UNIVERSIDAD RICARDO PALMA

FACULTAD DE INGENIERÍA

PROGRAMA DE TITULACIÓN POR TESIS

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

PROTOTIPO DE SEÑALIZACIÓN UTILIZANDO RASPBERRY PI 3B E IBM WATSON MEDIANTE RECONOCIMIENTO DE COMANDOS DE VOZ PARA PREVENIR ACCIDENTES.

TESIS

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO ELECTRÓNICO

PRESENTADA POR

Bach. ESCALAYA ANGULO ANTHONY ALEJANDRO

Bach. YACTAYO SÁNCHEZ YELSTIN MAICOL

Asesor: Dr. Ing. HUAMANÍ NAVARRETE, PEDRO FREDDY

LIMA-PERÚ

2020

DEDICATORIA

Dedico esta tesis a mi familia, a mi madre Maritza, por forjar en mi la sencillez y humildad que la caracterizan, y apoyarme constantemente. A mi padre Alejandro, por formar en mi un carácter fuerte e independiente, y ser mi ejemplo a seguir en la vida. A mi hermana Valeria, por darle a mi vida la alegría y diversión que la caracteriza.

Anthony A. Escalaya Angulo.

Dedico esta tesis a mis padres por haberme forjado con su ejemplo para ser la persona que soy en la actualidad; muchos de mis logros se los debo a ustedes entre los que se incluye este y también va dirigido a mis abuelos, son fuente de inspiración y todo logro siempre va a ser por y para ustedes.

Yelstin M. Yactayo Sánchez.

AGRADECIMIENTO

Agradecemos a nuestra casa de estudios la Universidad Ricardo Palma, a la Escuela Profesional de Electrónica y a todos los docentes que nos acompañaron en nuestra formación profesional.

A la Asociación de Ciclistas del Perú y a nuestro compañero Rodrigo Solís Serrano, por brindarnos todo su apoyo en el desarrollo de esta tesis.

Y un especial agradecimiento a nuestras familias, por darnos su incondicional apoyo a lo largo de nuestras vidas.

Anthony Escalaya y Yelstin Maicol.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

Este proyecto propone la implementación de un prototipo de señalización para ciclistas mediante el reconocimiento de comandos de voz. Su funcionamiento se basa en reconocer 3 comandos de voz, derecha, izquierda o alto. El prototipo reconocerá el comando dado independientemente si se pronuncia en una palabra o una frase, teniendo en cuenta el tiempo de grabación de 3 segundos. Para capturar la voz, el Raspberry Pi 3B realiza grabaciones de 3 segundos en intervalos de 3 segundos, y para minimizar el ruido ambiental, se utilizó un filtro Butterworth pasa banda digital con frecuencias de 200 Hz y 1300 Hz y orden 6. La conversión de voz a texto se realizó mediante el servicio de Speech to Text que pertenece al grupo de IBM Watson, el cual sube la información que recibe a la nube de IBM y descarga el texto traducido. Todo este proceso de traducción se realizó en la plataforma Node-Red. Además, se diseñó los prototipos de visualización compuestos por leds, los cuales se conectaron directamente a los pines del Raspberry Pi 3B, quedando a la espera de la señal de excitación que se envía desde el Node-Red. Finalmente, para las pruebas se obtuvo el apoyo de 6 personas, 3 hombres y 3 mujeres, de diferentes edades. Se evaluaron 2 casos principales, sin filtro digital y con filtro digital, y para cada caso se consideraron dos tipos de mascarillas, una de tela de 3 capas y una KN95, también se consideró la presencia y no presencia de ruido ambiental. Los resultados obtenidos al reconocimiento de comandos muestran un promedio del 90%, sin embargo, hay un retardo muy considerable de 7 segundos desde que se dice el comando de voz hasta que se realiza la señalización, pero es una respuesta aceptable si se tiene en cuenta el retardo. Demostrándose el funcionamiento y autonomía del prototipo con una gran escalabilidad.

Palabras claves: Filtro Pasa Banda, Raspberry Pi 3B, Node-Red, IBM Watson, Speech to Text.

ABSTRACT

This project proposes the implementation of a signage prototype for cyclists through the recognition of voice commands. Its operation is based on recognizing 3 voice commands, right, left or stop. The prototype will recognize the given command regardless if it is pronounced in a word or a phrase, taking into account the recording time of 3 seconds. To capture the voice, the Raspberry Pi 3B makes 3-second recordings in 3-second intervals, and to minimize ambient noise, a digital band-pass Butterworth filter was used with frequencies of 200 Hz and 1300 Hz and order 6. The conversion of Speech to text was done through the Speech to Text service that belongs to the IBM Watson group, which uploads the information it receives to the IBM cloud and downloads the translated text. This entire translation process was carried out on the Node-Red platform. In addition, the display prototypes composed of LEDs were designed, which were connected directly to the pins of the Raspberry Pi 3B, waiting for the excitation signal that is sent from the Node-Red. Finally, for the tests, the support of 6 people, 3 men and 3 women, of different ages, was obtained. Two main cases were evaluated, without digital filter and with digital filter, and for each case two types of masks were considered, one made of 3-layer fabric and one KN95, the presence and non-presence of environmental noise were also considered. The results obtained from the command recognition show an average of 90%, however, there is a very considerable delay of 7 seconds from when the voice command is said until the signaling is carried out, but it is an acceptable response if it is taken into account the delay. Demonstrating the operation and autonomy of the prototype with great scalability.

Keywords: Band Pass Filter, Raspberry Pi 3B, Node-Red, IBM Watson, Speech to Text.

INTRODUCCIÓN

Hoy en día, vivimos en un país que requiere diversos cambios para mejorar. Entre los problemas principales tenemos la seguridad ciudadana, la contaminación ambiental y la congestión vehicular que realmente es un caos que todos los ciudadanos pasamos día a día.

Uno de los cambios que se está poniendo en ejecución es la ley Nª 30936 que está vigente desde abril del 2019. Esta ley promueve el uso de la bicicleta como medio de transporte sostenible, siendo una posible solución para disminuir el tráfico.(Ley Nº 30936, 2019) Sin embargo, a pesar de que se promueve la creación de nuevas ciclovías, mayor información sobre educación vial y su constante apoyo por parte del gobierno, esta ley no protege al ciclista de manera íntegra y el ciclista está expuesto a sufrir accidentes, el uso de casco es obligatorio, sin embargo, ante un choque con un automóvil, el casco poco podrá proteger al ciclista.

Por tanto, el presente proyecto de tesis plantea la implementación de un prototipo de señalización utilizando Raspberry Pi 3B e IBM Watson mediante reconocimiento de comandos de voz, de forma que brinde al ciclista una opción para poder avisar la dirección de giro que realiza el ciclista mientras recorre rutas urbanas y de esta manera poder ayudar a prevenir accidentes. Para la investigación, el objetivo principal fue diseñar un prototipo de señalización utilizando Raspberry PI 3B e IBM Watson para reconocer comandos de voz y prevenir accidentes.

Así mismo, para este proyecto la investigación se estructuró de la siguiente manera:

En el capítulo 1 se presenta la formulación del problema, los problemas específicos y problema general, objetivo general y específicos, además de mencionar la importancia y justificación del proyecto, así como los alcances y delimitaciones del mismo.

El capítulo 2 da mención al marco teórico del proyecto, donde se presentan las investigaciones relacionadas con el tema y las bases teóricas. Además del diseño de la investigación.

En el capítulo 3 se describe el desarrollo e implementación de todo el prototipo, se detalla cada aspecto y parte de programación necesaria.

En el capítulo 4 se describen las pruebas realizadas y se detallan los resultados obtenidos en todas las condiciones dadas.

Finalmente, en las conclusiones se describen los resultados obtenidos y se verifica el cumplimiento de los objetivos planteados al inicio del proyecto.

CAPÍTULO I: PLANTEAMIENTO Y DELIMITACIÓN DEL PROBLEMA

1.1. Formulación del problema

Desplazarse en bicicleta por zonas urbanas en nuestro país es todo un desafío. La falta de ciclovías hace que los ciclistas compartan el mismo carril por donde transita los autos y buses, lo que provoca que el aviso de giro manual sea un problema y aumente enormemente la probabilidad de sufrir un accidente de tránsito. Según un informe de la Organización Mundial de la Salud (OMS) (2018), a nivel mundial los peatones y ciclistas representan el 26% de todas las muertes por accidentes de tránsito y en América Latina, sólo los ciclistas representan el 3% que aproximadamente son 4650 muerte al año. En el Perú, en el último año se ha propuesto una nueva ´´ley que promueve y regula el uso de la bicicleta como medio de transporte sostenible´´ (Ley N° 30936, 2019) y la construcción de nuevas ciclovías para brindar mayor seguridad a los ciclistas. Sin embargo, este proceso está comenzando y tomará unos años ver los resultados de la ley propuesta. Por tal razón, el problema de que el ciclista tenga la seguridad al desplazarse y no tener complicaciones cuando requiera indicar su giro, es necesario poder brindar una posible solución con un prototipo de señalización que permita al ciclista indicar su giro mientras recorre zonas urbanas. Sin embargo, esta posible solución presenta delimitaciones porque a pesar de estar obligados por ley a llevar elementos de seguridad como casco, prenda reflectante, luces para ser visibles en la noche y/o lugares con poca visibilidad y timbre; un elemento como un prototipo de señalización para ciclistas no está obligado a llevar como medio para prevenir accidentes.

De esta manera la formulación del problema será:

¿Cómo diseñar un prototipo de señalización utilizando Raspberry Pi 3B e IBM Watson para reconocer comandos de voz y prevenir accidentes?

Y, de los problemas específicos:

- ¿Cómo diseñar un filtro pasa-banda digital en el Raspberry Pi 3B para atenuar el ruido ambiental proveniente del micrófono?

- ¿Cómo reconocer comandos de voz en la plataforma Node-Red a través del servicio IBM Watson speech to text?
- ¿Cómo establecer la comunicación entre el Raspberry Pi 3B y el prototipo de señalización, a través de tres tipos de alerta luminosas?

1.2. Objetivos

1.2.1. Objetivo General

Diseñar un prototipo de señalización utilizando Raspberry Pi 3B e IBM Watson mediante reconocimiento de comandos de voz para prevenir accidentes.

1.2.2. Objetivos Específicos

a) Diseñar un filtro pasa-banda digital en el Raspberry Pi 3B para atenuar el ruido proveniente del micrófono.

b) Reconocer comandos de voz en la plataforma Node-Red a través del servicio de IBM Watson Speech to text.

c) Establecer la comunicación entre el Raspberry Pi 3B y el prototipo de señalización, a través de tres tipos de alerta luminosas.

1.3. Importancia y justificación

En busca de brindar una mayor seguridad al ciclista urbano, se toma en consideración ciertos aspectos: que el ciclista pueda ser reconocido a una distancia prudente y que el ciclista peda indicar con anticipación su dirección de giro, siendo esta última consideración un punto importante para poder prevenir accidentes. Los accidentes en los que se han visto involucrados los ciclistas urbanos, en su mayoría, son ocasionados por la falta de respeto que los conductores de vehículos tienen, por la falta visual del recorrido de los ciclistas o por la dificultad que tienen los ciclistas para indicar su giro.

El presente proyecto de tesis se justifica debido a la capacidad de dar una mayor seguridad al desplazarse en bicicleta y prevenir accidentes mientras se recorren rutas urbanas

1.4. Alcances y limitaciones

Se cuenta con algunas limitaciones como el espacio en el casco donde se instalará el módulo luminoso, de tal forma que no genere incomodidad al ciclista al usarlo. Así

mismo, la batería no es duradera y se considera utilizar un powerbank de 10000mAh para poder ser utilizado por un mayor rango de tiempo, pero teniendo en cuenta el límite de uso que nos ofrece el servicio de IBM Watson. Una limitación es la captura de los comandos de voz entregadas por el ciclista con presencia de ruido. Por tal razón se está optando por utilizar un filtro digital que atenue las frecuencias ajenas al rango de frecuencias de trabajo. Así mismo, el presente proyecto de tesis puede llegar a ser muy útil al contar con un prototipo que puede ser usado por cualquier ciclista que cuenten con un casco. De esta forma, el ciclista al realizar recorridos de rutas largas o bien transitadas podrá evitar accidentes.

CAPÍTULO II: MARCO TEÓRICO

2.1. Marco Histórico

Debido al gran aumento de vehículos en el parque automotor del Perú, los tiempos de viaje de las personas se ha incrementado considerablemente, lo cual obliga a los usuarios a planear las posibles rutas, para evitar zonas con mayor densidad vehicular. Salir de sus hogares más temprano de lo normal para llegar a sus centros de trabajo o estudio a tiempo.

El uso de una bicicleta como una solución siempre está presente, y más aún cuando en tiempo récord se ha incrementado su uso de manera significativa en otros países, debido a esta opción el Perú está empezado a evaluar su uso como un medio de transporte sostenible teniendo como ejemplo a la Unión Europea, la cual cuenta con leyes e infraestructuras bien organizadas en favor a los ciclistas.

En Sudamérica, se toma como referencia a Colombia quien promueve a través de su Ley N°1811, según las leyes de la República colombiana el uso y promoción de la bicicleta como medio de transporte sostenible desde el año 2016. Teniendo en su capital, Bogotá, una infraestructura de 392 kilómetros de ciclovías convirtiéndola en la ciudad con mayor cantidad de ciclovías en Sudamérica, según el artículo del diario virtual “Semana Sostenible” (Urrutia, 2016).

En el Perú, el uso de bicicleta no es muy habitual, según un reporte de Radio Exitosa (2020) sólo el 3% de la población en la ciudad de Lima utiliza este medio como transporte por la poca cantidad de leyes a favor de los ciclistas y accidentes en los que un ciclista estaba involucrado, provocado por la baja educación vial que tiene la mayoría de conductores de automóviles y debido a que los ciclistas se ven obligados a desplazarse en la autopista por la escasa cantidad de infraestructura.

En estos últimos años, nuestro país, se ha empezado a promover el uso de bicicleta con el fin de reducir la gran densidad de vehículos en horas específicas y como una alternativa ecológica por temas del aumento de la contaminación de CO2. El 14 de abril del 2019 el congreso de la República aprobó la Ley N°30936 “Ley que promueve y regula el uso de la bicicleta como medio de transporte sostenible”.

2.2. Investigaciones relacionadas con el Tema.

Según Thakur R. Manoj Kumar Pandey, N. (2018), en su artículo titulado ‘’Filtering of Noise in Audio/ Voice Signal”, tiene como objetivo diseñar diferentes tipos de filtros recursivos como Butterworth, Chebyshev tipo 1, Chevyshev tipo 2 y elliptic, para ser comparados y encontrar el mejor tipo de filtro para voz usando el software de simulación Labview. Entre los datos considerados para el diseño de los filtros se tiene una frecuencia de muestreo de 10000 Hz y el rango de frecuencia a usar es de 100 Hz a 3000 Hz, además se consideraron 4 tipos de orden para poder realizar la comparación entre los filtros. En los resultados se obtiene que para los datos antes mencionados el filtro recursivo Butterworth muestra un mínimo ruido en comparación con los demás.

Por otro lado, según Raquel Ruiz Cañamero (2015), en su tesis titulada “Sistema de señalización para ciclistas” se llevó a cabo el desarrollo de su sistema en una bicicleta. Donde el objetivo principal es poder realizar señalizaciones de giro y parada mediante un sistema inalámbrico basado en tecnología Xbee. Los dispositivos que utiliza para indicar el giro son pulsadores que se encuentran en el manubrio de la bicicleta junto a su controlador y módulo de transmisión, luego son recibidos por un módulo de recepción que se encuentra conectado a un segundo controlador, el cual se encarga de realizar la señalización mediante una plantilla de led que diseñó. Así mismo, las pruebas realizadas le permitieron comprobar la transmisión y recepción de la señal sin tener interferencias ni problemas en la comunicación inalámbrica.

Además, según Daniel Tomala Cuenca (2018), en su tesis titulada “Sistema domótico controlado por voz para personas con discapacidades superiores, utilizando tarjeta Raspberry Pi” se diseñó e implementó un sistema domótico accionado mediante la voz para personas con discapacidades motrices superiores. Utiliza un programa para el reconocimiento de voz que permite transcribir las palabras y luego realizar la acción predefinida. Mediante un micrófono inalámbrico recibe los comandos de voz para luego ser procesada y comparada con una base de datos en el Raspberry Pi y después activar o desactivar los actuadores que tiene establecidos en la vivienda de prueba. En sus resultados obtenidos detalla que uno de sus inconvenientes que presentó es el ruido ambiental, ya que no contaba con

ningún filtro, también observó complicaciones con el lenguaje de programación al recibir comandos con múltiples acciones, pero se solucionó con solo realizar comandos con acciones específicas. Al final de su trabajo, detalló que es muy importante tener en cuenta el nivel del ruido debido a que puede distorsionar los comandos de voz.

Además, según L.J. Gil, L.F. Castillo y R.D. Flórez (2016), en su proyecto titulado "Spanish speech recognition oriented to a wheelchair control" presentan una aplicación computacional que reconoce instrucciones de voz para el manejo de una silla de ruedas automatizada, en donde su trabajo se centra en la captación y filtrado del comando de voz. Teniendo en cuenta los rangos de nivel de ruido ambiental que establecen las normas colombianas, trabajaron con filtros de ponderación frecuencial A y filtros de ponderación frecuencial F obteniendo resultados muy favorables en sus pruebas de tres situaciones de ruido ambiental, pero recomiendan aumentar la voz en lugares de altos niveles de ruido, además de realizar los comandos con palabras claves que tengan más de una sílaba.

También según H. Anoja (2017), en su artículo titulado "Internet of Things using Node-Red and Alexa" tiene como objetivo desarrollar un sistema de bajo costo que pueda controlar diversos actuadores a través de diversos datos que recibe de sensores para controlar diferentes espacios como casas, fábricas, lugares públicos. Este sistema utiliza Raspberry Pi 3 que interactúa con un servidor local creado por Node-Red, así mismo utiliza Alexa Skill, que es un servicio de Amazon Web Services que le permite utilizar la voz para diferentes habilidades como comandos de voz. El desarrollo del sistema se basa en la plataforma Node-Red en el cual realiza toda la programación para los diferentes sensores y actuadores. Como conclusión, la elaboración de este sistema basado en la plataforma Node-Red y el servicio de Alexa de Amazon Web Services fue exitoso y agrega que Node-Red es una plataforma flexible que controla todos los dispositivos de IoT en la nube.

2.3. Bases Teóricas relacionadas con el tema

- Procesamiento digital de señales, es el conjunto de técnicas matemáticas que se aplican a las señales con el fin de mejorar, facilitar y/o manipular la información para un uso específico.

Dentro del procesamiento digital de señales se tiene el proceso de filtrado, el cual es un conjunto de técnicas que se desarrolla como pre procesamiento de señales, asi mismo, tiene como principales objetivos separar y/o restaurar señales. Una aplicación del uso de separación de señales es dividir el ruido existente de una señal. A continuación, en la Figura N°1 se muestra un diagrama de bloques del proceso de filtrado de señal.

Figura Nº 1: Proceso de filtrado de señal.

Fuente: Elaboración propia.

- Software Matlab: Según H. Moore (2007), es un software de computación científica para optimizar la resolución de problemas de ingeniería, así como también del tipo científico.

  El presente software es utilizado en diversos centros de investigación, universidades e institutos. Posee diversas aplicaciones desde ser una herramienta matemática hasta el desarrollo complejo de diversos proyectos tecnológicos. Adicionalmente, Matlab posee diversas librerías y aplicaciones como son el caso de las interfaces gráficas, entre las cuales tenemos: Simulink, GUIDE y App Designer, cada una de estas cuentan con una infinidad de aplicaciones.

  La ventaja del uso del software Matlab es que cada año está en constante actualización y es compatible con diversas tecnologías como Arduino, Raspberry, entre otras.

- IBM Cloud: Según IBM (2020), es una plataforma en la nube de IBM, combina una plataforma como servicio (PaaS) con la infraestructura como servicio (IaaS) para proporcionar una experiencia integrada. En esta plataforma se puede desarrollar proyectos multidisciplinarios, lo cuales utilizan diversos servicios de la nube para brindar la mejor solución posible.

- Watson speech to text: Según IBM (2020), es un servicio de IBM que cuenta con un potente reconocimiento de voz, precisión y poder de transcripción para convertir lo hablado en texto.

- Node-Red: Según JS Foundation (2019) es una herramienta de programación para conectar los diferentes dispositivos de hardware, aplicativos y servicios online de diversas formas innovadoras. Se basa en la conexión de flujos entre si utilizando nodos de diversas aplicaciones que se pueden implementar de manera sencilla.

2.4. Diseño de la Investigación

2.4.1. Variables de investigación

Las variables a considerar son:

- Variable independiente: Reconocimiento de comando de voz.
- Variable dependiente: Prototipo de señalización.

2.4.2. Tipo y Método de investigación

El tipo de investigación es aplicada y tecnológica. El método de investigación es del tipo empírico y experimental. Esto se debe a que se adquirirán señales de audio a través de un micrófono que serán procesados. Luego, se convertirá a texto a través del servicio de IBM Watson Speech to text en la plataforma Node-Red, para posteriormente ser procesado con un algoritmo de reconocimiento de caracteres.

2.4.3. Técnicas e Instrumentos de recolección de datos

Se recolectarán datos analógicos por medio de un micrófono USB conectado a una tarjeta de desarrollo Raspberry Pi 3B.

CAPÍTULO III: DESARROLLO DEL PROYECTO

En el presente capítulo se muestran las diferentes etapas en la cual se ha desarrollado el diseño e implementación del software y hardware del prototipo propuesto.

3.1. Diagrama de Bloques General

Seguidamente, en la Figura N° 2, se muestra el diagrama de bloques general del proyecto de tesis planteado. Donde, se observan tres bloques principales que se denominan: entrada, procesamiento y salida, respectivamente. El bloque de entrada, o también denominado de adquisición de voz, está conformado por un mini micrófono USB, el bloque de procesamiento está conformado por el miniordenador Raspberry Pi 3B y el bloque de salida está conformado por el prototipo visualizador.

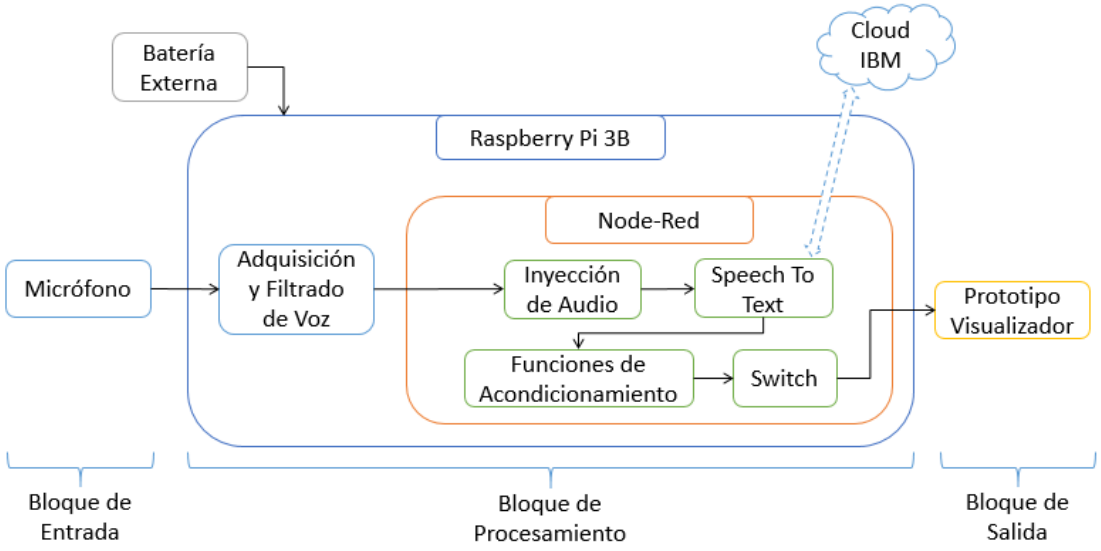

Figura Nº 2: Diagrama de bloques general

Fuente: Elaboración propia.

3.2. Bloque de Entrada

Este bloque también se denomina bloque de adquisición de voz está formado por un mini micrófono USB, el cual tiene como características técnicas la interfaz que es USB 2.0, la sensibilidad de -67dB, el rango de frecuencias entre los 100Hz y 16KHz, un peso de 8gr, una dimensión de 21mmx17mmx8mm, compatibilidad con los sistemas operativos Windows, Linux, OSX, así como también compatibilidad con las diversas tecnologías que posee la familia Raspberry. A continuación, la Figura N°3 muestra una imagen del mini micrófono utilizado.

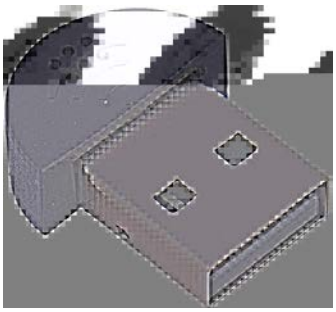

Figura Nº 3: Mini micrófono USB

Fuente: Amazon.

3.3. Bloque de Procesamiento

Este bloque está conformado por el miniordenador Raspberry Pi 3B, en el cual se realizó la instalación y configuración del sistema operativo, además de las rutinas de programación.

El modelo seleccionado es Raspberry Pi 3B como se viene mencionando desde un inicio, si bien ya existe un modelo más novedoso con mejores características se optó por conservar el modelo mencionado ya que las características satisfacen las necesidades del proyecto. Según la página web oficial de Raspberry, en la descripción de sus productos (s.f.). recuperado de https://www.raspberrypi.org/products/raspberry-pi-3-model-b/, las especificaciones técnicas del modelo son:

- CPU de cuatro núcleos a 1,2 GHz Broadcom BCM2837 de 64 bits
- 1 GB de RAM
- BCM43438 LAN inalámbrica y Bluetooth de baja energía (BLE) a bordo
- 100 Base Ethernet
- GPIO extendido de 40 pines
- 4 puertos USB 2
- Salida estéreo de 4 polos y puerto de video compuesto
- HDMI de tamaño completo
- Puerto de cámara CSI para conectar una cámara Raspberry Pi
- Puerto de pantalla DSI para conectar una pantalla táctil Raspberry Pi

- Puerto micro SD para cargar su sistema operativo y almacenar datos
- Fuente de alimentación micro USB conmutada mejorada de hasta 2,5 A

En las Figura N° 4 se muestra el Raspberry Pi 3B

Figura Nº 4: Raspberry Pi 3B

Fuente: raspberrypi.org.

Con respecto al sistema operativo, se trabajó con Raspberry Pi OS, anteriormente llamado Raspbian, siendo el sistema operativo oficial y más recomendado. De las tres opciones que ofrece la página oficial de Raspberry, se decidió trabajar con el sistema operativo de 32 bits con escritorio y software recomendado, de esta forma cuenta con un escritorio que hará más amigable el trabajo, además de tener varios softwares de programación dentro de los cuales se encuentra Node-Red que es el ambiente de programación en al cual se trabajó. Todo el sistema operativo se instaló en una memoria micro SD de 16GB.

Dentro del Raspberry Pi 3B se encuentran dos bloques secundarios, el bloque de adquisición y filtrado de voz y el bloque de Node-Red. Cada bloque cuenta con una descripción que a continuación se detalla.

3.3.1. Adquisición y Filtrado de Voz

En el bloque filtro digital se encuentra la programación del filtro digital pasabanda el cual se ha diseñado para filtrar las frecuencias de audio ajenas al rango de trabajo, que son entre 100Hz-1.3KHz, denominando estas frecuencias ajenas como ruido. Al disminuir el ruido, que en su mayoría se produce por el ambiente en que se encuentra la persona, mejora en gran parte la señal de tal forma que su procesamiento es más eficiente.

Para el diseño del filtro digital pasabanda se procedió a tomar como referencia el rango de frecuencia en que la voz es escuchada, que va de

100hz a 3khz. Se tomó como diseño un filtro recursivo que por trabajos anteriores según Meiniar W. Ayu F. Irmasari I. Mukti A. Astharini D. (2017) nos indican que tiene una mejor respuesta cuando se utiliza como filtro para voz.

Así mismo, en primera instancia se realizó el diseño y las prueba en el software Matlab para ver la respuesta del filtro digital y verificar el correcto funcionamiento. Posteriormente se procedió a obtener la función de transferencia con la cual podemos utilizar el filtro en diversas plataformas.

El bloque de filtro digital cuenta con una programación en lenguaje java, en el cual contiene la función de transferencia y procede a filtrar las frecuencias ajenas al rango establecido anteriormente. A continuación, en la Figura N° 5 se muestra la función de transferencia del filtro pasabanda digital de orden 6.

$$\frac{1.703e\text{-}07\ z^{12} - 1.022e\text{-}06\ z^{10} + 2.554e\text{-}06\ z^{8} - 3.406e\text{-}06\ z^{6} + 2.554e\text{-}06\ z^{4} - 1.022e\text{-}06\ z^{2} + 1.703e\text{-}07}{z^{12} - 11.36\ z^{11} + 59.24\ z^{10} - 187.3\ z^{9} + 399.8\ z^{8} - 607.6\ z^{7} + 673.6\ z^{6} - 549.1\ z^{5} + 326.7\ z^{4} - 138.3\ z^{3} + 39.54\ z^{2} - 6.856\ z + 0.5454}$$

Figura Nº 5: Función de transferencia de filtro pasa-banda digital
Fuente: Elaboración propia

Asimismo, en Figura N° 6 se muestra la respuesta en frecuencia del filtro digital pasa-banda diseñado.

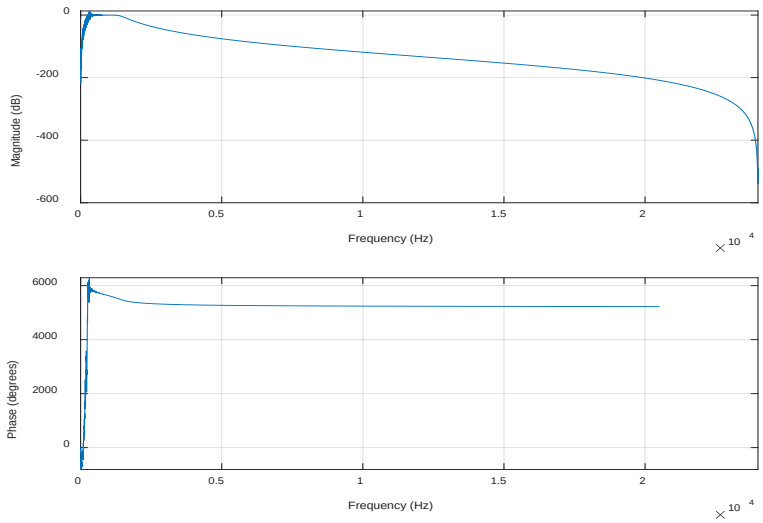

Figura Nº 6: Función de transferencia de filtro digital pasa-banda

Fuente: Elaboración propia

Con los datos obtenidos anteriormente y habiendo comprobado el filtro en el dominio de la frecuencia, realizamos la programación del algoritmo que adquiere y filtra la voz. En este algoritmo se tuvo las siguientes consideraciones, se grabó segmentos de audio con duración de 3 segundos y con intervalos de 3 segundos, se agregó el encendido de un led el cual indica el momento en el que se está realizando la grabación y, por último, el algoritmo guarda la grabación en un archivo de sonido de formato WAV. El algoritmo final en Python se muestra a continuación en la Figura N° 7.

```python
#!/usr/bin/env python3
import pyaudio
import wave
import RPi.GPIO as GPIO
import time
from io import open
import numpy as np
from scipy import signal
while True:
        CHUNK = 512
        FORMAT = pyaudio.paInt16 #paInt8
        CHANNELS = 1
        RATE = 44100 #sample rate
        RECORD_SECONDS = 3
        WAVE_OUTPUT_FILENAME = "grabar.wav"
        pin = 7
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BOARD)
        GPIO.setup(pin, GPIO.OUT)
        p = pyaudio.PyAudio()
        stream = p.open(format=FORMAT,
                    channels=CHANNELS,
                    rate=RATE,
                    input=True,
                    frames_per_buffer=CHUNK) #buffer
        print("* recording")
        GPIO.output(pin, GPIO.HIGH)
        frames = []
        for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
            data = stream.read(CHUNK)
            frames.append(data) # 2 bytes(16 bits) per channel
        print("* done recording")
        GPIO.output(pin, GPIO.LOW)
        stream.stop_stream()
        stream.close()
        p.terminate()
        wf = wave.open(WAVE_OUTPUT_FILENAME, 'wb')
        wf.setnchannels(CHANNELS)
        wf.setsampwidth(p.get_sample_size(FORMAT))
        wf.setframerate(RATE)
        wf.writeframes(b''.join(frames))
        wf.close()

        #carga de archivo de audio
        audioin=mixer.Sound('grabar.wav')
        #filtro matlab
        b=[0.00000017 0 -0.000001022 0 0.000002554 0 -0.000003406 0 0.000002554 0 -0.000001022 0 0.00000017]
        a=[1 -11.3645 59.2353 -187.2537 399.8429 -607.5651 673.6464 -549.1462 326.65 -138.27 39.5359 -6.8562 0.5454]
        #proceso de filtrado
        audioout=signal.filtfilt(b, a, audioin)

        time.sleep(3)
```

```
pi@raspberrypi: ~
Archivo  Editar  Pestañas  Ayuda
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unlock
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for -1, skipping unlock
* recording
* done recording
```

Figura Nº 7: Algoritmo adquisición y filtrado de voz en Python

Fuente: Elaboración propia

3.3.2. Node-Red

Como detalla en la presentación de su página web (s.f.) recuperado de https://nodered.org/, Node-Red, es una herramienta de programación basada en flujo, que fue originalmente desarrollada por el equipo de Servicios de Tecnología Emergente de IBM. Su programación se basa en flujo y describe

su comportamiento como una red de bloques, que reciben la nominación de nodos como se muestra en la Figura N° 8. Cada uno de estos nodos cumple una determinada función, se les asignan datos, lo procesan y luego transmiten al siguiente nodo.

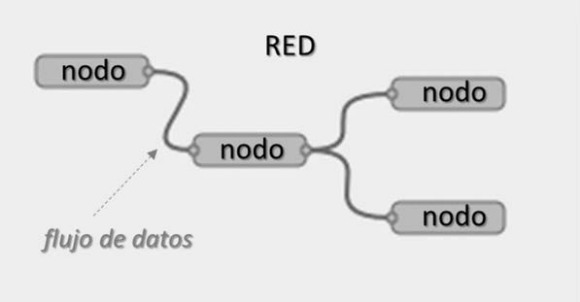

Figura Nº 8: Esquema de flujo de datos en nodos

Fuente: https://www.techedgegroup.com/es/blog/fundamenos-node-red

La herramienta Node-Red tiene un tiempo de ejecución basado en Node.js, ideado como un entorno de programación de JavaScript que se orienta a un navegador web para ser ejecutado. Los diferentes tipos de nodos se encuentran agrupados en una paleta al lado izquierdo, y para poder usarlos solo basta con arrastrar el nodo que deseamos al espacio de trabajo que se encuentra al centro de la pantalla. Las paletas de nodo son versátiles y se pueden ampliar fácilmente instalando nuevos nodos diseñados por otros usuarios de la comunidad de Node-Red.

Además, Node-Red es nativo de Raspberry, esto quiere decir que viene empaquetado en la lista de software recomendados y se encuentra pre instalado en el sistema operativo, por lo que solo se debe actualizar. La forma de comenzar a usar Node-Red es iniciar la aplicación que se encuentra en el menú, dentro de la pestaña de programación, y una vez iniciado el sistema aparecerá una ventana como se muestra en la Figura N° 9 donde se podrá observar como carga todo lo requerido para levantar el software.

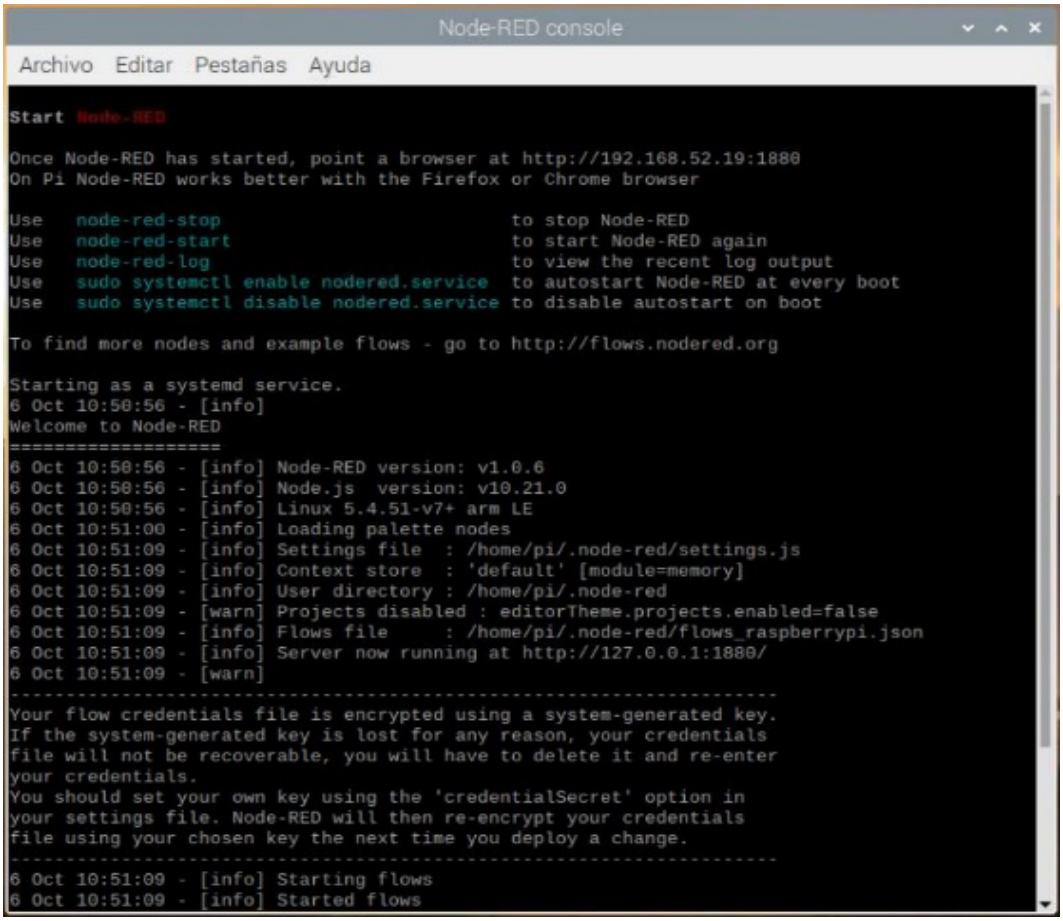

Figura Nº 9: Aplicación de Node-Red

Fuente: Elaboración propia

Luego de que el programa haya cargado correctamente, se inicia la aplicación en un navegador web, en Raspberry Pi OS el navegador por defecto es Chromium, en donde hay dos opciones: se ingresa a la dirección IP del Raspberry con la extensión 1880, http://192.168.52.19:1880; o se ingresa a la dirección http://127.0.0.1:1880/. Se recomienda esta última opción por un tema de permiso de uso del micrófono. A continuación, en la Figura N° 10 se muestra la ventana de trabajo de Node-Red en Raspberry Pi OS.

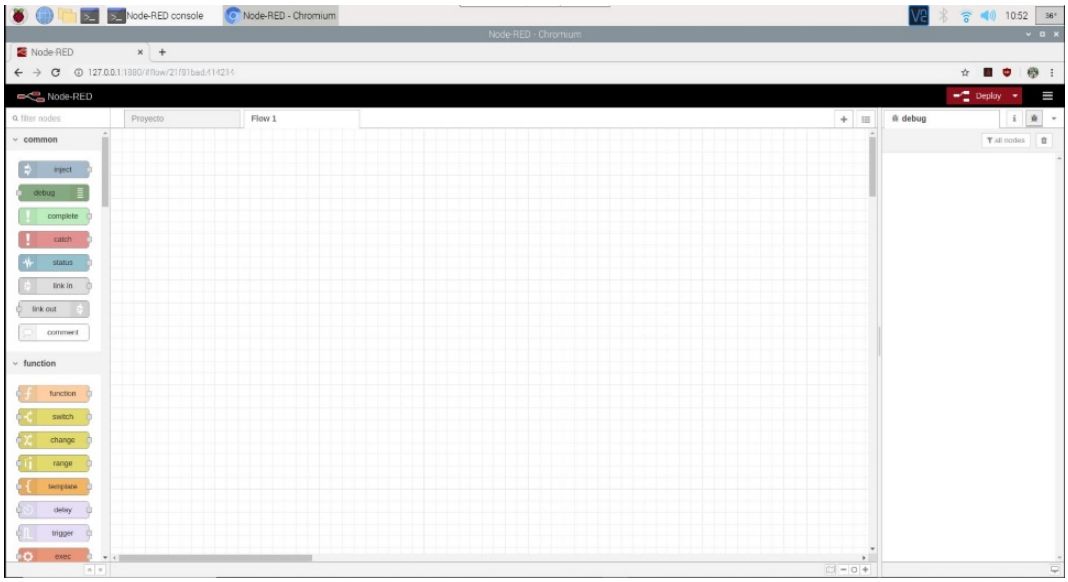

Figura Nº 10: Ambiente de Node-Red

Fuente: Elaboración propia

El bloque Node-Red está conformado por 4 nodos principales, estos son el nodo inyección de audio, el nodo Speech to Text, el nodo funciones de acondicionamiento y el nodo switch. Cada uno de estos nodos cumplen una tarea específica que se detalla a continuación, no sin antes comentar sobre IBM Cloud que permite el acceso a los diferentes servicios que se requieren.

3.3.2.1. IBM Cloud

Según su página web de IBM, en la sección de Cloud (s.f.) recuperado de https://www.ibm.com//pe-es/cloud?lnk=mpr_bucl_pees&lnk2=learn,
definen el Cloud de IBM como una plataforma de multinube híbrida de próxima generación con recursos avanzados de datos e inteligencia artificial, y una profunda experiencia empresarial en 20 industrias ya que su principal objetivo es ofrecer la nube pública más segura y abierta para las empresas.

IBM Cloud cuenta con más de 170 productos y servicios que se extiende a datos, contenedores, inteligencia artificial (IA), internet de las cosas (IoT) y blockchain. Este proyecto de tesis se enfoca a un grupo de servicios en especial de inteligencia artificial, IBM Watson. IBM Watson ofrece servicios de procesamiento de lenguaje natural, reconocimiento visual y machine learning.

Para el uso de todos los servicios que brinda IBM Cloud, solo debe registrarse y crear una cuenta gratuita. Todos los servicios que brindan son de pago como también libres, la única limitación con los servicios libres o gratuitos es el tiempo o almacenamiento de uso. Pero IBM apoya y fomenta el desarrollo de proyectos por lo que el tiempo que brinda para el uso de los servicios de forma gratuita, es considerable y suficiente. A continuación, en la Figura N° 11 se muestra la imagen del panel de control del IBM Cloud.

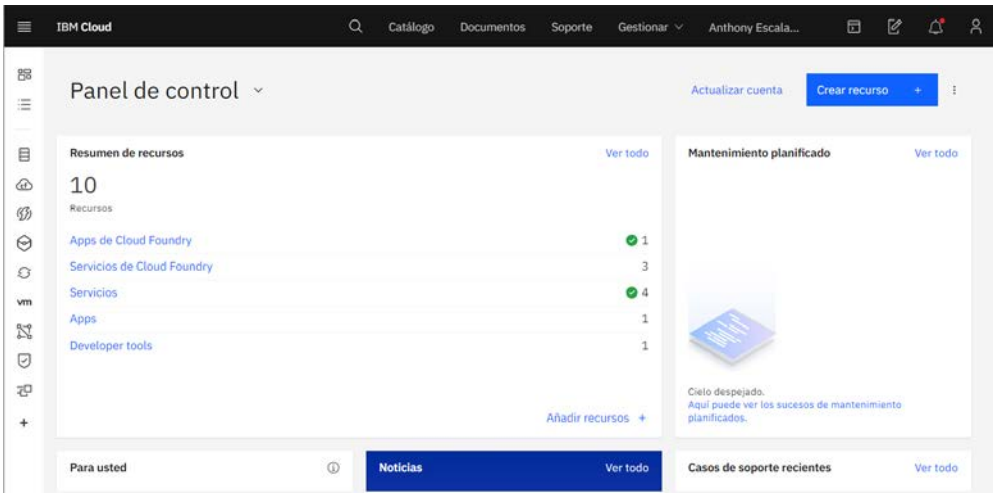

Figura Nº 11: Panel de control IBM Cloud

Fuente: Recuperado de https://cloud.ibm.com/

Después de obtener una cuenta, lo que resta es suscribir o crear los servicios. Este proyecto se enfoca en uno de los servicios de IBM Watson: Speech to Text, y adicionalmente del servicio de Node-Red y su interacción dentro de este, ya que hay que realizar algunas configuraciones adicionales si trabajamos en el Node-Red de forma local.

Los servicios de IBM Watson se encuentran en la categoría de IA/Machine Learning del catálogo, una vez dentro de esta categoría el servicio Speech to Text será uno de los principales en sugerir. Al ingresar a este servicio, se dan algunas especificaciones y lo más importante el tipo de plan que se desea. Speech to Text cuenta con un plan Lite gratuito, el cual brinda 500 minutos de uso al mes, pero este tiempo es suficiente para poder desarrollar un proyecto. Además, solo se permite tener un plan Lite a la vez.

Luego de tener el servicio de Speech to Text creado, se procede a generar las credenciales, las cuales permitirán usar el servicio en cualquier

ambiente de programación. Estas credenciales son las que darán acceso al IBM Cloud desde el Raspberry, para poder cargar datos, se procese y retorne convertido. En la Figura N° 12 se muestra el servicio de Speech to Text con sus respectivas credenciales.

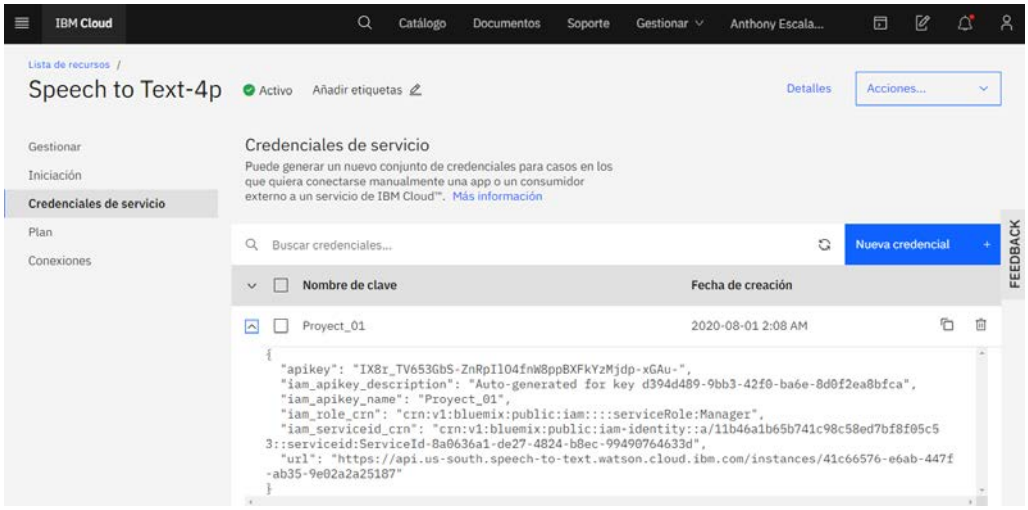

Figura Nº 12: Credenciales servicio Speech to Text

Fuente: Recuperado de https://cloud.ibm.com/services/speech-to-text/

Más adelante, se abordará la relación entre las credenciales y el Node-Red nativo de Raspberry.

3.3.2.2. Inyección de Audio

Una vez se tenga el audio filtrado y guardado en la ubicación específica, cómo se mencionó anteriormente, se procede a insertar este archivo de audio al Node-Red para ser procesado. En este caso se utiliza un nodo en particular denominado File in (ver Figura N° 13), este nodo se encarga de leer el archivo especificado en su configuración y envía el contenido como msg.payload. Así mismo, dentro de las configuraciones del nodo se debe especificar el tipo de mensaje que saldrá, en este caso será single buffer object. Estas configuraciones mencionadas se pueden apreciar en la Figura N° 13.

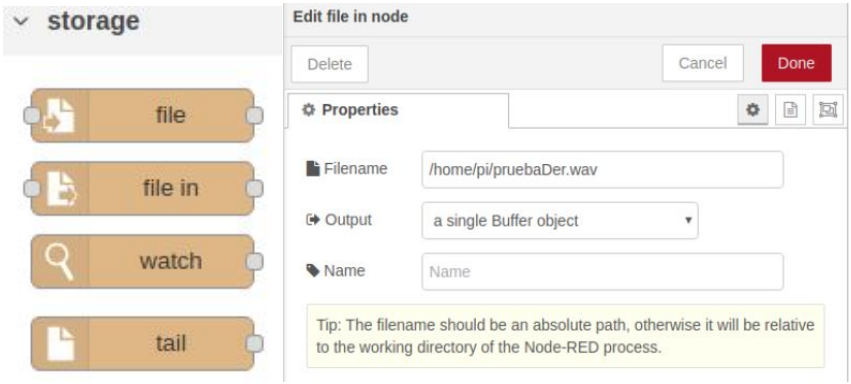

Figura Nº 13: Nodo File in y su configuración

Fuente: Elaboración propia

Además del nodo File in, se necesita un nodo que inicie el proceso de carga de archivo periódicamente y se seleccionó el nodo Inject (ver Figura N° 14). Este nodo puede inyectar diferentes tipos de mensajes, pero en este caso se requiere que inyecte una marca de tiempo o timestamp. Además de activar la repetición en opción de intervalos de 3 segundos. A continuación, se muestra las configuraciones del nodo en la Figura N° 14

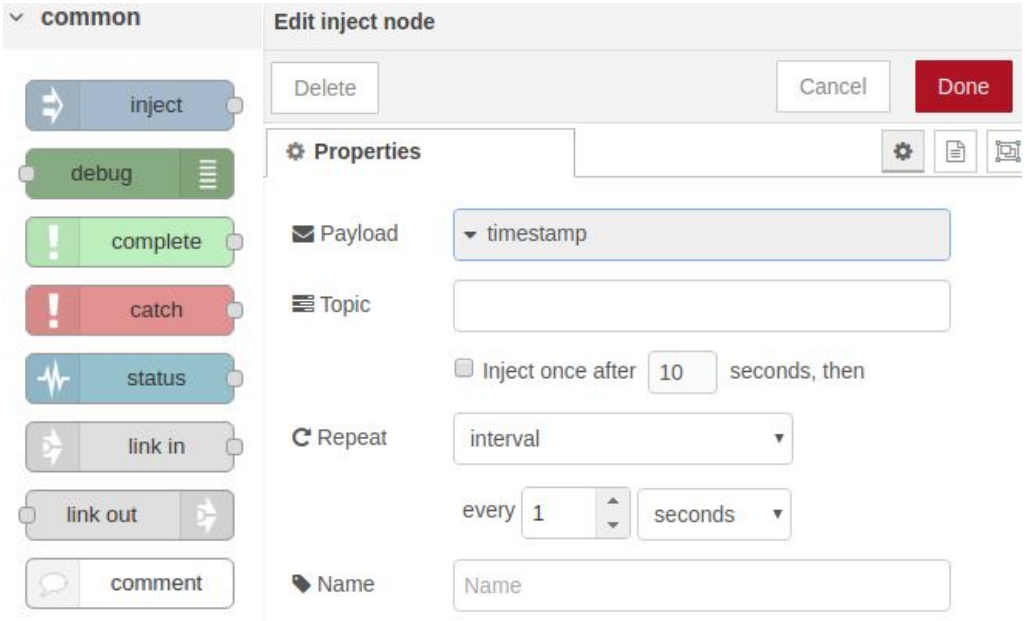

Figura Nº 14: Nodo Inject y su configuración

Fuente: Elaboración propia

De tal forma, se tiene que el nodo inyección de audio está compuesto por dos nodos simples, y la conexión de estos se aprecia en la Figura N° 15.

Figura Nº 15: Nodo compuesto inyección de audio

Fuente: Elaboración propia

3.3.2.3. Nodo Speech To Text

Anteriormente se detalló cómo generar las credenciales necesarias para el uso del servicio de Speech to Text. Este sub capitulo se enfocará en el nodo Speech to Text de Node-Red.

Antes de iniciar a diseñar una red se necesita instalar la paleta correspondiente a Watson IBM, en el menú Node-Red se selecciona importar paletas, luego instalar y busca la paleta node-red-node-watson. Esta paleta cuenta con 22 nodos los cuales pertenecen a todos los servicios de IBM compatibles con Node-Red, como se muestra a continuación en la Figura N° 16, se observa la paleta de Watson con algunos de sus nodos.

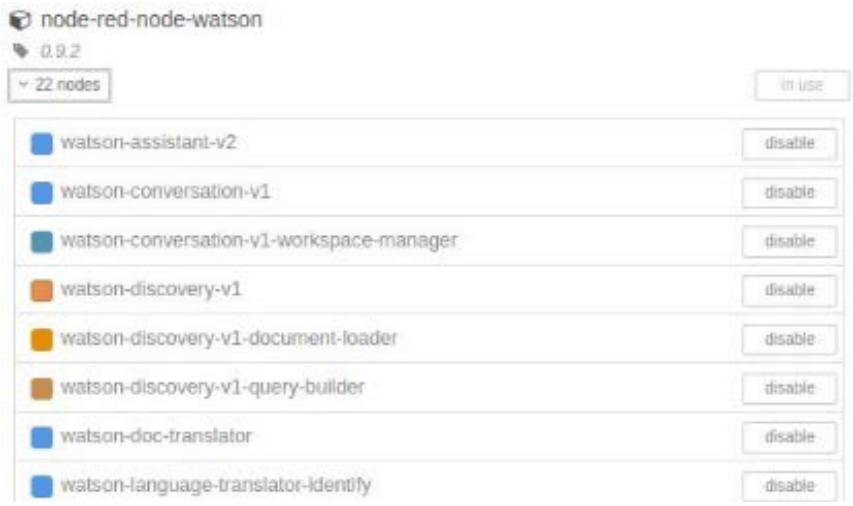

Figura Nº 16: Paleta node-red-watson

Fuente: Elaboración propia

Una vez la paleta este instalada, se busca e inserta al área de trabajo el nodo Speech to Text. El nodo se muestra en la Figura N° 17

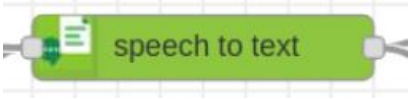

Figura Nº 17: Nodo Speech to text

Fuente: Elaboración propia

Por último, se configura el nodo ya que sin las credenciales este nodo no funcionará. Se ingresa el password y API Key que se encuentran en las credenciales como iam_apikey_description y apikey respectivamente. También se selecciona el lenguaje, los demás valores ya están pre establecidos. En la Figura N° 18 se muestra la configuración del nodo.

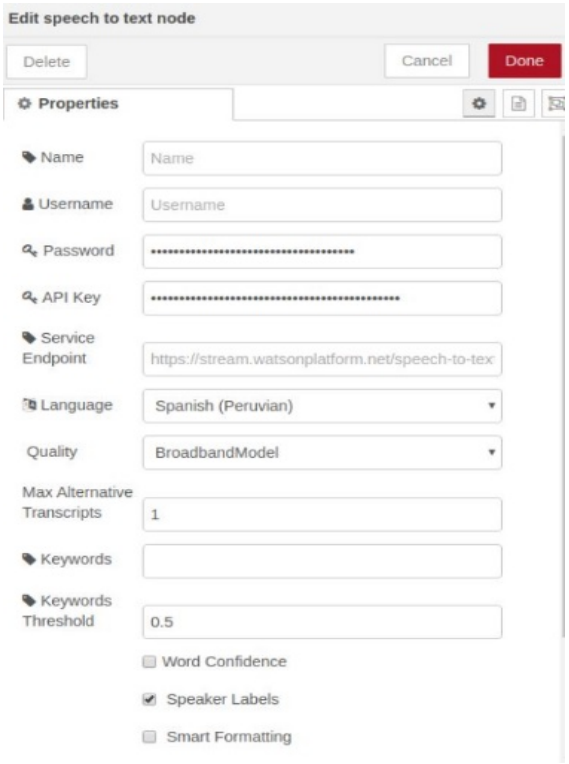

Figura Nº 18: Configuración Nodo Speech to text

Fuente: Elaboración propia

Cabe recalcar que el texto entregado por este nodo se encuentra en una extensión del mensaje o variable. El nodo Speech to Text guarda la transcripción en la variable msg.trancription.

3.3.2.4.Funciones de Acondicionamiento

Este nodo compuesto está conformado por 03 nodos de función, los cuales cumplen detalles específicos que ayudan a tener un mejor procesamiento del mensaje o variable. En la siguiente Figura N° 19 se observan los 03 nodos mencionados.

Figura Nº 19: Nodo compuesto funciones de acondicionamiento

Fuente: Elaboración propia

El nodo de función permite la programación de códigos o algoritmos en lenguaje JavaScript, para el procesamiento de los datos recibidos o realizar acciones específicas. El primer nodo de función usado se denomina Payload y su función es mover la carga del mensaje de msg.trancription a msg.payload. Esto se hace para facilitar la programación y configuración en los siguientes nodos ya que la mayoría de nodos trabaja con msg.payload. La programación en este nodo es muy sencilla y se puede apreciar en la Figura N° 20.

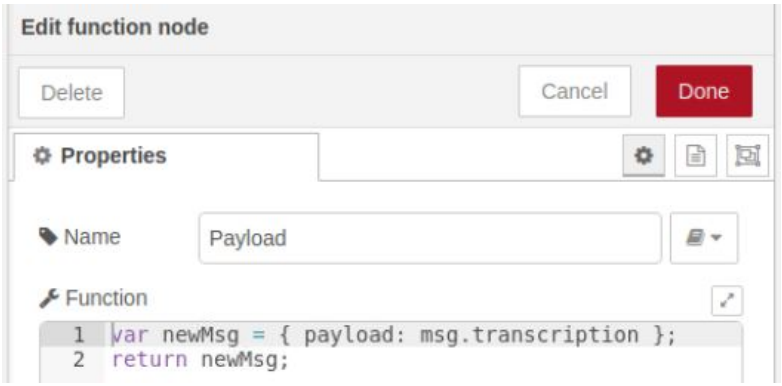

Figura Nº 20: Programación nodo función Payload

Fuente: Elaboración propia

El segundo nodo de función se denomina Separar Palabras, y como su nombre lo indica, se encarga de separar las palabras del mensaje en caso se reciba una frase. El nodo se encarga de separar palabra por palabra y enviarlos en mensajes individuales hacia el siguiente nodo. Este algoritmo se consigue mediante la detección de espacios entre las palabras y se aprecia el código en la Figura N° 21

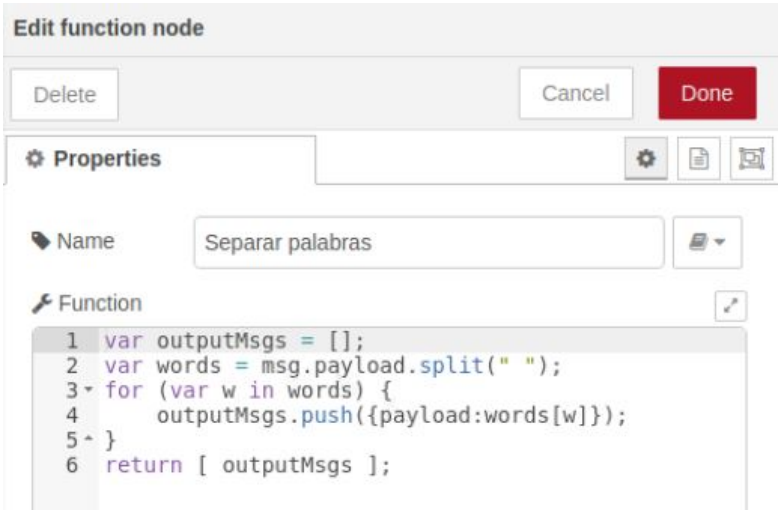

Figura Nº 21: Programación nodo función Separar palabras

Fuente: Elaboración propia

El tercer y último nodo de función denominado Plural, se encarga de identificar los comandos de activación, pero en plural y es que, al realizar algunas pruebas el nodo Speech to Text traduce los comandos como derechas, izquierdas o altos. Para no tener falsos errores con los nodos siguientes, se reconoce el comando en plural y se le remplaza con el mismo comando en singular. De esta forma se evita los posibles errores de transcripción y se puede apreciar la programación en la Figura N° 22.

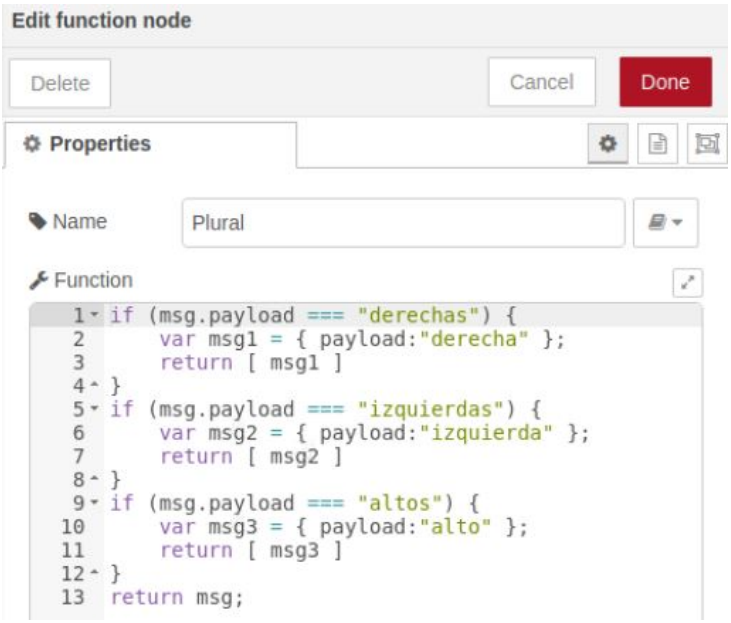

Figura Nº 22: Programación nodo función Plural

Fuente: Elaboración Propia

3.3.2.5. Nodo Switch

El nodo switch tiene una función simple, es el filtro que deja pasar los mensajes. Después de haber transcrito y acondicionado los mensajes, el nodo switch realiza la comparación de cada mensaje que ingresa, con las palabras de control que tiene programado y que en este caso son los comandos de derecha, izquierda y alto. Este nodo tiene tres salidas, una por cada palabra de control, así direcciona los mensajes a los pines correspondientes para su excitación. En la Figura N° 23 se observa la configuración de nodo switch

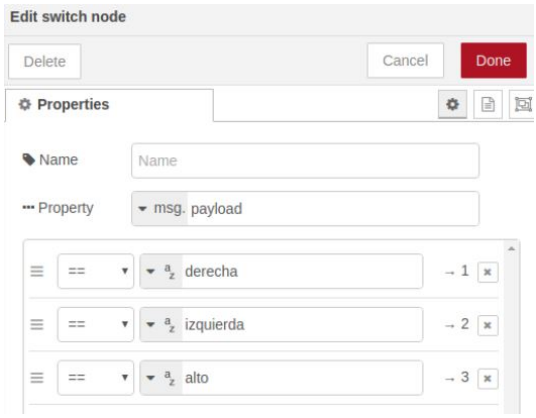

Figura Nº 23: Configuración nodo switch

Fuente: Elaboración Propia

Adicional a esto, a cada salida se le conectó un nodo Trigger para que realiza la función de encendido y apagado intermitente. Seguidamente, se conectó a cada salida un nodo RPI GPIO con el cual se designó el pin del Raspberry Pi 3B al cual se conecta el módulo de visualización. En la Figura N° 24 se muestra la distribución de todos los nodos antes mencionados.

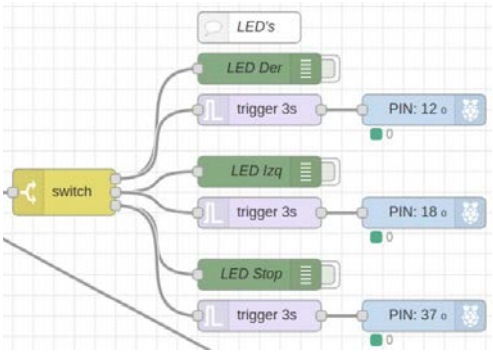

Figura Nº 24: Distribución nodo compuesto switch

Fuente: Elaboración propia

3.4. Bloque de Salida

El bloque de salida está conformado por el prototipo de visualización, el cual consta de 03 plantillas LED’s que representan cada uno de los comandos. Para el desarrollo de las plantillas se procedió a utilizar el software Proteus para su diseño. Para la primera etapa del diseño de las plantillas LED’s se elaboró el diagrama esquemático con los componentes ha utilizarse. A continuación, en la figura N° 25 se muestra el esquemático de las plantillas LED`s

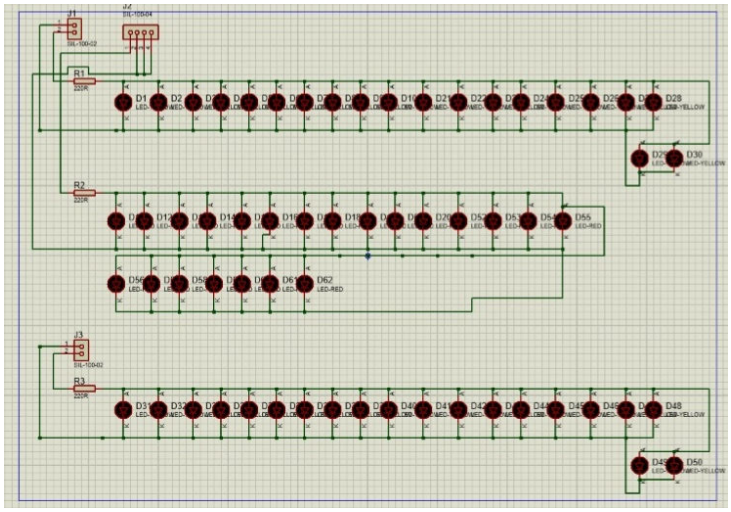

Figura Nº 25: Esquemático de plantilla de LED|s

Fuente: Elaboración propia

Seguido a ello, se procedió a elaborar el modelado de las pistas en tarjeta PCB. En esta etapa se procedió a dar forma el diseño de flechas para indicar dirección tanto izquierda como derecha y un triangulo que representa una señal de alerta. A continuación, en la Figura N° 26 se muestra el resultado del diseño de las pistas en PCB de las plantillas de LED.

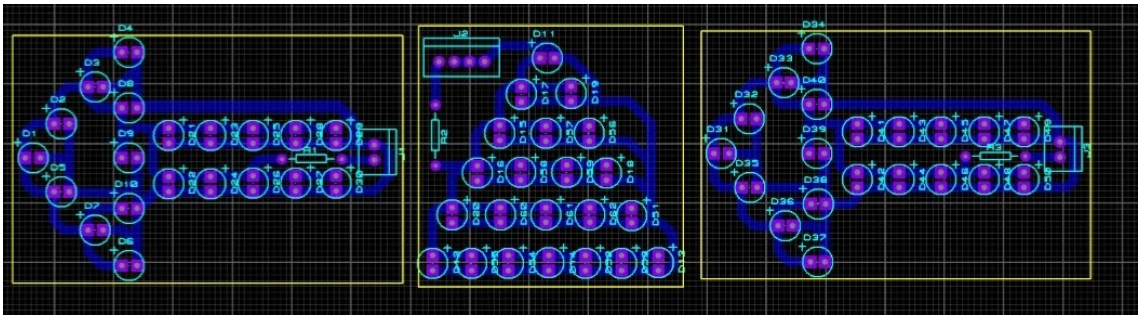

Figura Nº 26: Pistas en PCB de las plantillas de LED|s

Fuente: Elaboración propia

En adición, para tener una mejor visualización del resultado propuesto de las plantillas LED’s se procedió a una visualización en 3D. En la Figura N° 27 se muestra el resultado en 3D del diseño propuesto de las plantillas LED’s.

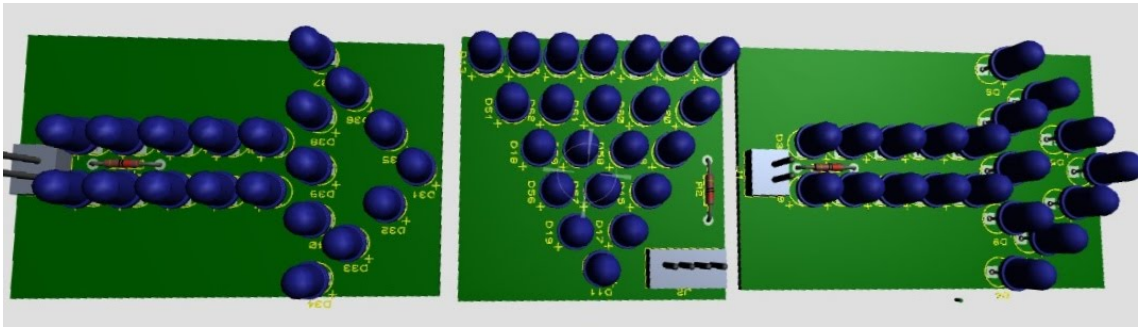

Figura Nº 27: Modelo 3D de plantilla de LED|s

Fuente: Elaboración propia

Adicionalmente a los prototipos de visualización, se tiene el LED de grabación que por ser uno solo no lleva ni plantilla ni una tarjeta PCB. Para este LED solo se realizó la conexión mediante jumpers.

3.5. Batería Externa

El energizado del prototipo se basa en una batería externa portátil que tiene las siguientes especificaciones técnicas

- Model: TS-D218
- Dimensiones: 3.8*2.1*1 pulg.
- Peso: 230g
- Input: 5V, 2A
- Output 5V, 2A
- Battery Power: 10050mAh
- Port Type: Type-C, Micro USB, USB
- Battery Type: Lithium Lion

A continuación, en la figura 28 se muestra la batería externa.

Figura Nº 28: Batería Externa PISEN

Fuente: Recuperado de https://www.lazada.com.my/

3.6. Acondicionamiento del Prototipo

El prototipo consta principalmente de un casco básico de ciclista en el cual se montó los componentes mencionados anteriormente. Sin entrar mucho en detalle, el casco es de un modelo bastante simple fabricado con espuma de ABS (acrilonitrilo butadieno estireno).

Para tener mayor comodidad al realizar todas las conexiones necesarias, se ancló un case de Raspberry en la parte superior del casco el cual contiene y protege el miniordenador Raspberry Pi 3B. Si bien este case modifica mínimamente la estructura del casco, no compromete su seguridad estructural. Además, el contar con el miniordenador Raspberry Pi 3B en el casco ayuda a proteger las conexiones con los demás dispositivos, minimizando posibles errores. En la Figura N° 29 se muestra una imagen parcial del case y Raspberry Pi 3B acondicionados en el prototipo.

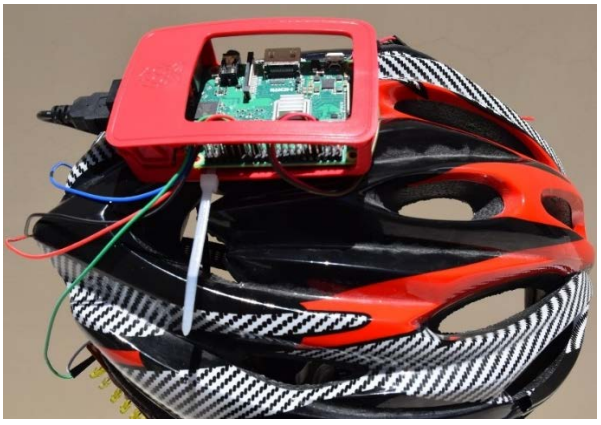

Figura Nº 29: Case y Raspberry Pi 3B implementado

Fuente: Elaboración propia

Para el acondicionamiento del micrófono, se buscó ubicarlo en una posición que se encuentre lo más cercano a la boca, y de esta manera captar mejor la voz. Esto se logró mediante un alambre de cobre de electricidad calibre 14 el cual sujeta el micrófono al casco. De esta forma se da seguridad y flexibilidad al micrófono para ser reacomodado si es necesario. Por otro lado, el micrófono cuenta con una única cara por la cual ingresa el sonido, esta cara ha sido direccionada hacia la boca de la persona para aprovechar la voz y minimizar en pequeña parte el ruido ambiental. También, se conectó siguiendo el mismo camino, el LED que indica el momento de grabación de audio. Este LED se encuentra a una altura superior a la vista, pero dentro del campo periférico de la vista para poder ser percibido, pero sin impedir la visión del ciclista, En la Figura N° 30 se muestra una imagen parcial del micrófono y su ubicación, así como también del LED de grabación.

Figura Nº 30: Microfono y LED grabacion implementado

Fuente: Elaboración propia

Los prototipos de visualización se añadieron a la parte posterior del casco, estos se sujetaron permanentemente con un pegamento para garantizar su adhesión. La conexión de los prototipos de visualización se hizo mediante cables jumpers que se despliegan al interior del casco. En la Figura N° 31 se muestra una imagen parcial de los prototipos de visualización y su conexión.

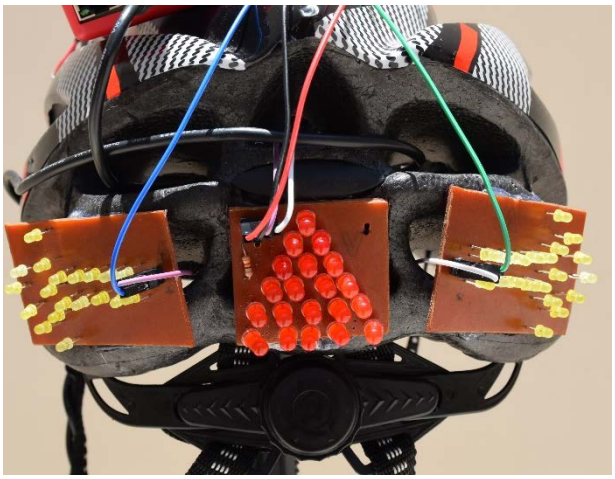

Figura Nº 31: Prototipo de visualización implementado

Fuente: Elaboración propia

La batería externa, que se encarga de alimentar todo el prototipo, se colocó fuera del casco debido a que por su peso daba instabilidad al casco. Por temas prácticos, la batería es transportada por el ciclista, ya sea en un chaleco o en algún bolsillo de toda su vestimenta tomando en cuenta la longitud del cable de alimentación y que este no estorbe al ciclista para realizar maniobras o giros.

En referencia a las recomendaciones mencionadas, el prototipo general tendría la siguiente visualización. En la Figura N° 32 se aprecia en un plano frontal y en la Figura N° 33 se aprecia en un plano posterior

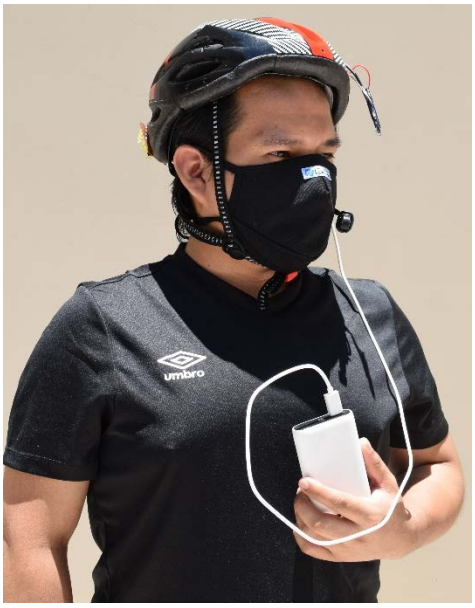

Figura Nº 32: Prototipo general implementado vista frontal

Fuente: Elaboración propia

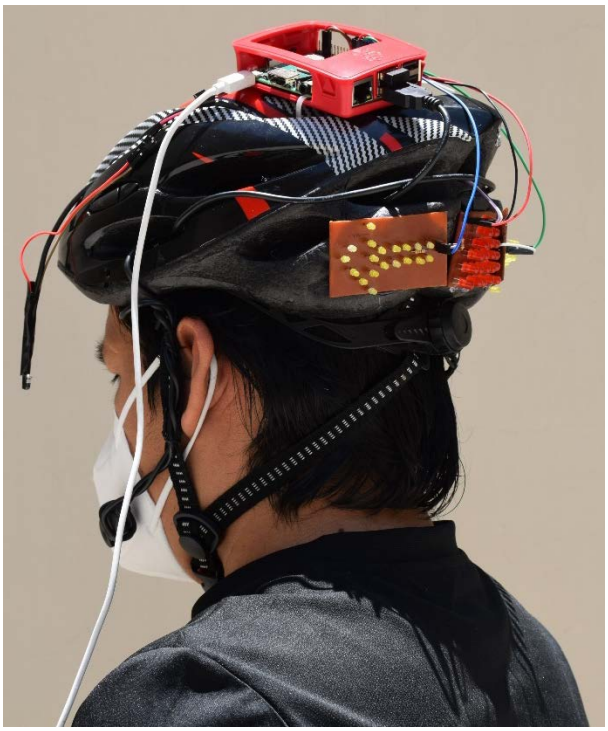

Figura Nº 33: Prototipo general implementado vista posterior

Fuente: Elaboración propia

Cabe mencionar que la mascarilla que se aprecia en las Figuras 32 y 33, se debe a la situación de pandemia que se está viviendo a causa de la enfermedad por el coronavirus.

CAPÍTULO IV: PRUEBAS Y RESULTADOS

En este capítulo se muestran las diferentes pruebas que se realizaron y los resultados obtenidos. Se consideraron dos casos para las pruebas, un caso con filtro digital y otro caso sin filtro digital. De esta forma, dentro de cada caso, se consideró el uso de dos tipos de mascarillas, una mascarilla de tela de 3 capas y otra mascarilla KN95. También, se tomó en cuenta los sujetos de prueba, que fueron 3 hombres (H_1, H_2, H_3) y 3 mujeres (M_1, M_2, M_3) de distintas edades, en ambientes libres de ruido y con ruido presente. El tiempo promedio de duración de cada prueba estuvo alrededor de los 25 minutos. Se dieron 20 comandos por cada sujeto de prueba en cada situación, obteniendo una cantidad de 480 pruebas por casa caso, obteniendo un total de 960 pruebas.

Además, se realizó una prueba simple de visibilidad de los prototipos de visualización en un ambiente diurno y otro nocturno.

4.1. Pruebas de Comandos de Voz: Sin Filtro Digital

A continuación, se observan los resultados obtenidos en las pruebas realizadas sin filtro digital, estos resultados son el punto de partida y control. Para estas pruebas se tomaron dos ambientes, uno controlado sin nada de ruido ambiental dentro de una vivienda y el otro en la vía pública con tránsito de vehículos y ruido ambiental. Cada sujeto de prueba realizó 20 comandos de voz aleatorios por cada prueba. Así mismo se evaluó a cada sujeto de prueba con dos mascarillas distintas. Cabe resaltar que estas pruebas se realizaron en días y momentos diferentes, es decir que hay pruebas que se realizaron en la mañana, en la tarde y por la noche, de acuerdo a la disponibilidad de tiempo libre de los sujetos de prueba. En la Tabla N°1 se observa el porcentaje de eficiencia que obtuvo cada sujeto de prueba en las distintas pruebas realizadas. El porcentaje de eficiencia se obtuvo dividiendo el total de comandos acertados sobre el total de comandos realizados.

Tabla N° 1: Eficiencia de pruebas de comandos de voz: sin filtro digital

	Mascarilla Tela 3 Capas						**Mascarilla KN95**					
	H_1	**H_2**	**H_3**	**M_1**	**M_2**	**M_3**	**H_1**	**H_2**	**H_3**	**M_1**	**M_2**	**M_3**
Ambiente sin ruido	80%	90%	85%	80%	75%	80%	85%	95%	85%	80%	80%	85%
Ambiente con ruido	70%	75%	70%	65%	65%	70%	75%	80%	75%	75%	70%	75%

Fuente: Elaboración propia

De igual forma, en la Tabla N° 2 observamos la cantidad total de comandos acertados y no acertados que se obtuvieron al realizar las pruebas sin filtro digital.

Tabla N° 2: Cantidad total de comandos emitidos: sin filtro digital.

		Derecha		**Izquierda**		**Alto**		**TOTAL**
		Acertados	No Acertados	Acertados	No Acertados	Acertados	No Acertados	
Masc. tela 3	Ambiente sin ruido	31	7	40	9	27	6	120
	Ambiente con ruido	26	12	34	15	23	10	120
Masc. KN95	Ambiente sin ruido	32	6	42	7	28	5	120
	Ambiente con ruido	29	10	36	12	25	8	120
TOTAL		118	35	152	43	103	29	**480**

Fuente: Elaboración propia

De estas pruebas realizadas, concluimos que hay un mayor porcentaje de eficiencia en un ambiente sin ruido, en promedio un 83.33%, comparado con un ambiente con ruido que presenta una eficiencia promedio de 72.08%. Además, con respecto a las mascarillas usadas, se obtuvo una eficiencia promedio de 80% con la KN95

mientras que con la mascarilla de tela de 3 capas se obtuvo una eficiencia promedio de 75.42%

4.2. Pruebas de Comandos de Voz: Con Filtro Digital

En estas pruebas realizadas, se repitieron las mismas condiciones anteriores, con la diferencia que se habilitó el filtro digital. Se repitieron los casos de ruido ambiental e intercambio de mascarillas con los mismos sujetos de prueba. Cabe resaltar que estas pruebas se realizaron en días distintos de acuerdo a la disponibilidad de tiempo de los sujetos de prueba. En la Tabla N° 2 se observa el porcentaje de eficiencia que obtuvo cada sujeto de prueba en las distintas pruebas realizadas. El porcentaje de eficiencia se obtuvo dividiendo el total de comandos acertados sobre el total de comandos realizados.

Tabla N° 3: Eficiencia de pruebas de comandos de voz: con filtro digital

	Mascarilla Tela 3 Capas						**Mascarilla KN95**					
	H_1	**H_2**	**H_3**	**M_1**	**M_2**	**M_3**	**H_1**	**H_2**	**H_3**	**M_1**	**M_2**	**M_3**
Ambiente sin ruido	85%	100%	90%	85%	95%	85%	90%	95%	90%	90%	95%	85%
Ambiente con ruido	90%	95%	90%	80%	85%	90%	85%	85%	90%	85%	85%	90%

Fuente: Elaboración propia

De igual forma, en la Tabla N° 4 observamos la cantidad total de comandos acertados y no acertados que se obtuvieron al realizar las pruebas con filtro digital.

Tabla N° 4: Cantidad total de comandos emitidos: con filtro digital

		Derecha		**Izquierda**		**Alto**		**TOTAL**
		Acertados	No Acertados	Acertados	No Acertados	Acertados	No Acertados	
Masc. tela 3	Ambiente sin ruido	35	4	39	5	34	3	120
	Ambiente con ruido	33	3	41	4	35	4	120
Masc. KN95	Ambiente sin ruido	36	4	39	5	31	5	120
	Ambiente con ruido	33	5	43	7	28	4	120
TOTAL		137	16	162	21	128	16	**480**

Fuente: Elaboración propia

De estas pruebas realizadas con el filtro digital, concluimos que la eficiencia promedio en un ambiente sin ruido es de 90.42%, y en el ambiente con ruido la eficiencia promedio es de 87.50%. Además, con respecto a las mascarillas usadas, se obtuvo una eficiencia promedio de 88.75% con la KN95 mientras que con la mascarilla de tela de 3 capas se obtuvo una eficiencia promedio de 89.17%

Por otro lado, tanto en este caso como en el anterior, mientras se realizaban las pruebas el sujeto de prueba conversaba y emitía frases y palabras ajenas a los comandos que reconoce el prototipo, siendo estos irrelevantes puesto que el prototipo puede llegar a traducirlos, pero luego simplemente los ignora por no pertenecer a los comandos de control.

4.3. Prueba de Visibilidad

Por motivo de la pandemia que se afronta al momento de realizar esta tesis y en su momento la cuarentena obligatoria y focalizada que debimos realizar, se desarrolló el prototipo con los escasos componentes que se pudo adquirir. Por tal motivo los prototipos de visualización no son los más óptimos y presentan una baja intensidad de emisión de luz por parte de estos, por lo que se realizó una prueba de visibilidad de la luz emitida en diferentes momentos del día. A continuación, en la Tabla N° 5 se observa lo obtenido.

Tabla N° 5: Visibilidad del prototipo de visualización

	Mañana	**Tarde**	**Noche**
Visibilidad	Nula	Nula	Moderada

Fuente: Elaboración Propia

Esta prueba se realizó teniendo una distancia promedio de 3 metros entre el ciclista y el observador, los valores mencionados en la Tabla N°4 fueron tomados a criterio de los tesistas. Y como se aprecia en la Figura N° 34 la visibilidad de los prototipos de visualización de alto (Leds rojos) es nulo por la mañana y por tarde, en la figura mencionada a la izquierda tenemos el led encendido y a la derecha tenemos los leds apagados.

Figura Nº 34: Visibilidad de led por la tarde

Fuente: Elaboración propia

Sin embargo, en la Figura N° 35 podemos observar la visibilidad de los leds del prototipo de visualización por la noche, es más notorio y se aprecia con más claridad que en los otros horarios. En la figura mencionada, a la izquierda se aprecia el prototipo de visualización de Alto. Y a la derecha de la figura, se aprecia el prototipo de visualización de Derecha

Figura Nº 35: Visibilidad de led por la noche

Fuente: Elaboración propia

4.4. Tabla de Costos

Los materiales que se utilizaron para la elaboración del presente proyecto de tesis dependieron de la disponibilidad de stock del mercado actual. A continuación, se toma en consideración el costo de los componentes mostrados en la Tabla N°6.

Tabla N° 6: Costo de componentes

Dispositivos	**Precio S/.**
Casco de ciclista	45.00
Raspberry Pi 3B	300.00
Case para Raspberry Pi 3B	25.00
Mini micrófono USB	30.00
Cable de extensión USB	5.00
Cables jumper	10.00
Led’s	10.00
Placa PCB	5.00
TOTAL	S/ 430.00

Fuente: Elaboración Propia

CONCLUSIONES

1. Se realizó el diseño del filtro digital pasabanda Butterworth de orden 6 con frecuencias de corte de 200hz y 1300hz en Matlab para poder obtener la función de transferencia indicada en la Figura N°5 del Capitulo3, y posteriormente se observa gráficamente la respuesta de frecuencia del filtro como muestra en la figura N°6 del Capítulo 3. Con la función de transferencia obtenida, se realizó un algoritmo dentro del Raspberry Pi 3B utilizando el lenguaje Python, el cual procede a filtrar la grabación de voz periódicamente y de esta forma atenuar el ruido, mejorando la eficiencia de percepción de los comandos de voz en ambientes con ruido, de un 72.08% a un 87.50% como se muestra en los resultados del Capitulo4, subtitulo 4.1 y 4.2.
2. Se diseñó toda la programación con bloques dentro de la plataforma Node-Red como se describe en el Capítulo 3, subtitulo 3.3.1 Node-Red. Al ser una plataforma nativa con el sistema operativo de Raspberry, se conectó eficientemente al servicio Speech to Text de IBM Watson, como se muestra en la Figura N°18, el cual leía los segmentos de audio grabados por el Raspberry Pi 3B y los traducía exitosamente como se pueden observar en los resultados obtenidos en el Capítulo 4, obteniendo en todos los casos de prueba un promedio de eficiencia mayor al 70%.
3. Se diseñó tres plantillas de LED en Proteus y se implementó en un PCB como se muestra en el Capítulo 3 subtitulo 3.4 Bloque de salida, las plantillas de señalización se conectaron a los pines de salida del Raspberry Pi 3B los cuales esperan a recibir la señal de excitación proveniente del Node-Red, como se puede apreciar en el Capítulo 3 subtitulo 3.4 Bloque de salida la Figura N°28 y Figura N° 32. El funcionamiento integral del Raspberry Pi 3B, micrófono y plantillas leds son alimentados con una batería de 10000 mAh haciendo factible el funcionamiento durante 3 horas.

RECOMENDACIONES

1. Se recomienda considerar el tiempo de retardo de 7 segundos aproximadamente producido por el servicio de Speech to Text, debido a que este se carga la data a la nube de IBM y luego la descarga, produciendo un tiempo muerto considerable. Así mismo, el grabar un segmento de audio para ser filtrado aumenta el retardo.
2. Evaluar la necesidad del filtro, dependiendo del rango de frecuencias que acepta el micrófono se puede verificar si es necesario un filtro pasa banda o simplemente un filtro pasa bajo.
3. Se recomienda considerar la autonomía del funcionamiento del Raspberry junto a la batería externa. Para el caso presentado se estima un tiempo de 4 horas.
4. Para el uso de un tiempo mayor a lo recomendado considerar contar con una batería externa de mayor capacidad.
5. Se recomienda considerar la intensidad luminosa de los Leds. El paso planteado solo es óptimo en la noche por la baja intensidad de luz que el prototipo emite.
6. Se debe tomar en consideración, la conectividad a internet para el uso del proyecto debido a que los datos enviados a IBM Cloud para el servicio de speech to text son en tiempo real.
7. Para una posible mejora del proyecto en portabilidad se recomienda utilizar el módulo Raspberry Zero.

REFERENCIAS BIBLIOGRÁFICAS

Thakur R. Manoj Kumar Pandey, N. (2018). Filtering of Noise in Audio/Voice Signal. Recuperado el 03 de Junio del 2020 del sitio web: https://ieeexplore.ieee.org/document/9007299

Organización Panamericana de la Salud (2018). Nuevo informe de la OMS destaca que los progresos han sido insuficientes en abordar la falta de seguridad en las vías de tránsito del mundo. Recuperado el 06 de julio del 2020, del sitio web: https://www.paho.org/per/index.php?option=com_content&view=article&id=4210:nuevo-informe-de-la-oms-destaca-que-los-progresos-han-sido-insuficientes-en-abordar-la-falta-de-seguridad-en-las-vias-de-transito-del-mundo&Itemid=1062

Ruiz, R. (2015). Sistema de señalización para ciclistas. Recuperado el 10 de julio del 2020, del sitio web: https://e-archivo.uc3m.es/bitstream/handle/10016/25108/PFC_Raquel_Ruiz_Ca%C3%B1amero_2015.pdf

Tomala, D. (2018). Sistema domótico controlado por voz para personas con discapacidades superiores, utilizando tarjeta Raspberry Pi. Recuperado el 12 de julio del 2020, del sitio web: https://dspace.ups.edu.ec/bitstream/123456789/15141/1/UPS-GT002060.pdf

Gil, J. Castillo, L. Flórez, R. (2016). Spanish speech recognition oriented to a wheelchair control. Recuperado el 12 de julio del 2020, del sitio web: https://www.academia.edu/37809832/Spanish_speech_recognition_oriented_to_a_wheelchair_control

Ley Nª 30936 (2019). Ley que promueve y regula el uso de la bicicleta como medio de transporte sostenible. Recuperado el 8 de julio del 2020, del sitio web: https://busquedas.elperuano.pe/normaslegales/ley-que-promueve-y-regula-el-uso-de-la-bicicleta-como-medio-ley-n-30936-1762977-4/

IBM (2020). IBM Cloud. Recuperado el 15 de julio del 2020, del sitio web: https://cloud.ibm.com/docs/overview?topic=overview-whatis-platform&locale=es

IBM (2020). Watson Speech To Text. Recuperado el 15 de julio del 2020, del sitio web: https://www.ibm.com/ar-es/cloud/watson-speech-to-text

OpenJS Foundation (2020). Node-RED About. Recuperado el 15 de julio del 2020, del sitio web: https://nodered.org/about/

IBM (2020). ¿Qué es IBM Cloud?. Recuperado el 15 de julio del 2020, del sitio web: https://www.ibm.com/pe-es/cloud?lnk=mpr_bucl_pees&lnk2=learn

Raspberry Pi Foundation (2020). Raspberry Pi 3 Model B. Recuperado el 15 de julio del 2020, del sitio web: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/?resellerType=home

Moore, H. (2007). Matlab para Ingenieros, Utah, USA: Pearson Educación Recuperado el 18 de julio del 2020

Meiniar W. Ayu, F. Irmasari, A. Astharini, D. (2017). Human Voice Filtering with Band-Stop Filter Design in MATLAB. Recuperado el 4 de agosto del 2020, del sitio web: https://ieeexplore.ieee.org/document/8272563

Carral, M. (2014). La tecnología LED. Recuperado el 18 de julio del 2020

MCI Electronics (2019). ¿Qué es Raspberry Pi?. Recuperado el 20 de julio del 2020, del sitio web: https://raspberrypi.cl/que-es-raspberry/

Oppenheim A. Schafer R. Buck J.(1999). Discrete time signal processing, pag 439, segunda edición, editorial Prentice Hall. Recuperado el 02 de agosto del 2020

Anoja, H. (2017). Internet of Things using Node-Red and Alexa. Recuperado el 15 de agosto del 2020 del sitio web: https://ieeexplore.ieee.org/document/8261194