Brake Lights. Papers.ssrn.com. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3211943

Alsalman, M., Alkhudher, N., & Shabakouh, H. (2021). Smart LED Bike Jacket. Dspace.auk.edu.kw. https://dspace.auk.edu.kw/items/da79d94b-4a61-46ba-a9cf-35fa810fe1b4

AASHTO. Highway Safety Manual. AASHTO, Washington, DC, 2010.

Mittal, A., & Jain, R. (2022). Voice-Based Direction Indicator for the Cycle. IJournals:International Journal of Software & Hardware Research in Engineering ISSN:2347-4890, 10(4). https://ijournals.in/journal/index.php/ijshre/article/view/112

Journal, I., & H, H. (2020). iSmart Cyclist Jacket. IJCSMC. https://www.academia.edu/43933186/iSmart_Cyclist_Jacket_