SISTEMA DE IDENTIFICACIÓN DE LOCUTOR TEXTO DEPENDIENTE EN RASPBERRY PI 3 B CON APLICACIÓN EN CONTROL DE ACCESO

RAUL ALEJANDRO BLANCO ORTIZ
CÓD. 20171383006

NELSON JAVIER GARZÓN GAMEZ
CÓD. 20171383008

TESIS DE INGENIERÍA EN CONTROL

UNIVERSIDAD DISTRITAL "FRANCISCO JOSÉ DE CALDAS"
FACULTAD TECNOLÓGICA

INGENIERÍA EN CONTROL

Bogotá, agosto de 2019

SISTEMA DE IDENTIFICACIÓN DE LOCUTOR TEXTO DEPENDIENTE EN RASPBERRY PI 3 B CON APLICACIÓN EN CONTROL DE ACCESO

RAUL ALEJANDRO BLANCO ORTIZ

NELSON JAVIER GARZÓN GAMEZ

Tesis presentada al programa de Ingeniería en Control de la Universidad

Distrital "Francisco José De Caldas" Facultad Tecnológica, para optar por el título de Ingeniero en Control

Programa:

Ingeniería en Control

Director del Proyecto

Ing. MIGUEL RICARDO PEREZ PEREIRA

Evaluador del Proyecto

Ing. HOLMAN MONTIEL ARIZA

Bogotá, agosto de 2019

NOTA DE ACEPTACIÓN

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Jurado 1

Jurado 2

RESUMEN

En este proyecto se desarrolla un sistema de reconocimiento de locutor texto dependiente con base a los coeficientes cepstrales de la voz utilizando una Raspberry Pi 3 B, en conjunto con una tarjeta de audio USB y un micrófono con conexión tipo jack de 3.5mm.

En este dispositivo embebido se implementa la captura de la señal de audio, el acondicionamiento de la señal, la extracción de las características de la voz, la técnica de reconocimiento de locutor, el manejo de la base de datos para cada los usuarios del sistema, el registro de acceso de los usuarios y la implementación de la interfaz.

Todo el desarrollo del sistema se realiza en software libre utilizando el lenguaje de programación Python en la versión 3.5, utilizando la librería pyaudio para la captura de la señal, librerías matemáticas como numpy, scipy, wave para el acondicionamiento de la señal, para la extracción de los parámetros cepstrales de la voz se utiliza la librería speech_features, la técnica de reconocimiento implementada es DTW y la base de datos del sistema es implementada en MySQL.

Al final, se realizan pruebas para identificar falsas aceptaciones y falsos rechazos y así, determinar el porcentaje de error del sistema, su confiabilidad y eficiencia.

Palabras Clave: Reconocimiento de locutor, parámetros cepstrales, Raspberry, base de datos.

ABSTRACT

In this project, a recognition system of dependent text speaker using cepstral coefficients of the voice is developed using a Raspberry Pi 3 B with a USB audio card and a microphone with a 3.5mm jack connection. In this embedded device is implemented the capture of the audio signal, the signal conditioning, the extraction of the characteristics of the voice, the speaker recognition technique, the management of the database for each user of the system, the access registry of the users and the implementation of the interface. All system development is done in free software using the Python programming language in version 3.5, using the pyaudio library to capture the signal, mathematical libraries such as numpy, scipy, wave for signal conditioning, for extraction of the speech cepstral parameters the speech_features library is used, the recognition technique implemented is DTW and the database of the system is implemented in MySQL.

In the end, tests are performed to identify false acceptances and false rejections and thus, determine the percentage of system error, its reliability and efficiency.

Keywords: speaker recognition, cepstral parameters, Raspberry, database.

Tabla de Contenido

pág.

LISTA DE FIGURAS

pág.

LISTA DE TABLAS

pág.

LISTA DE ANEXOS

pág.

1. INTRODUCCIÓN

En el amplio espectro de la ingeniería en control, el procesamiento de señales es una de las ramas más importantes, pues se encarga del acondicionamiento y el tratamiento de la señal para obtener información precisa y exacta que luego es utilizada para la toma de decisiones en un sistema.

Los sistemas de seguridad biométricos son un claro ejemplo de la importancia del procesamiento de señales, porque el funcionamiento de estos sistemas se basa en la capacidad de no equivocarse en el reconocimiento de los usuarios. Estos sistemas juegan un papel fundamental en el control de acceso; garantizan un medio para la supervisión de la asistencia, el cumplimiento de los horarios de llegada y de salida de los empleados en una empresa y permiten restringir el ingreso a determinadas áreas en una compañía.

Existen varios sistemas de seguridad biométricos, los cuales se diferencian entre sí de acuerdo a el tipo de información utilizada y la técnica aplicada para reconocer al usuario. Los sistemas biométricos más antiguos están basados en la firma y la huella dactilar, sin embargo, el desarrollo de la tecnología ha permitido incrementar los tipos de sistemas biométricos, ahora existen sistemas basados en el reconocimiento de la voz, el reconocimiento del iris, el reconocimiento de retina, el reconocimiento de la palma, entre otros.

Los sistemas biométricos basados en el reconocimiento de voz tienen algunas ventajas frente a los otros, por ejemplo, son un 50% más fáciles de usar que un reconocimiento de retina y un 25% que un reconocimiento de iris [1]; ya que es más natural y eficiente para el ser humano la acción de hablar que ubicarse al frente de un escáner de iris o de retina durante un determinado tiempo. Además, la aceptación de los sistemas basados en reconocimiento de voz es 25% más alta que los sistemas de reconocimiento de retina, de iris y de huella dactilar debido a su fácil implementación y bajo costo de desarrollo [2].

Estos sistemas se componen de dos fases, la fase de entrenamiento y la fase reconocimiento, además se clasifican en dos tipos, los de reconocimiento de locutor texto dependiente y los de reconocimiento de locutor texto independiente. Los sistemas de reconocimiento de locutor texto dependiente consisten en

reconocer al hablante con base a una palabra o frase única que identifica a cada hablante y solo lo reconoce con esa palabra o frase, mientras que los sistemas de reconocimiento de locutor texto independiente tienen la capacidad de reconocer al hablante sin importar la palabra o frase.

A lo largo de este documento se describe el desarrollo actual de los sistemas de reconocimiento de locutor texto dependiente, las técnicas más utilizadas y luego, el desarrollo de este proyecto.

2. PLANTEAMIENTO DEL PROBLEMA

El desarrollo exponencial de la tecnología ha permitido incrementar los niveles de control y seguridad en todo el mundo por medio de dispositivos biométricos, como escáneres de retina y lectores de huella; estas implementaciones se utilizan principalmente en la industria para llevar un registro del acceso de cada uno de los trabajadores y el control del cumplimiento de su jornada laboral. Estos sistemas se encuentran en el mercado a un alto costo, por ejemplo, Cucorent[3] es una empresa ubicada en España dedicada al desarrollo y comercialización de sistemas biométricos; esta compañía ofrece sistemas destinados al control de acceso basados en reconocimiento de huella o reconocimiento facial que van hasta los 1.124 euros.

Además de su elevado costo, los sistemas biométricos basados en el reconocimiento de la huella dactilar y el reconocimiento facial, e incluso los sistemas más económicos como aquellos basados en contraseñas, carecen de una gran ventaja que poseen los sistemas de reconocimiento por voz y es la velocidad. Unisys [4] es una empresa dedicada a la seguridad que realizó un estudio para identificar el sistema de seguridad biométrico preferido por los consumidores; identificó que los sistemas basados en el reconocimiento de la voz tienen la mayor aceptación del mercado con un 35%, seguido por el reconocimiento de huella dactilar con un 27% y el reconocimiento facial con un 20%, completan la lista el reconocimiento basado en la palma de la mano y el basado en el escaneo de iris con un 12% y 10% respectivamente. El estudio realizado por Unisys reveló que la velocidad y la agilidad fueron los factores diferenciales a favor del sistema de reconocimiento basado en voz.

En la actualidad, los sistemas de reconocimiento de voz se han convertido en algo habitual de nuestra vida cotidiana con herramientas como Google Assistant y Siri presentes en nuestros smartphones; facilitan la posibilidad de desempeñar la multitarea y ofrecen una interacción hombre-máquina más natural y más rápida; teniendo en cuenta las ventajas que ofrecen estos sistemas, ¿Es posible desarrollar un sistema de control de acceso basado en el reconocimiento de voz utilizando software libre y la Raspberry Pi 3 B?

3. OBJETIVOS

3.1 Objetivo general

Diseñar e implementar un sistema de identificación de locutor texto dependiente en Raspberry Pi 3 B destinado al control de acceso.

3.2 Objetivos específicos

- Diseñar e implementar una interfaz de usuario en la Raspberry Pi 3 B.

- Diseñar el algoritmo que reconozca la voz de locutor con base a los parámetros cepstrales de la voz.

- Generar una base de datos que contenga los usuarios pertenecientes al sistema y que permita llevar un registro de acceso de los mismos.

4. ESTADO DEL ARTE

El desarrollo exponencial de la tecnología durante los últimos años ha permitido la mejora de los sistemas de seguridad y control de acceso como lo son los sistemas de reconocimiento de voz, los cuales procesan y analizan señales para determinar e identificar los patrones de voz de un individuo.

De acuerdo a la forma de verificación existen dos categorías en los sistemas de reconocimiento de voz, locutor texto-dependiente y locutor texto-independiente; en la primera categoría el sistema conoce un texto hablado por la persona, este texto puede ser una palabra o una frase y sirve para entrenar al sistema. Esta verificación es muy utilizada para aplicaciones con un alto control sobre la entrada del usuario, las ventajas de este tipo de reconocimiento es que posee una base de información pequeña y tiene una buena tasa de reconocimiento [5]. La verificación texto-independiente acepta cualquier entrada, es decir, está diseñado para reconocer al usuario sin importar la palabra o frase que diga; este método requiere más entrenamiento y base de información porque es más sensible a la calidad acústica de la entrada [5].Las áreas de implementación del reconocimiento de voz son muchas; por ejemplo los call-center, los servicios de bancos, la industria militar, el cuidado de la salud, el comercio electrónico, el entretenimiento, los servicios del gobierno, los sistemas de control de accesos, la industria del transporte y la tecnología de la información[5].

En la categoría de reconocimiento de voz locutor texto-dependiente se han desarrollado trabajos enfocados al aumento de la eficiencia y la confiabilidad del sistema por medio de la combinación de diferentes técnicas ampliamente utilizadas en este campo como lo son los modelos de mezcla Gaussiana (GMMs) [6][7][8][9], componentes cepstrales de la frecuencia de mel (MFCC) [6][7][8][9][10], coeficientes de auto regresión (AR) [11], medida de distancia basada en Hausdorff (HDM) [12] y los modelos ocultos de Markov (HMM) [13][14]. Estas técnicas tienen como objetivo principal extraer los parámetros característicos de la señal de voz para poder reconocer al locutor, esta información es almacenada para posteriormente ser comparada con una entrada en vivo y determinar si existe o no una coincidencia y así, de forma general, completar el proceso de reconocimiento del locutor; luego, viene la aplicación o implementación en el área que se desee.

Por ejemplo, el arranque automático e inteligente de un motor basado en un sistema de reconocimiento de voz [12] es una de las muchas aplicaciones que se le puede dar a este sistema biométrico. En este proyecto [12], buscan reemplazar la funcionalidad de la llave de un carro por el comando de voz que permita identificar las palabras "encendido" y "apagado" con el propósito de iniciar o apagar el motor. Con este fin, realizan el sistema de la Figura 1.

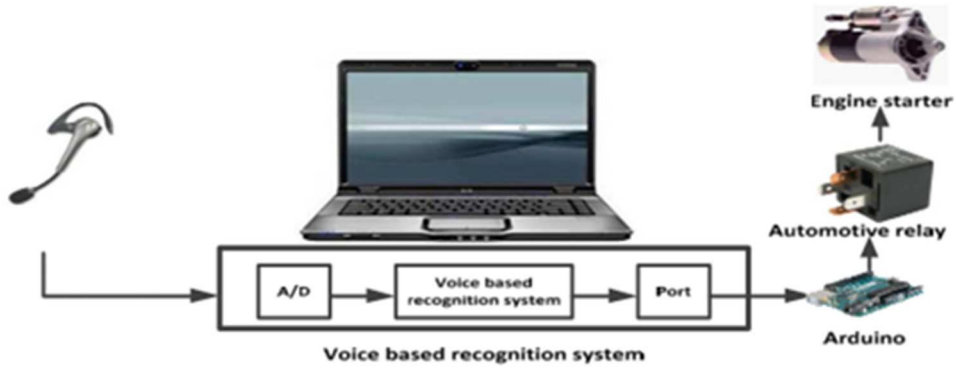

Figura 1. Propuesta del arranque automático e inteligente para un motor basado en la voz [12]

El sistema propuesto en la figura 1 ofrece un rendimiento biométrico con la habilidad de realizar una identificación y verificación positiva de las características de la voz del individuo autorizado para acceder al mecanismo de arranque del motor [12]. El sistema es compuesto de cinco partes, sensor de grabación, sistema de reconocimiento basado en voz, arduino, relé y arrancador de motor. Un micrófono de bajo costo para computador es usado para capturar y grabar la voz de la persona autorizada, esta señal es procesada por el sistema de reconocimiento de voz el cual, reconocerá si la palabra es encendido o apagado y este reconocimiento es usado como entrada en el arduino que activará el relé y este, a su vez, activará en arranque del motor [12].

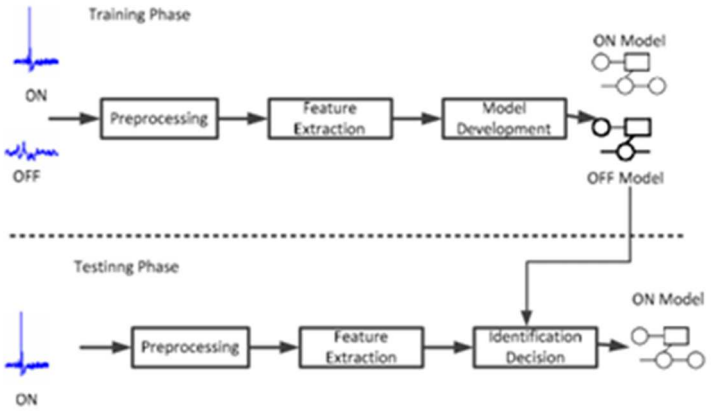

Figura 2. Sistema de reconocimiento de voz [12]

La figura 2 muestra el sistema general propuesto para el reconocimiento de voz. La señal digital de la voz es pasada por el pre procesamiento donde se normaliza la señal para hacerla menos susceptible a los efectos de las siguientes etapas. Una vez normalizada la señal, pasa al bloque de extracción de características donde se convierte la señal de entrada en un vector de características de la señal el cual es usado como entrada del bloque de clasificación del sistema donde, se determina si la palabra fue encendido o apagado [12].

Hay dos fases muy importantes en el sistema propuesto, la primera fase es la de entrenamiento y la segunda fase es la de prueba. En la fase de entrenamiento, el sistema es entrenado para desarrollar un modelo base de la palabra de la persona autorizada, una plantilla para ese patrón de voz es almacenado en la memoria; para esto, son almacenadas diez palabras de encendido y diez palabras de apagado. En la fase de prueba, la información de entrada debe coincidir con el modelo almacenado con el propósito de reconocer la palabra [12].

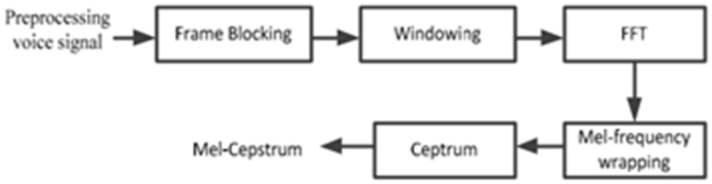

Figura 3. Diagrama de bloques del método de extracción MFCC [8]

El bloque de extracción de características de la figura 2 es la sección más importante del sistema de reconocimiento de voz. La figura 3, muestra el procedimiento y la técnica utilizada en la extracción de características de la señal; la técnica utilizada es MFCC que provee de un buen desempeño y una sencilla implementación [12].En este caso, utilizaron el concepto de las máquinas de vectores de soporte (SVM) como herramienta de clasificación de la información almacenada y optimización en el proceso de toma de decisión ya que, este método permite evitar caer en redundancias durante la evaluación de los parámetros de la señal de entrada respecto a la información almacenada [12].

Otro ejemplo de aplicación en esta área es el reconocimiento de locutor texto dependiente para vietnamitas [5]. Este proyecto busca la optimización del reconocimiento del locutor por medio de la combinación de dos técnicas, una es GMM y la otra HMM. Para cumplir con el propósito desarrollaron el modelo general para el sistema de reconocimiento del locutor presentado en la figura 4.

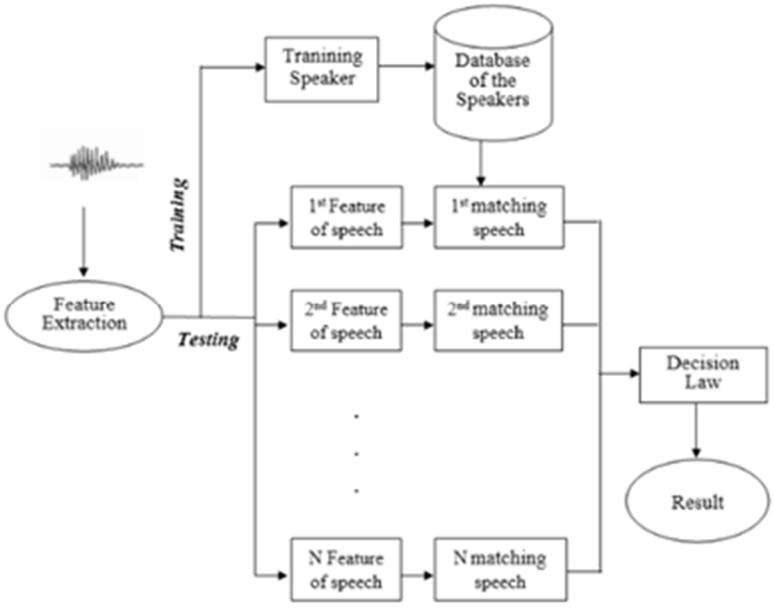

Figura 4. Modelo general del sistema de reconocimiento de locutor texto dependiente [7].

En el modelo de la figura 4 se diferencian claramente dos etapas del sistema; una de entrenamiento y una de prueba. Durante la fase de prueba desarrollaron las siguientes operaciones:

- Grabaron las voces de los locutores, donde cada locutor dice una secuencia de palabras prestablecidas. Cuantas más veces el locutor repita esta secuencia, más alta es la tasa de reconocimiento del sistema.
- Extracción de características, como lo son los coeficientes cepstrales, MFCC, predicción lineal de los coeficientes cepstrales (LPCC).
- Modelo del locutor, donde el objetivo es darle a cada locutor un único patrón. Para esto están las técnicas de GMM, HMM, redes neronales, entre otras.
- Almacenar las características de cada locutor, estas características se dividen en dos partes; las características propias del locutor y las características de cada palabra para cada locutor

Cada locutor es representado por un modelo GMM, la figura 5 describe en entrenamiento de la información para cada locutor.

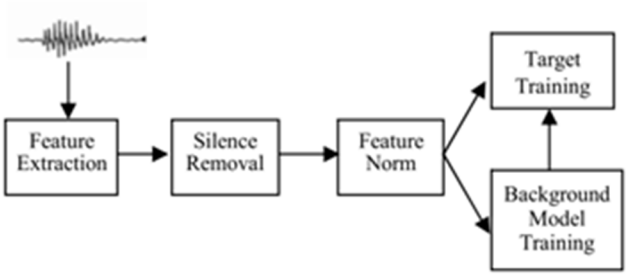

Figura 5. Modelo GMM de entrenamiento para cada locutor [7].

El modelo de la figura 5 describe los siguientes pasos para obtener el modelo GMM de cada locutor:

- Extracción de características, donde la técnica utilizada es MFCC y las características son almacenadas en vectores.
- Eliminación del silencio, donde se identifica las partes de la grabación que corresponden al silencio y se eliminan. Para esto se usó la técnica de distribución logarítmica de la energía para cada segmento de la grabación.

- Normalización de las características, donde se normalizan los vectores que contienen las características de la señal de cero a uno.
- Formación del modelo de fondo, donde las muestras de entrenamiento son transformadas en modelos generales.
- Formación del modelo del objetivo, donde, a partir de los parámetros de la etapa anterior, se obtiene el modelo de cada locutor.

Además, se genera el modelo para cada palabra de cada locutor y para esto utilizando modelo de la figura 6.

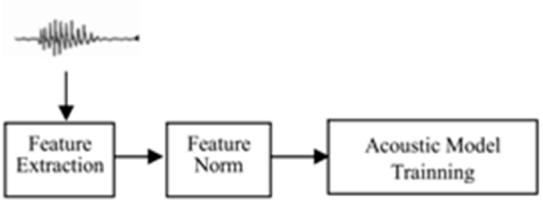

Figura 6. Modelo HMM de entrenamiento para cada palabra de cada locutor [7].

En este proyecto, la clave de la información de cada locutor está representada en el modelo HMM para cada palabra de cada locutor. Este es un modelo muy flexible que con un pequeño número de fonemas se puede crear la información de muchas palabras clave. Este proceso se describe en la figura 6 y se compone de las siguientes etapas:

- Extracción de características, que corresponde a la información obtenida con la técnica MFCC
- Normalización de las características, donde se normalizan los vectores que contienen las características de la señal acorde a la longitud del generador de tono.
- Formación del modelo fonético, donde el algoritmo de maximización de la expectación (EM) es utilizado para obtener el modelo de los fonemas.

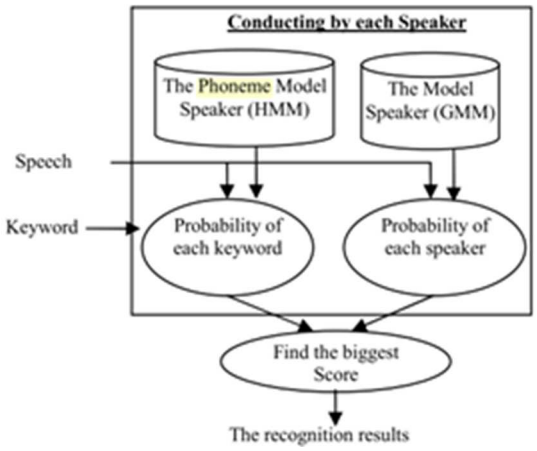

Figura 7. Proceso de reconocimiento de locutor [7]

Entonces, la figura 7 representa el proceso implementado en el reconocimiento de locutor texto dependiente para vietnamitas, donde el sistema reconocerá de forma independiente al locutor y la palabra clave y, al encontrar la mayor probabilidad en cada caso arrojará quien habló y cual palabra pronunció [7].

5. MARCO TEORICO

5.1 SOFTWARE LIBRE

Es el software que respeta la libertad de los usuarios y la comunidad, significa que los usuarios tienen la libertad de ejecutar, copiar, distribuir, estudiar, modificar y mejorar el software. Es decir, el software libre es una cuestión de libertad, no de precio. Un programa es software libre si los usuarios tienen las cuatro libertades esenciales [15]:

- La libertad de ejecutar el programa como se desee, con cualquier propósito.
- La libertad de estudiar cómo funciona el programa, y cambiarlo para que haga lo que usted quiera. El acceso al código fuente es una condición necesaria para ello.
- La libertad de redistribuir copias para ayudar a otros.
- La libertad de distribuir copias de sus versiones modificadas a terceros. Esto le permite ofrecer a toda la comunidad la oportunidad de beneficiarse de las modificaciones. El acceso al código fuente es una condición necesaria para ello.

Los sistemas operativos basados en Linux hacen parte del denominado software libre como es el caso de Raspbian, el cual, es un sistema operativo libre basado en Debian (una distribución de linux) optimizado para el hardware de la Raspberry Pi que contiene alrededor de 35.000 paquetes y software pre-compilado que facilita su instalación [16].

5.1.1 PYTHON

Es un lenguaje de programación que no necesita compilador, es de alto nivel y solo necesita un intérprete, por lo tanto, cuando se ejecutan las líneas de código se hace directamente y sin necesidad de generar ejecutables. Otra característica importante es que Python es orientado a objetos, cuenta con elementos como [17]:

- Clase: Modelo sobre el cual se crean y estructuran los objetos.
- Propiedad: Características de un objeto que se manejan como variables.

- Método: Función que contiene acciones que deben realizar los objetos, su sintaxis es Def nombre método ().
- Objeto: Cualquier sustantivo cuyas características puedan describirse como cualidades o atributos. Algunos tipos de objetos son: String, entero, float, char, etc. Algunos de estos objetos pueden caracterizarse nombrando otros objetos. Cuando se programa en Python no es necesario especificar el tipo del objeto, solo se le asigna un nombre y valor especifico.
- Herencia: Una clase puede heredar a otra si los objetos comparten propiedades y/o métodos.

5.2 LA VOZ

La voz es producida por los órganos fonéticos. El sistema de generación de la voz está compuesto de:

- El aparato respiratorio; que es la parte del sistema que determina la intensidad, la fuerza, el poder y la duración del sonido. Este aparato se divide en dos partes, las cuales son [18]:
- El tracto superior, conformado por la cavidad nasal, la faringe y la forma de la cavidad nasal. Esta es la primera parte de la trayectoria del aire, penetrando a través de la nariz.
- El tracto inferior, conformado por la laringe, la tráquea, el tubo bronquial y los pulmones. En los pulmones se encuentran los alvéolos y es allí donde se produce el efecto de la respiración.
- El órgano de la vibración vocal, es el mecanismo de generación del sonido; el tono está dado por las vibraciones de las cuerdas vocales y está compuesto por la laringe, las cuerdas vocales y los ventrículos [19].
- La laringe es el órgano donde el sonido es generado y es convertido por la membrana mucosa, proveniente de las glándulas de segregación. En la región media de la laringe, existe una parte llamada glotis, conformada por las cuerdas vocales; estas son dos bandas móviles, juntas en su parte anterior, dejando un espacio de forma triangular para la glotis junto con los músculos de tensión y constricción, respectivamente [12][13].

- El sistema de resonancia produce el timbre, el color y los componentes armónicos de la voz. Permite localizar la voz y darle un rango de sonido. Está conformado por los resonadores y la resonancia de las cavidades. Se puede dividir en dos partes [19]:
- La parte fija y dura, conformada por los huesos como el maxilar superior, los huesos nasales, las cavidades, la bóveda platina ósea y los dientes. Estas partes son rígidas, fijas y duras. Con el objetivo de favorecer la resonancia, estas partes deben ser planas.
- La parte móvil y suave, conformada por los músculos y membranas de la faringe, el paladar, la lengua, las mejillas, los labios y existe un hueso móvil, el maxilar inferior.

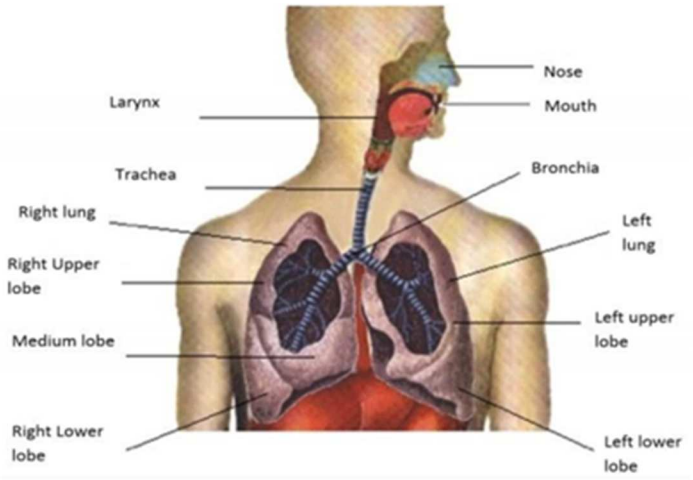

Figura 8. Aparato fonético humano [19]

La vibración de los pliegues vocales permite modelar la fuente de sonido de la voz como la velocidad del volumen en la glotis ($U_G(j\Omega)$). Las gráficas de la figura 9 representan una señal de voz en el tiempo, la velocidad del volumen en el tiempo de esa señal de voz y su magnitud expresada en frecuencia [20].

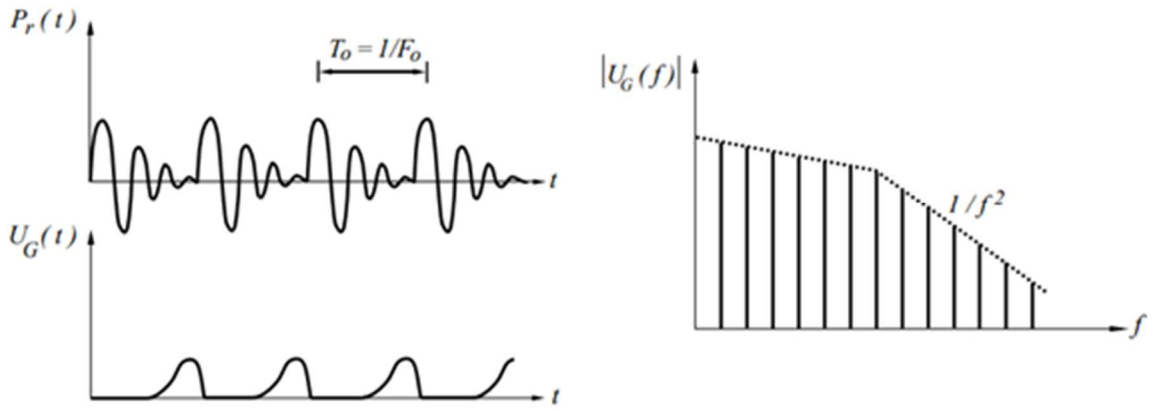

Figura 9. Señal de voz en el tiempo, velocidad del volumen y magnitud [20].

La tabla 1 representa la frecuencia fundamental de la voz según el sexo, su valor típico, su valor mínimo y su valor máximo.

	F_0 ave (Hz)	F_0 min (Hz)	F_0 max (Hz)
Men	125	80	200
Women	225	150	350
Children	300	200	500

Tabla 1. Frecuencias fundamentales de la voz según el sexo [20]

5.3 REPRESENTACIÓN DE LA SEÑAL DE VOZ

Los sonidos consisten en variaciones en la presión de aire a través del tiempo y a frecuencias que podemos escuchar. Una de las maneras de representar el sonido es por medio de una onda (waveform), como se puede ver en la figura 10 [21].

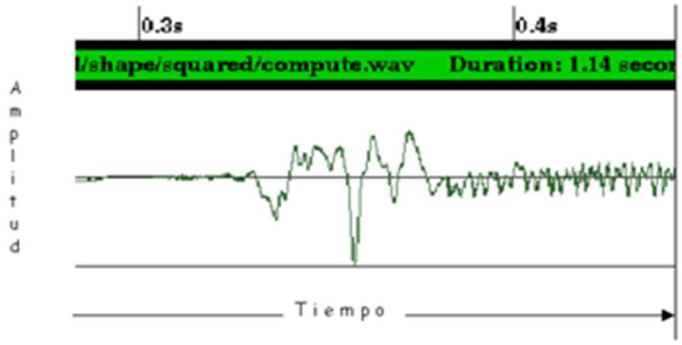

Figura 10. Representación del sonido en forma de onda [21]

Una de las grandes ventajas de este tipo de representación es que no ocupa mucho espacio de memoria. Y una desventaja es no describe explícitamente el contenido de la señal en términos de sus propiedades [21]. Los espectrogramas contienen mayor información sobre los datos de la voz, son una transformación que muestran la distribución de los componentes de frecuencia de la señal como se muestra en la figura 11 [21].

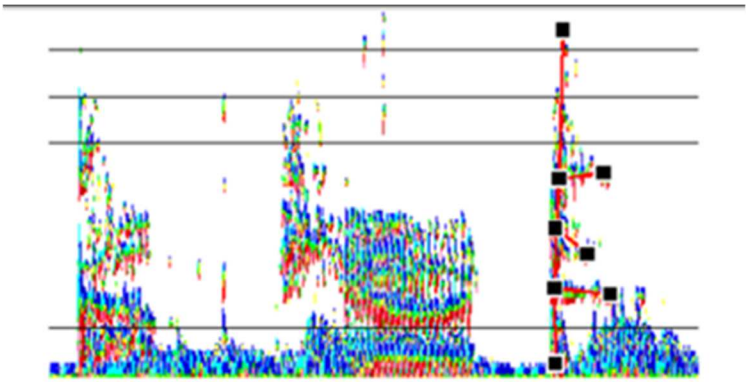

Figura 11. Espectrograma como representación del sonido [21].

Las partes más oscuras de la figura 11, representan la concentración de energía y sus denominadas formantes [21].

5.4 SISTEMA DE RECONOCIMIENTO DE LOCUTOR

De forma general, los sistemas de reconocimiento de voz tienen la estructura mostrada en la figura 12[22].

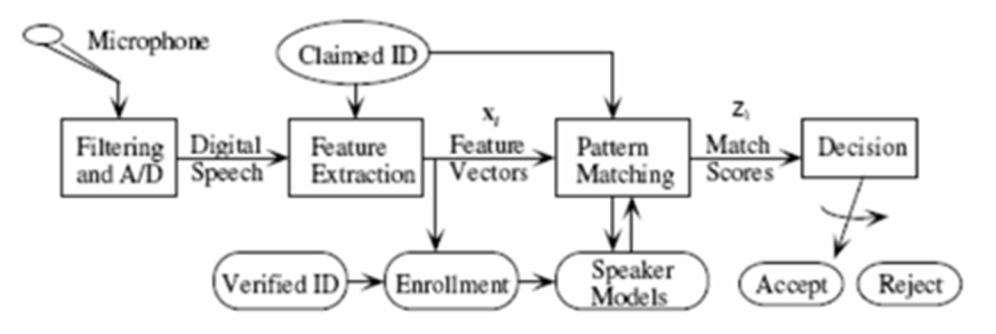

Figura 12. Sistema genérico de verificación automática de locutor [22]

Para realizar el modelo presentado en la figura 12, los sistemas de reconocimiento de locutor se basan en dos fases; una primera fase de entrenamiento y una segunda fase de verificación [22]:

- Fase de entrenamiento, que es la fase correspondiente a la creación de la base de datos donde se almacenan los modelos correspondientes a cada locutor a partir de una serie de palabras pronunciadas por los futuros usuarios del sistema [20][21].
- Fase de verificación, donde una vez obtenidos los modelos para cada uno de los locutores se comparan las locuciones a verificar con los modelos prestablecidos y se obtiene una puntuación acorde a su nivel de coincidencia. A partir de esta puntuación y con base a un umbral determinado, se toma la decisión de aceptar o rechazar la locución actual [20].

  En ambas fases, el sistema de reconocimiento de locutor se compone de las siguientes partes: adquisición de la voz, extracción de parámetros, clasificador y decisión [19][20].
- Adquisición de la voz, donde por medio de un micrófono se convierte la onda acústica (la voz) en una señal análoga. A esta señal se le aplica un filtro antialiasing para limitar el ancho de banda de la señal a la frecuencia de Nyquist y entonces, la señal se muestrea para convertirla en una señal digital con un conversor análogo/digital. En aplicaciones locales de verificación de locutor, el canal analógico es simplemente el micrófono, el cable y el acondicionamiento de la señal; debido a que la señal digital puede llegar a ser alta calidad al no tener las distorsiones que se producen en las líneas telefónicas [20].
- Extracción de parámetros, donde a través de una o más técnicas de caracterización de la voz se obtienen los parámetros que, posteriormente, permiten identificar al locutor.
- Clasificador, donde se crea un modelo probabilístico con base a los coeficientes MFCC y se almacenan en una base de datos para luego poder acceder a estos datos y comparar la señal actual con las almacenadas y dar un porcentaje de coincidencia [19].
- Decisión, donde con base en el porcentaje de coincidencia entre la señal de entrada y los datos almacenados y un umbral, predeterminado en el sistema o configurado por usuario, se determina quien hablo o si no coincide con ninguno de los datos almacenados [20].

5.5 COEFICIENTES CEPSTRALES DE LA VOZ (MFCC)

Una de las técnicas más utilizadas para la extracción de parámetros de la voz es el análisis cepstral, el cual se define en (1) como la transformada inversa de Fourier del logaritmo del valor absoluto del espectro de una señal [22]

$$c[n] = F^{-1}\{log|F\{x[n]\}|\} \quad (1)$$

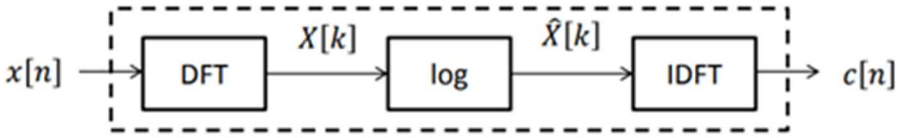

Figura 13. Modelo del análisis cepstral[22].

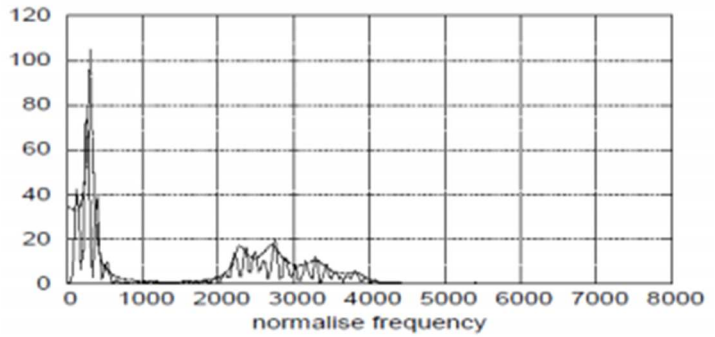

Figura 14. DFT de la señal discreta de la voz x[n] (x[k])[22]

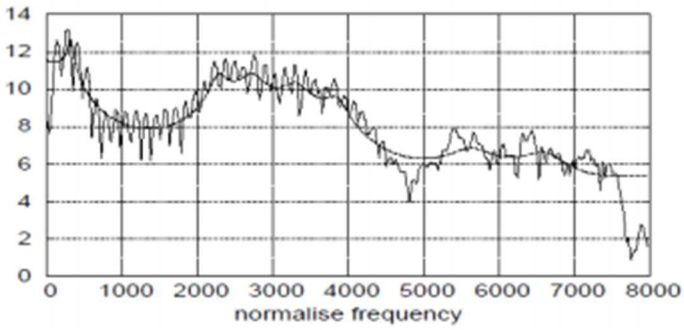

Figura 15. Logaritmo de la magnitud de la DFT de x[n] (ẋ[k])[22]

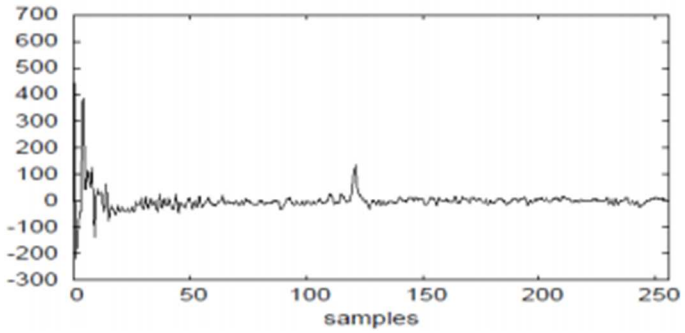

Figura 16. DFT inversa del logaritmo de la magnitud de la DFT de x[n] (c[n])[22]

Las figuras 14, 15 y 16 son la representación de la señal x[n] durante las diferentes etapas del modelo de la figura 13; la figura 14 corresponde a la salida del primer bloque, donde se aplica la DFT a x[n]; la figura 15 corresponde a la salida del segundo bloque, donde se aplica la función logaritmo a la información resultante del bloque 1 y la figura 16 corresponde a la salida del tercer bloque, donde se aplica la DFT inversa a la información resultante del bloque 2 y se obtiene c[n].

Sin embargo, este análisis por sí sólo no representa una utilidad para el reconocimiento del locutor; por tanto, se emplea la técnica de coeficientes cepstrales en la escala de frecuencia Mel o MFCC que permite representar la amplitud del espectro de manera compacta [24]. Estos coeficientes están basados en la percepción auditiva humana, sus bandas de frecuencia están ubicadas logarítmicamente, lo que modela la respuesta auditiva humana más apropiadamente que las bandas espaciadas linealmente [25]. Esto permite un procesado de datos más eficiente, por ejemplo, en la compresión de audio [20]. La figura 17 representa el procesado de la señal que se realiza en un sistema típico para computar los coeficientes MFCC.

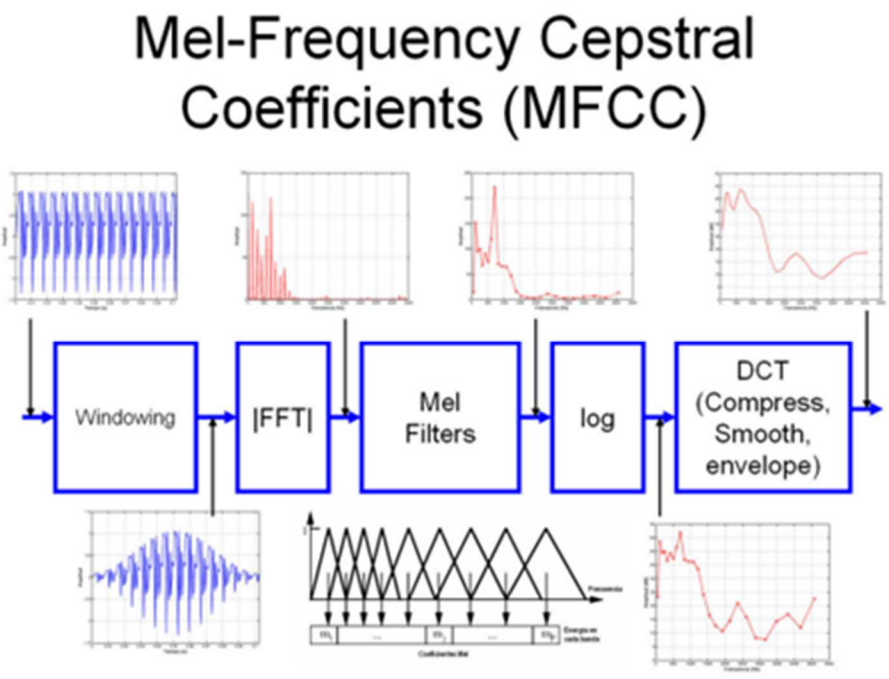

Figura 17. Diagrama de bloques del proceso de cálculo de los MFCC [20]

La señal acústica, muestreada a 8 KHz en el caso de las señales telefónicas, se diferencia (filtro de preénfasis) y se divide en un número de segmentos solapados (ventanas) cada uno de 25 ms de longitud solapados 15 ms entre sí [20]. A continuación, la señal se filtra mediante un banco de filtros de diferentes frecuencias y amplitudes para dar más resolución en las bajas frecuencias, como ocurre en el sistema auditivo humano [20]. Este filtrado se realiza en el dominio de la frecuencia al que se pasa calculando previamente la FFT. De la salida de cada filtro se calcula la energía en promedio (par la ventana de 25 ms) y los valores obtenidos se pueden ver como una nueva señal de tiempo discreto [20]. Así, por ejemplo, usando un banco de 40 filtros se obtiene, para cada trama de voz de 25 ms, un vector de 40 coeficientes. Al transformar esta señal a través de una DCT (Transformada de coseno discreta), se obtienen unos parámetros (de los que se toman habitualmente de 13 a 20) aproximadamente incorrelados entre ellos: estos son los coeficientes MFCC [20]. En particular, el primer coeficiente, C_0 representa la energía de la señal y se usa o no dependiendo de la aplicación (en caso de usarlo habitualmente se normaliza para compensar variaciones de energía debidas a la proximidad del micrófono u otros efectos indeseados) [20]. Aparte de

estos primeros coeficientes, se suelen usar también las velocidades y/o aceleraciones, que representan la evolución temporal de los fonemas al pasar de unos a otros (Delta-MFCC y Delta-Delta-MFCC) [20]. Los coeficientes Delta representan la variación de los coeficientes MFCC alrededor del instante de tiempo considerado. Suelen, por esto, llamarse coeficientes de primera derivada o velocidad. De modo similar los Delta-Delta se denominan aceleración [20].

5.6 TECNICA DE RECONOCIEMTO DTW

El Alineamiento Temporal Dinámico es un método empleado en reconocimiento de locutor texto dependiente [16]. Esta técnica trata de compensar la variabilidad que existe entre las duraciones de los fonemas en las distintas realizaciones o pronunciaciones de una misma frase [16]. Consiste en comparar la locución de entrada con una serie de plantillas que representan a las unidades a reconocer [16]. El entrenamiento consiste únicamente en almacenar las distintas plantillas correspondientes a cada una de las unidades a reconocer. Las plantillas son por lo tanto un conjunto de características acústicas ordenadas en el tiempo [16]. Para el cálculo de puntuaciones es necesario un alineamiento temporal con posibles deformaciones elásticas y una medida de distancia [16]. A continuación, se describe el algoritmo para calcular esa distancia:

El objetivo es alinear de manera óptima la secuencia de vectores de parámetros de entrada $T = (t_1, t_2, \dots, t_N)$ con el modelo de referencia $R = (r_1, r_2, \dots, r_M)$ }, donde N es en general distinto a M debido a la variabilidad de la duración ya comentada antes [16]. Se necesita entonces una función que relacione las N muestras de la secuencia de entrada y las M de la plantilla, minimizando la distorsión entre ambas [16]. La función será de la forma m = W(n) y debe de cumplir además las siguientes restricciones:

- W (1) = 1
- W (N) = M.

Dadas dos secuencias cualesquiera, la función W(n) es el camino de alineamiento óptimo entre ambas y se obtiene resolviendo la siguiente ecuación:

$$D *= min\left\{\sum_{n=1}^{N} d\left[t_n, r_{w(n)}\right]\right\} \text{ (2) [16].}$$

Donde $d\left[t_{n,}, r_{w(n)}\right]$ es la distancia euclídea entre el instante n de la secuencia de entrada y el instante W(n) de la plantilla. Al final del alineamiento, D* es la distancia acumulada sobre el camino óptimo W(n) entre R y T y constituye la base para la puntuación resultante, en la que también pueden incluirse costes adicionales que penalicen caminos que sean demasiado no diagonales [16].

5.6.1 Distancia Euclidiana

La distancia euclidiana de los puntos $R = (r_1, r_2, \dots, r_n)$ y $T = (t_1, t_2, \dots, t_n)$, del espacio euclídeo n-dimensional, se define como:

$$d_E(R,T) = \sqrt{(r_1 - t_1)^2 + (r_2 - t_2)^2 + \cdots + (r_n - t_n)^2} \quad (3)$$

Nótese que esta definición depende de la existencia de coordenadas cartesianas sobre la variedad diferenciable, aunque en un espacio euclídeo pueden definirse sistemas de coordenadas más generales, siempre es posible definir un conjunto global de coordenadas cartesianas (a diferencia de una superficie curva donde sólo existen localmente).

Este cálculo permite identificar que tan diferentes son dos vectores o matrices de igual dimensión; donde la diferencia es directamente proporcional a la distancia calculada, es decir, cuanto mayor sea el valor de la distancia euclidiana entre dos vectores o matrices, más diferente es una de la otra.

6. DESARROLLO DEL PROYECTO

La figura 18 describe el sistema de reconocimiento de locutor texto dependiente en sus bloques funcionales, los cuales son la captura de la señal, el acondicionamiento de la señal, la identificación, extracción y reconocimiento, la base de datos y la interfaz de usuario.

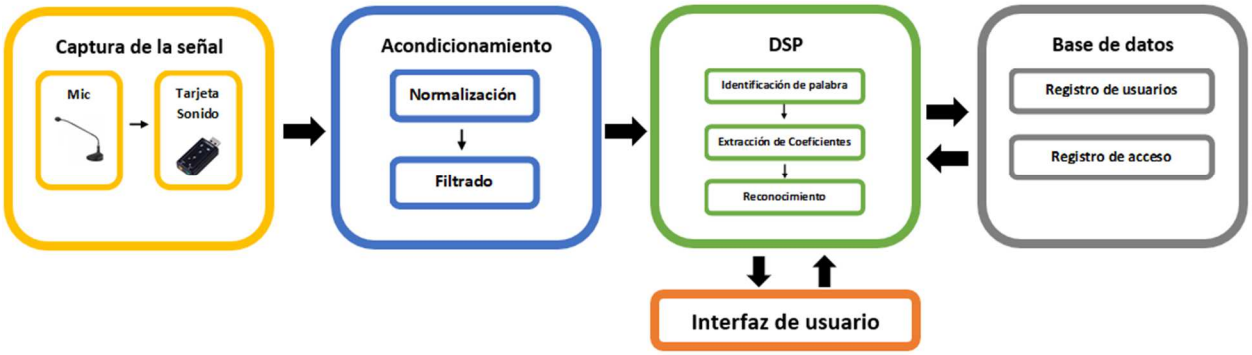

Figura 18. Diagrama de bloques del sistema

6.1 HARDWARE

A continuación, se describe el hardware utilizado para el desarrollo del proyecto.

6.1.1 Raspberry Pi 3 B

La raspberry pi 3 modelo b que es una placa de bajo costo y gran desempeño que tiene las siguientes características:

Procesador:

- Chipset Broadcom BCM2387.
- 1,2 GHz de cuatro núcleos ARM Cortex-A53

GPU

- Dual Core VideoCore IV ® Multimedia Co-procesador. Proporciona Open GL ES 2.0, OpenVG acelerado por hardware, y 1080p30 H.264 de alto perfil de decodificación.

- Capaz de 1 Gpixel / s, 1.5Gtexel / s o 24 GFLOPs con el filtrado de texturas y la infraestructura DMA

RAM: 1GB LPDDR2.

Conectividad

- Ethernet socket Ethernet 10/100 BaseT
- 802.11 b / g / n LAN inalámbrica y Bluetooth 4.1 (Classic Bluetooth y LE)
- Salida de vídeo
  - HDMI rev 1.3 y 1.4
  - RCA compuesto (PAL y NTSC)
- Salida de audio
  - jack de 3,5 mm de salida de audio, HDMI
  - USB 4 x Conector USB 2.0
- Conector GPIO
  - 40-clavijas de 2,54 mm (100 milésimas de pulgada) de expansión: 2x20 tira
  - Proporcionar 27 pines GPIO, así como 3,3 V, +5 V y GND líneas de suministro
- Conector de la cámara de 15 pines cámara MIPI interfaz en serie (CSI-2)
- Pantalla de visualización Conector de la interfaz de serie (DSI) Conector de 15 vías plana flex cable con dos carriles de datos y un carril de reloj.
- Ranura de tarjeta de memoria Empuje / tire Micro SDIO

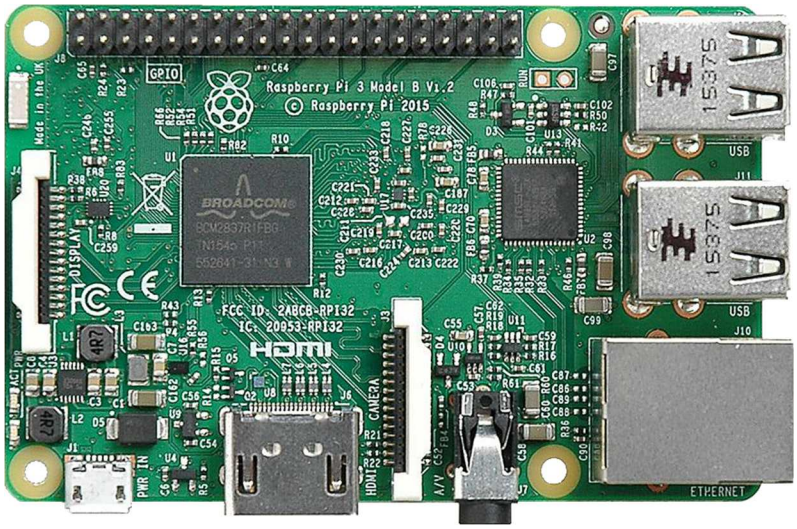

Figura 19. Raspberry Pi 3 modelo b [27]

6.1.2 CAPTURA DE AUDIO

La captura de la señal en la Raspberry Pi 3 B se realiza utilizando una tarjeta de audio y un micrófono. La selección del hardware utilizado es con base a la disponibilidad de los productos en el mercado nacional y su bajo costo; esto implica que la calidad de los elementos es baja y por tanto el sistema debe ser muy estable y robusto para garantizar su correcto funcionamiento. El hardware seleccionado es el siguiente:

6.1.2.1 Tarjeta de audio

La tarjeta de sonido utilizada en este proyecto es la 3-D 7.1 usb de alta velocidad tiene las siguientes características:

Técnicas

- Frecuencia de muestreo: 44100 Hz
- Ancho de banda: 10 Hz – 22050 Hz

Estándares y certificaciones

- USB 2.0
- CE
- FCC

- RoHS
- WEEE

Conexiones

- 1 entrada de micrófono
- 1 salida de audio
- 1 USB 2.0 A macho

General

- Sonido virtual envolvente de 7.1 canales
- Soporta sonido 3-D (AC-3)
- Amplificador digital de poder Clase 3
- Alimentados por bus

Diseño

- Dimensiones: 1.4 x 2.6 x 5.7 cm
- Peso: 8.5 g
- Chasis: plástico

Figura 20. Tarjeta de sonido

6.1.2.2 Micrófono

El micrófono utilizado para el desarrollo del proyecto es el de la figura 21.

Figura 21. Micrófono

Características:

- Conexión Jack de 3.5 mm
- Ancho de banda: 20 Hz – 18 KHz
- Patrón polar: Omnidireccional

6.2 SOFTWARE

El desarrollo del programa se realizó en el lenguaje Python en la versión 3.5.

6.2.1 Captura de la señal de voz

La captura de la señal de voz se realizó por medio de la librería pyaudio, la instalación de esta se realiza desde la terminal de la raspberry con los siguientes comandos en el orden especifico:

- sudo apt-get install git
- git clone http://people.csail.mit.edu/hubert/git/pyaudio.git
- sudo apt-get install libportaudio0 libportaudio2 libportaudiocpp0 portaudio19-dev
- sudo apt-get python-dev
- sudo python pyaudio/setup.py install

Una vez instalada se procede a la captura de audio definiendo los parámetros de la grabación:

- Frecuencia de muestreo
- Tiempo de grabación
- Nombre del archivo a guardar
- Formato del archivo

De esta forma se obtienen 5 señales de referencia para agregar un usuario y la señal a identificar. Las señales de referencia que el sistema almacena son como la que se muestra en la figura 22 y la señal a identificar es como la que se muestra en la figura 23. Estas figuras permiten visualizar que no hay saturación en las señales de referencia ni en las señales a identificar.

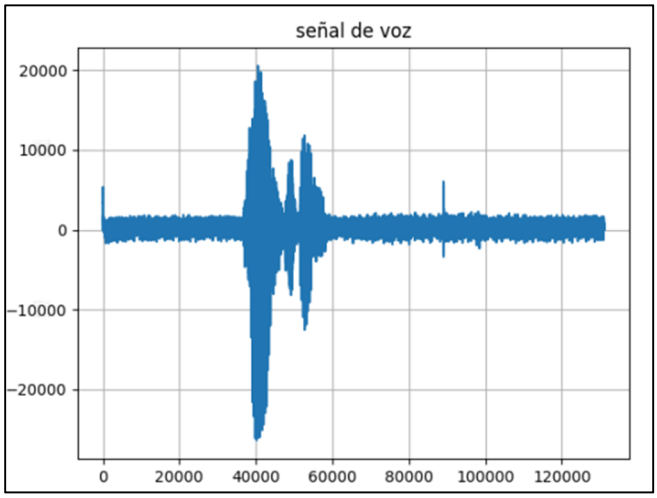

Figura 22. Señal de referencia.

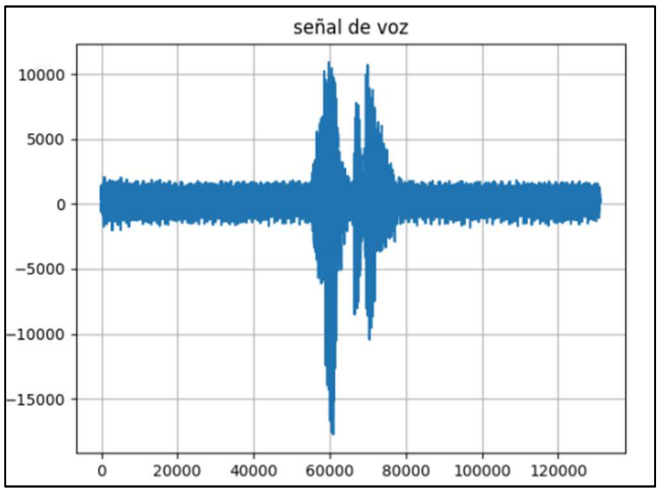

Figura 23. Señal a identificar.

6.2.2 Acondicionamiento de la señal de voz

Es muy importante acondicionar la señal antes de realizar cualquier tipo de manipulación y de extraer sus coeficientes cepstrales, para esto se Normaliza y se filtra la señal.

6.2.2.1 Normalización

Con el fin de realizar una manipulación optima se normaliza la señal, obteniendo el valor absoluto de la muestra más grande y posterior dividir la señal por este valor. Las figuras 24 muestra la señal de referencia normalizada. Este proceso nos permite eliminar diferencias en el análisis causadas por la intensidad con la que el usuario hable; así todas las señales son analizadas entre -1 y 1.

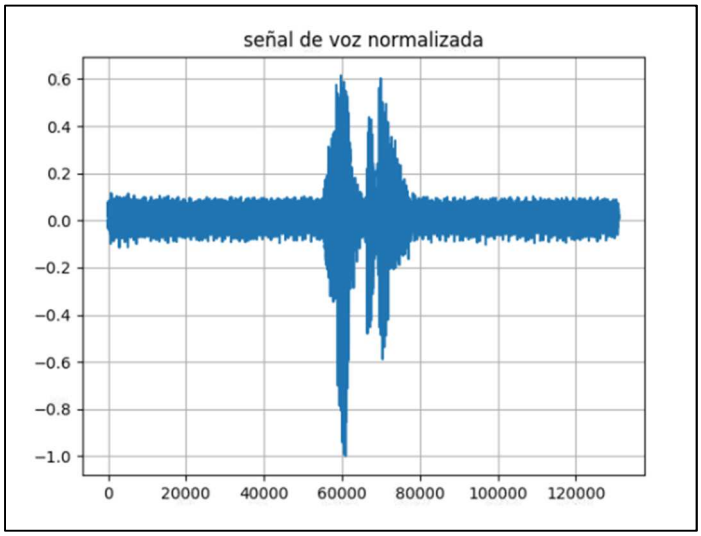

Figura 24. Señal de referencia normalizada.

6.2.2.2 Filtrado

Para eliminar ruidos ajenos a la voz se realizan dos etapas de filtrado:

- Filtro rechaza banda a 60Hz, elimina ruido proveniente de la red eléctrica.
- Filtro pasa bajos 4kHz, rango de frecuencia donde se encuentran los armónicos más significativos de la señal.

La implementación de los filtros se realizó con la librería Scipy. En las siguientes figuras se representa la respuesta en frecuencia de los filtros diseñados.

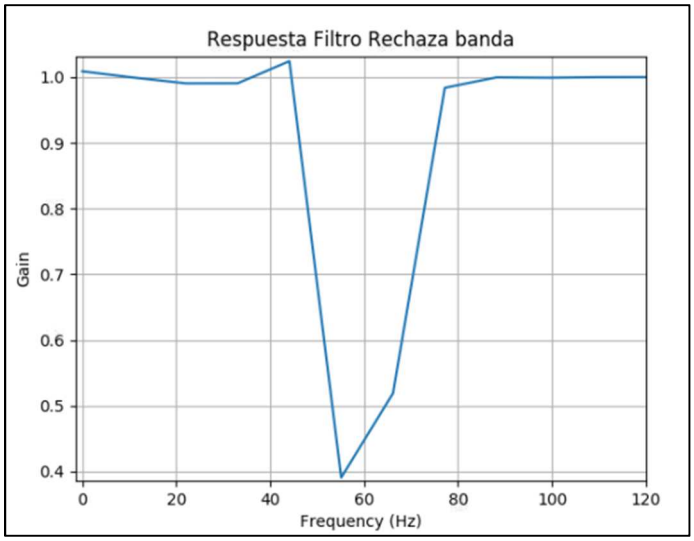

Figura 25. Respuesta filtro notch sintonizado en 60 Hz.

La figura 25 representa la respuesta en frecuencia el filtro notch con frecuencia central de 60 Hz implementado en el sistema. Es un filtro tipo butter con frecuencia baja de 55 Hz, frecuencia alta de 65 Hz y orden 3 para eliminar el ruido generado por la red.

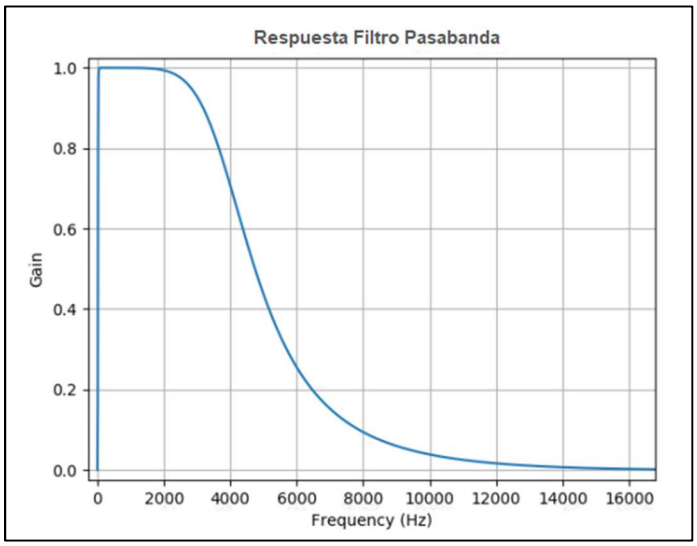

Figura 26. Respuesta filtro pasa banda 20 Hz a 4000 Hz

La figura 26 representa la respuesta en frecuencia el filtro pasa banda tipo butter con frecuencia baja de 20 Hz, frecuencia alta de 4000 Hz y orden 3 para eliminar la información que esta fuera del rango de frecuencias de la voz humana.

En el espectro de frecuencia de cada señal se visualiza como es atenuado el ruido en 60 Hz y en las frecuencias más altas.

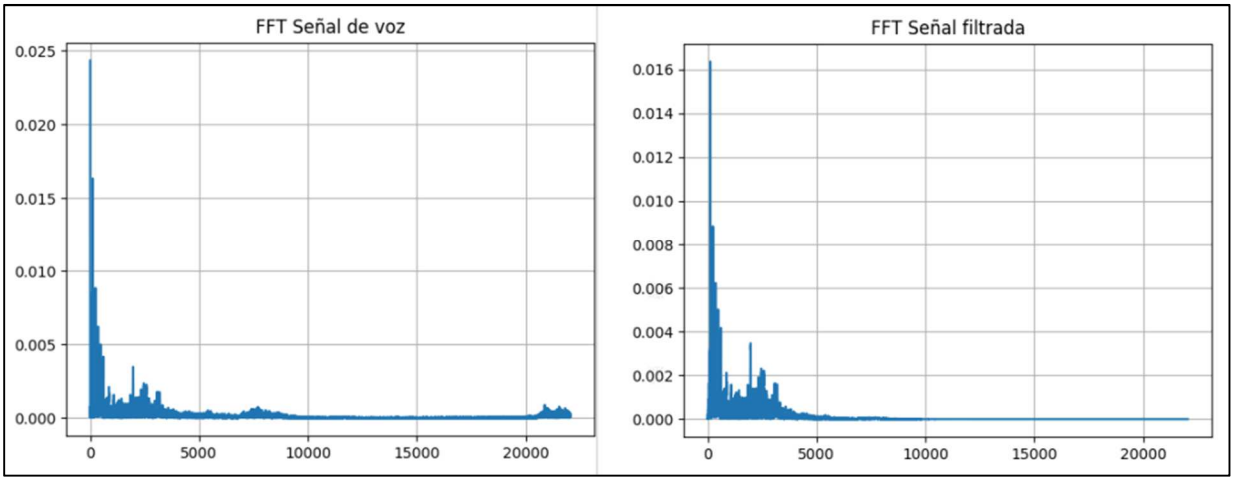

Figura 27. Análisis espectral señal antes y después de filtrar referencia.

En la figura 27 se puede ver cómo es atenuado el ruido después de 4KHz, eliminando frecuencias no pertenecientes a la voz que puedan entorpecer el proceso de reconocimiento.

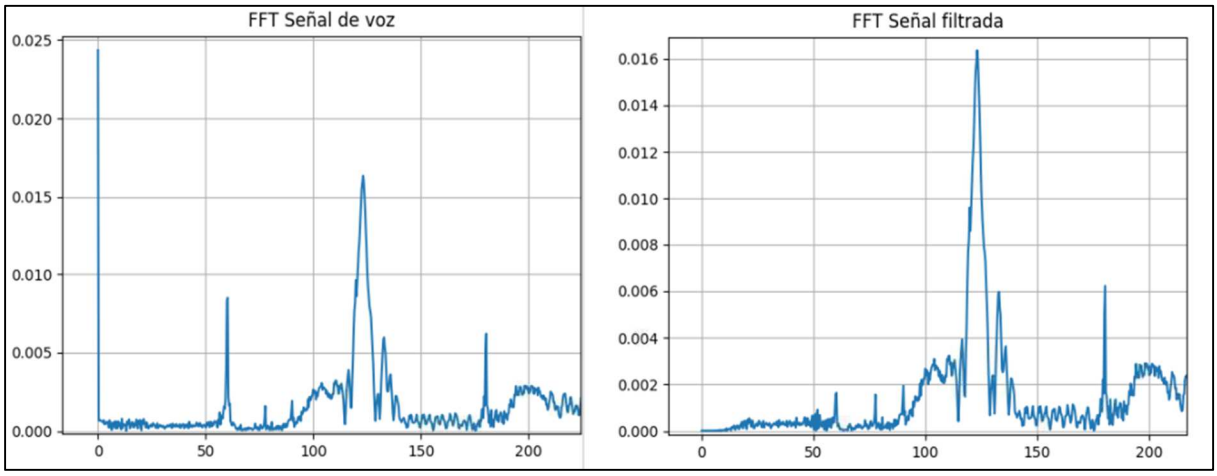

Figura 28. Ruido atenuado en 60 Hz Referencia.

En la figura 28 se presenta la atenuación de un armónico a 60 Hz que no pertenece a la voz del locutor.

6.2.3 DSP

En el procesamiento de la señal se determina el inicio y final de la palabra, se extraen los coeficientes cepstrales y se realiza la identificación del locutor.

6.2.3.1 Identificación del inicio y final de palabra

Una vez acondicionada la señal, lo primero es determinar en qué instante de tiempo se encuentra la información del locutor, para eso se implementó un algoritmo de identificación de inicio y fin de la palabra y así, eliminar información innecesaria. Este algoritmo se basa en el cálculo de energía total de la señal y determina en qué intervalo la energía es mayor indicando el inicio de la palabra y en que intervalo la energía vuelve a bajar indicando el fin de la palabra. Para este algoritmo es necesario instalar la librería numpy de igual forma que se instaló pyaudio.

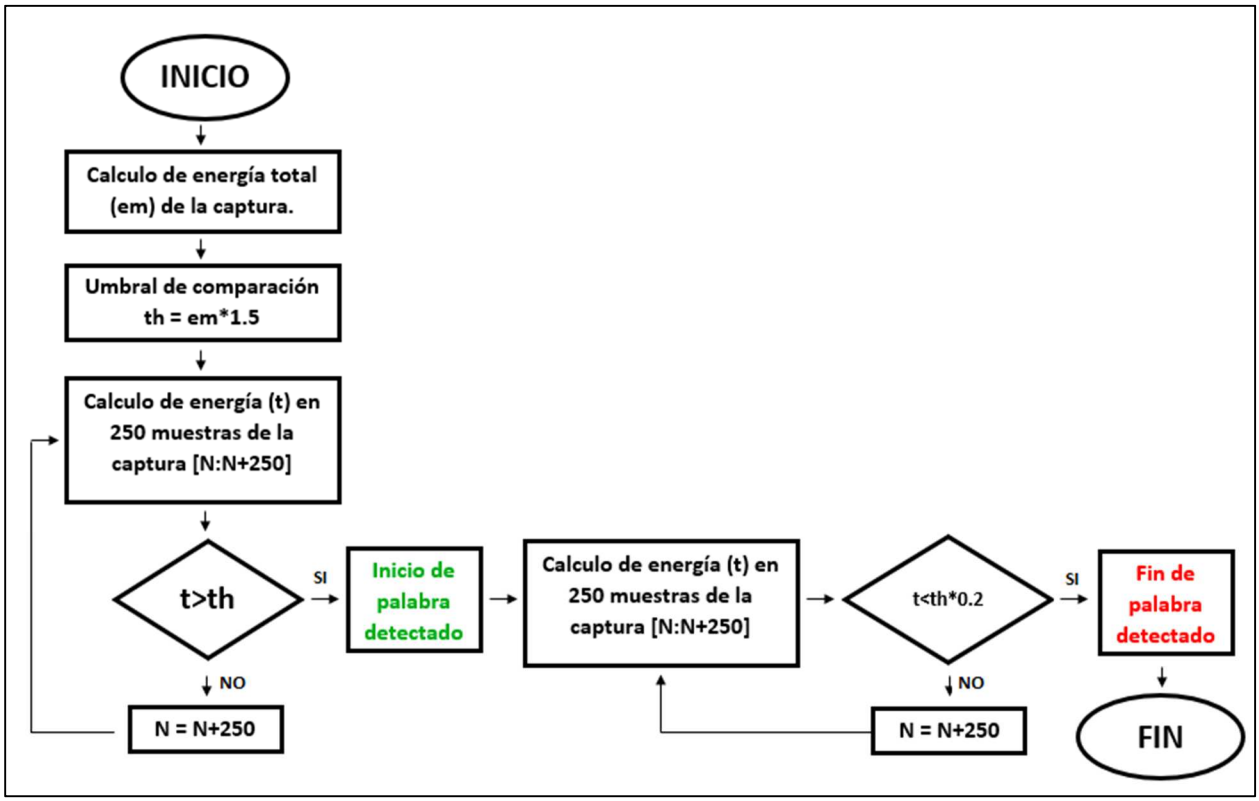

Figura 29. Diagrama de flujo algoritmo de identificación de inicio y fin.

La figura 29 muestra el diagrama de flujo del algoritmo implementado para la identificación del inicio y el final de la palabra. El algoritmo consiste en determinar la energía total de la señal capturada y con base a un umbral respecto a la energía total de la palabra se determina cual es el inicio y el final de la palabra calculando la energía de la señal cada 250 muestras.

6.2.3.2 Extracción de coeficientes cepstrales

Después de obtener una señal limpia se extraen los parámetros cepstrales de la palabra, para esto se utilizó la librería "Python_speech_features", esta librería proporciona las características de voz comunes (mfcc y banco de filtros) para los sistemas de reconocimiento de voz.

Para la extracción de los coeficientes es necesario especificar los siguientes parámetros:

- Signal: la señal de audio a partir de la cual se computan las características. Debe ser una matriz N * 1.
- Samplerate: la frecuencia de muestreo de la señal con la que estamos trabajando.
- nfft: el tamaño FFT. El valor predeterminado es 512.
- Winfunc: la ventana de análisis para aplicar a cada segmento. Por defecto no se aplica ninguna ventana. Para esa se utilizó la librería Windows y la función haming.

Como resultado de esta función se obtiene una matriz donde el número de filas corresponden a cada segmento de la palabra y el número de columnas al banco de filtros que más aportan a la señal. Aplicando la función a las señales de audio se obtienen sus coeficientes cepstrales que son almacenados en la base de datos. En la siguiente tabla se observan los datos que son obtenidos de la función.

-4.9279406	41.3512699	-25.977916	27.1443917	13.9719788	20.316928	6.37599635	1.03914087	-4.7558293	-5.1386027	4.00613539	-2.6167969	-0.913002
-4.3843479	52.2479048	-36.200272	34.196888	4.26431742	28.205261	-5.6370923	4.83011077	-14.181034	-0.41458	-0.6340742	1.88990289	-9.6679212
-3.6085011	50.1965999	-51.58355	45.8764088	3.47947328	23.2131794	0.4149522	2.61187919	-9.3579415	-7.4632949	-1.2736427	5.58662367	-11.212748
-1.4741808	45.583352	-55.177105	24.3414217	14.2937573	22.7547201	-8.5364628	-27.44461	14.162363	-17.205768	13.5836858	20.3787978	-1.5103969
-0.3477364	52.0953345	-65.35427	26.3994426	1.49446051	44.9091747	-31.379033	-35.923866	28.3328372	-21.97202	14.7980479	33.1923142	-6.8416681
-0.0304727	49.7857066	-64.773057	22.7853232	4.49972787	41.6937872	-33.690678	-39.114781	32.4895809	-20.998676	18.4898237	31.7376196	-11.569923
0.15792663	43.7094107	-58.899694	17.1714856	11.4681774	37.138669	-32.668797	-48.903532	36.0942323	-24.040928	21.4409617	35.7535367	-13.552363
0.25198981	30.6315804	-40.504126	-3.6315062	30.0183315	24.6064306	-35.541166	-36.046191	19.2127403	-8.7109397	19.9814892	29.2113296	-11.821736
0.26436679	42.2306061	-53.595269	8.89691417	22.7169665	28.9879868	-36.453913	-41.140447	25.744757	-13.235565	24.2415541	29.2649294	-13.725029
0.21997763	42.6627168	-52.159416	7.33594861	27.5776392	28.5555218	-39.139664	-42.533387	28.1337031	-14.225177	28.1526935	33.748679	-21.516101
0.12667268	50.549735	-62.612779	22.0458457	15.6660642	38.6219193	-46.183397	-39.867126	28.6726223	-18.15228	30.5984846	28.7890488	-24.59416
0.06935465	47.2746418	-56.904422	12.7568906	25.5840478	31.4747856	-39.93093	-39.518125	24.4926262	-17.134234	25.6454673	24.256366	-19.376017
-0.1313815	48.0048254	-58.940991	21.5077977	23.8287781	31.6801867	-37.273674	-44.841217	20.2019922	-11.160049	19.7340158	18.2558847	-15.13707
-0.3618801	38.491193	-47.755059	8.87448732	34.6391037	24.0026948	-25.212968	-36.667911	15.6135343	-15.835951	2.2458788	16.4099122	-13.926107
-0.7979656	45.2458871	-53.539891	19.6363597	28.6832524	27.346667	-19.288637	-40.432747	13.4229885	-21.403886	-4.1256846	7.6023873	-10.688028
-1.7568274	44.2566283	-56.088112	28.7984727	22.3716978	23.3540096	-10.368536	-31.707069	1.61877899	-21.273782	-0.3743645	5.54298963	-7.3237442
-2.6135938	30.6612321	-49.005103	33.0859089	30.8685614	-6.8937412	10.9010216	-10.008528	-18.115822	-5.7055505	14.7057039	0.19998882	-17.988785
-3.1941636	40.946037	-57.398858	41.9513577	29.522544	16.3468359	8.59496903	-14.633906	-22.540445	-14.092062	0.92819081	-0.736703	-22.493063
-3.708139	41.8538242	-59.85812	47.1622574	20.0677922	14.3977629	6.30066602	3.62225791	-19.706786	-11.627959	-1.6335315	-3.9466447	-16.095544
-4.3228688	39.9972067	-51.853926	51.1226985	10.9519357	8.45813263	12.7948017	-0.3831505	-13.940327	-18.126361	2.85678622	4.72579229	-17.03864
-4.8300681	31.9451209	-39.196997	47.3278141	11.4849977	5.87271587	18.3922559	-7.0301836	-11.35881	-23.906108	1.47701255	8.02643498	-10.286506
-4.856327	33.3639699	-52.010828	54.2228427	-2.2790349	15.0390558	17.713643	8.15828458	-10.958255	-23.949688	-2.2167751	14.032256	-23.390341
-5.3329134	25.7600814	-48.062261	45.2794169	-2.7791075	3.07794334	19.6749836	-3.6175197	1.287067	-31.688951	-10.555041	11.1011084	-23.908642
-5.0816139	24.6985279	-49.746447	37.0213121	9.02186507	-0.0076833	-4.8560976	-5.1342216	10.1297301	-30.351177	-20.499985	9.8956781	-15.47867
-3.2360986	19.5845208	-60.520036	50.0271432	6.09422152	2.80098059	-13.482858	8.14604999	20.0977021	-15.394801	-23.143796	13.7487035	-16.521226
-2.9385403	4.69935522	-51.923101	53.0913264	-13.512849	-0.8485225	-8.147694	-11.044639	3.08126704	-22.870596	-22.435787	13.9974365	-7.9205387
-2.4106648	-0.4736421	-53.930589	57.6703402	-9.6954008	0.25965283	-3.4683672	-5.0010793	12.3970539	-16.428445	-27.120923	9.85458143	-2.7682428
-1.7671012	-2.1951767	-61.043796	51.791879	-8.9168815	-2.5666489	-7.2084338	-19.642622	8.53721904	-13.933215	-19.048785	11.9801414	-6.9814327
-2.1067441	8.72138604	-66.515399	47.0417828	-17.135221	-0.5771691	-8.9022075	-14.453123	7.22742067	-17.49189	-20.08161	22.0449151	2.78798427
-3.4448002	6.06672641	-54.711258	34.6088742	-0.40544	-1.6228532	9.27742594	-14.415128	12.0315893	-20.083331	-11.770005	12.9722088	1.52281216
-4.5231007	19.6801997	-62.018318	34.7523668	-11.615162	6.50332441	-1.1279322	-20.787656	4.41236676	-25.528394	-13.631377	6.45207064	-1.0891735

Tabla 2. Matriz de coeficiente cepstrales de una señal.

Cada una de las señales que son analizadas en el sistema generan una matriz como la que se muestra en la tabla 2 y, con base a estas matrices, el sistema realiza el reconocimiento del locutor.

Todas las matrices se componen de trece columnas las cuales corresponden a la cantidad de frecuencias que analiza la función en cada señal, estas frecuencias son las que más información poseen de la señal y son las más bajas. Cada una de las filas corresponde a una ventana de 25 ms de la señal, este es el ventaneo que realiza la función y cada ventana es analizada en las trece frecuencias para obtener un dato significativo. La cantidad de filas de la matriz siempre es dependiente de la longitud de la señal a analizar, mientras que la cantidad de columnas siempre es de trece.

6.2.3.3 Técnica de reconocimiento de locutor

Con el fin de reconocer al locutor se realiza una comparación entre los coeficientes cepstrales de la voz a identificar y los coeficientes cepstrales de las referencias almacenadas en el sistema. Esta comparación se realiza fila por fila calculando la distancia euclidiana que existe entre ellas.

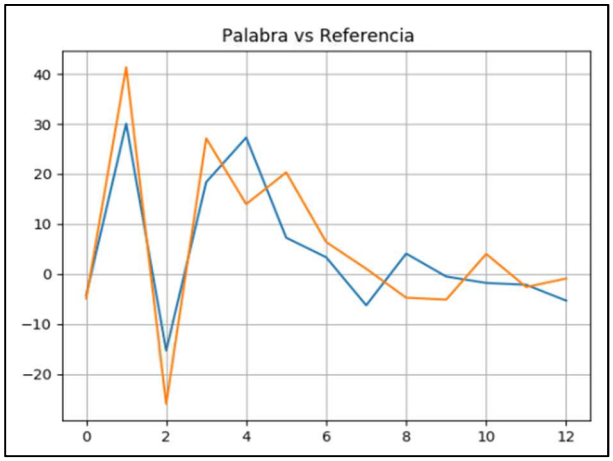

Figura 30. Palabra vs Referencia

La figura 30 muestra la comparación entre una señal de solicitud de acceso, para este caso denominada palabra, y una referencia almacenada en el sistema. La distancia euclidiana es calculada para cada una de las filas de las matrices de coeficientes de las dos señales y luego, se calcula la distancia promedio. Para que esta operación pueda ser efectuada es necesario que las matrices tengan igual número de filas, por tanto, se calcula la longitud de las dos matrices y se completa con 0 la matriz más pequeña.

Para hacer más eficiente el sistema y garantizar un alto grado de confiabilidad, una vez se encuentre una distancia menor a 40, el sistema deja de comparar y determina que la identidad del locutor es la de la última referencia analizada concediendo el acceso al usuario, en caso de no encontrar una referencia que cumpla con el umbral de distancia establecido, el sistema niega el acceso al usuario.

6.2.4 Manejo de base de datos local

Para la creación de la base de datos del sistema se utiliza MySQL. Esta base de datos tiene el propósito de almacenar toda la información correspondiente a cada uno de los usuarios del sistema, desde los parámetros de las muestras adquiridas en la fase de entrenamiento del sistema hasta el registro de acceso de cada uno. Cada una de las matrices correspondientes a los parámetros de cada muestra de los usuarios es almacenada en una tabla en la base de datos con el nombre correspondiente al usuario que se ha registrado y el número de la muestra. Además, se crea otra tabla que solo almacena el nombre de los usuarios del sistema y el orden en el cual se registraron y, una última tabla que registra con nombre, fecha y hora el acceso de cada uno de los usuarios.

La creación de la base datos se realiza directamente en el intérprete de MySQL, MariaDB, y la creación, escritura y lectura del contenido de cada una de las tablas se realiza desde Python con la librería MySQL.connector, la cual permite establecer la conexión con la base de datos y ejecutar cada uno de queries necesarios para manipular el contenido de las tablas en la base datos. El detalle de este procedimiento de puede ver en el anexo 8.

6.2.5 Creación del registro de acceso de los usuarios

Para generar el registro de acceso de los usuarios, se realiza el almacenamiento del nombre del usuario, la fecha y la hora en la tabla de registro de la base de datos cuando se identifica a un locutor como un usuario perteneciente al sistema. Haciendo uso de la librería MySQL.connector y la libraría csv el sistema puede crear y modificar de archivos .csv con base a la información almacenada en la base datos como se detalla en el anexo 17.

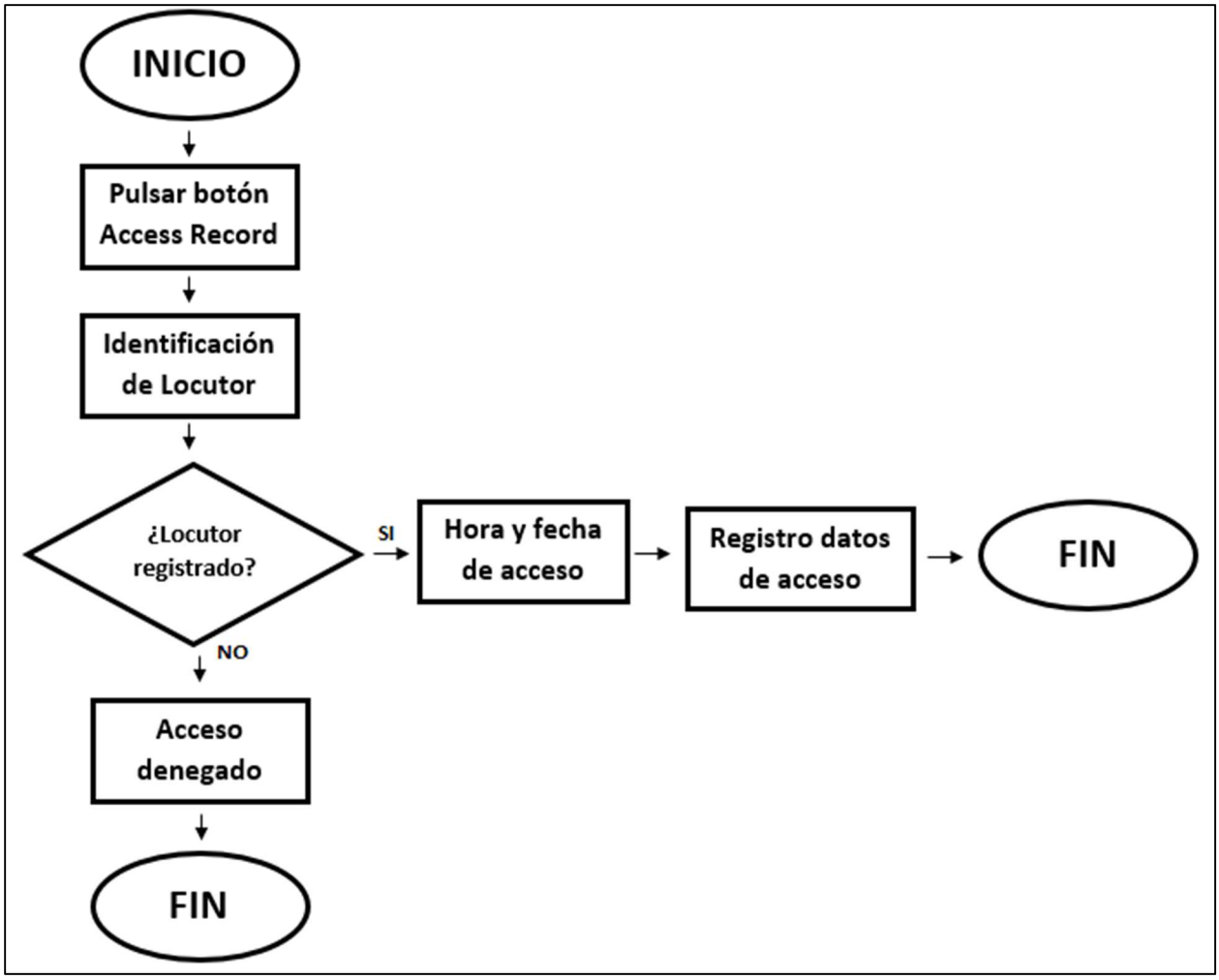

Figura 31. Diagrama de flujo Registro de acceso.

La figura 31 representa el diagrama de flujo del algoritmo implementado para llevar el registro de acceso de los usuarios; teniendo en cuenta que el sistema ya tiene usuarios registrados, se solicita el acceso y en caso de existir una coincidencia con alguna de las referencias de los usuarios almacenados, se concede el acceso, se adquiere la hora y la fecha del momento del acceso y se guarda en la base de datos regenerado el registro con el nombre del usuario que accedió al sistema. Si se quiere acceder al registro completo de accesos, se pulsa el botón “Access Record” y el sistema genera un archivo con extensión .csv.

6.2.6 Diseño e implementación de la interfaz de usuario

La interfaz de usuario del sistema es la que se muestra en la figura 32.

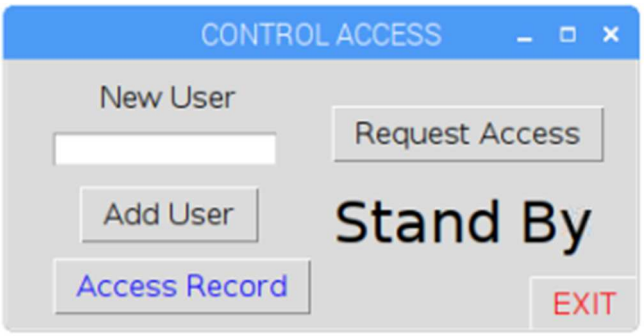

Figura 32. Interfaz de usuario del sistema de control de acceso.

La interfaz se compone de un texto en la parte superior izquierda para indicar que es la parte para agregar nuevos usuarios al sistema “New User”, debajo esta una caja de texto donde el usuario escribe el nombre por el cual será identificado en el sistema y debajo está el botón de agregar usuario “Add User” con el cual, se inicia la adquisición de las 5 muestras de los usuarios para almacenarlas en la base de datos, tras cada muestra almacenada se indica, a través de una ventana emergente, la cantidad de muestras restantes por adquirir. Además, en la parte inferior izquierda se ubica el botón “Access Record” que permite generar el registro de acceso de los usuarios y almacenarlo en un archivo csv.

En la parte superior derecha se encuentra el botón “Request Access”, el cual, activa el modo “Listenning” del sistema por tres segundos para escuchar al locutor, procesar la señal de audio e identificar si quien acaba de hablar pertenece o no el sistema; en caso de pertenecer se muestra una ventana emergente como aparece en la figura 33, de lo contrario la ventana emergente que se muestra es la que aparece en la figura 34.

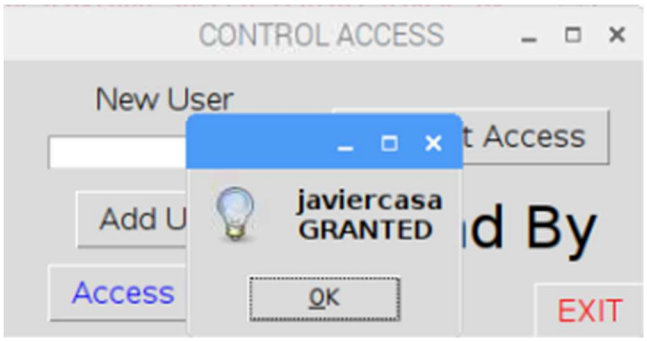

Figura 33. Acceso concedido

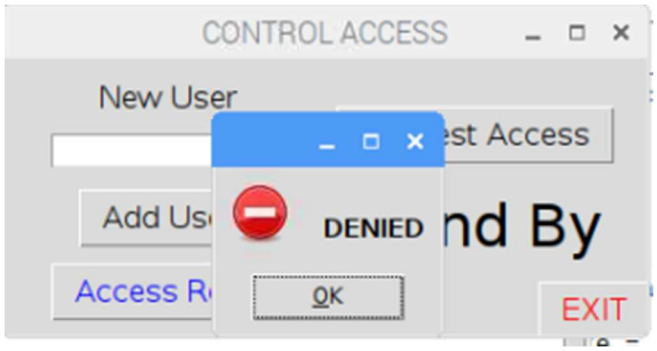

Figura 34. Acceso denegado.

En el centro de la parte derecha de la interfaz se ubica un texto que indica al usuario si el sistema está a la espera de una instrucción o si está escuchando y en la parte inferior derecha se encuentra el botón “EXIT” que finaliza el programa.

7. PRUEBAS, RESULTADOS Y ANÁLISIS

7.1 PRUEBAS DE ACONDICIONAMIENTO

Para someter a prueba el desempeño del sistema de reconocmiento y así determinar su error y confiabilidad, se ejecutaron algunas validaciones. Primero, se identificó la importancia del acondicionamiento de la señal en la extracción de los parametros cepstrales. Con un solo locutor se tomaron 5 referencias, se realizó la extracción y el almacenamiento de sus parametros cepstrales sin el acondicionamiento de la señal y luego, se acondicionó cada una de las 5 referencias y se realizó la extracción y el almacenamiento de los parametros cepstrales.

Resultados sin acondicionamiento:

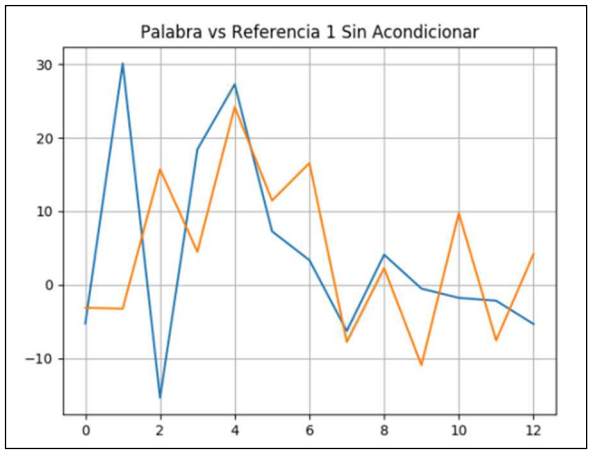

Figura 35. Muestra vs ref. 1 sin acondicionamiento.

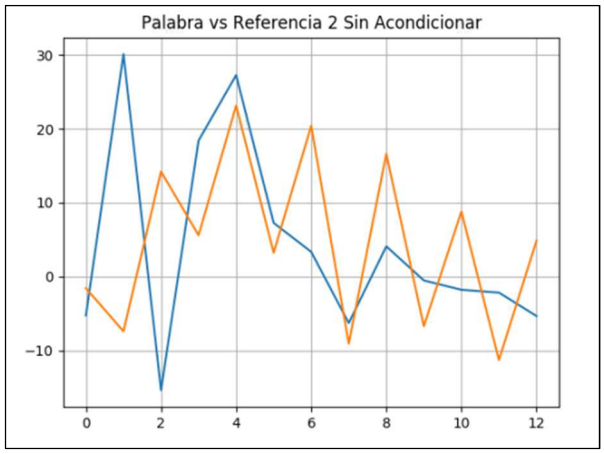

Figura 36. Muestra vs ref. 2 sin acondicionamiento.

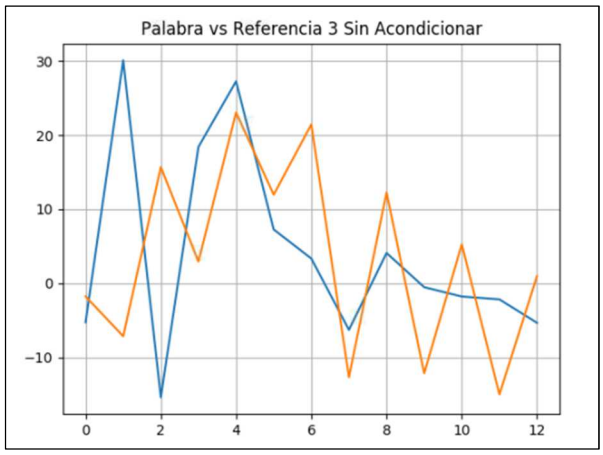

Figura 37. Muestra vs ref. 3 sin acondicionamiento.

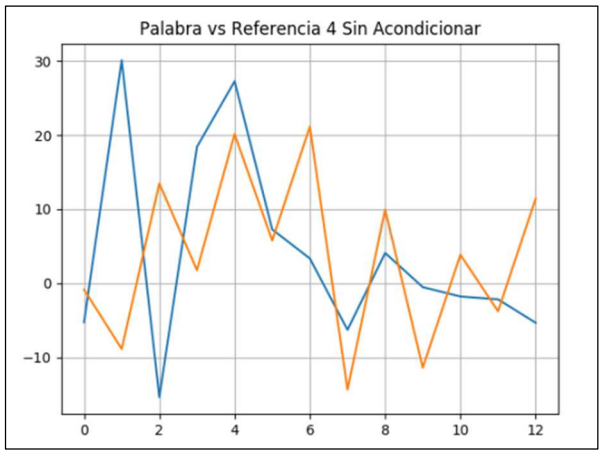

Figura 38. Muestra vs ref. 5 sin acondicionamiento.

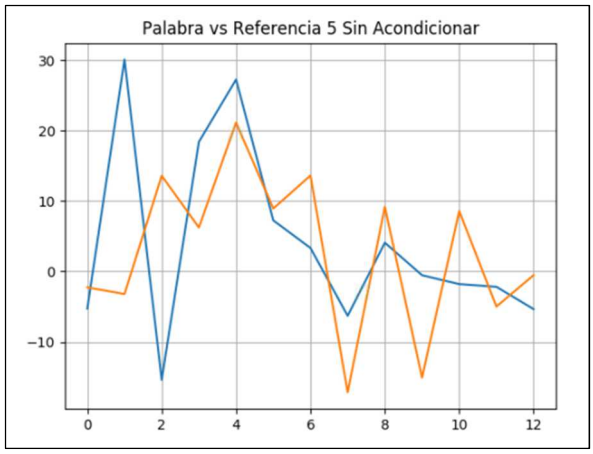

Figura 39. Muestra vs ref. 5 sin acondicionamiento.

Resultados con acondicionamiento:

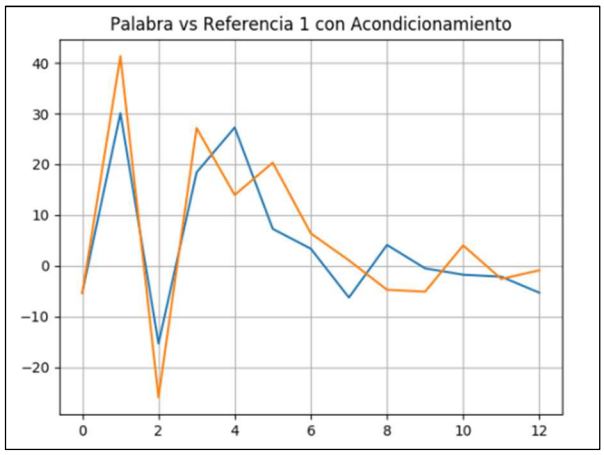

Figura 40. Muestra vs ref. 1 con acondicionamiento.

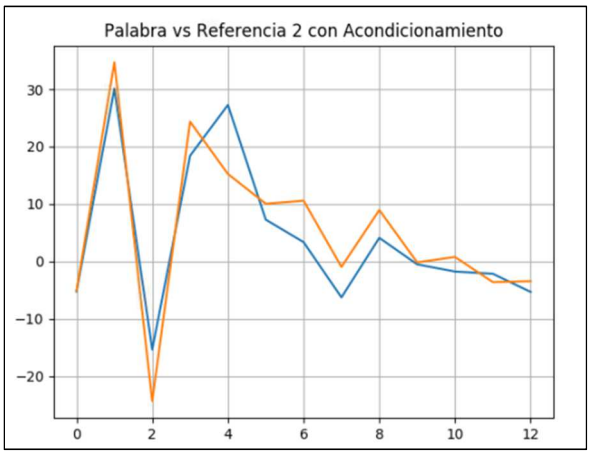

Figura 41. Muestra vs ref. 2 con acondicionamiento.

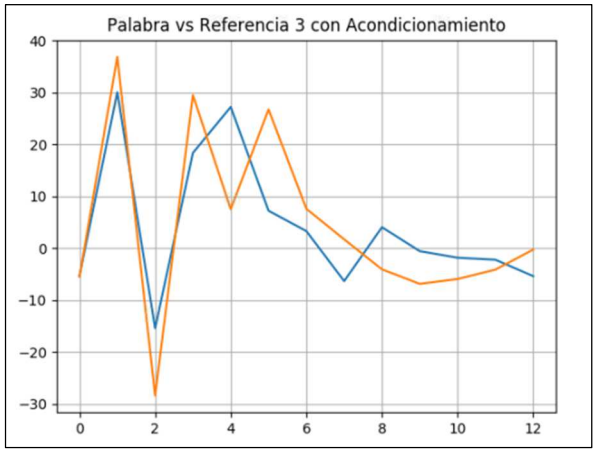

Figura 42. Muestra vs ref. 3 con acondicionamiento.

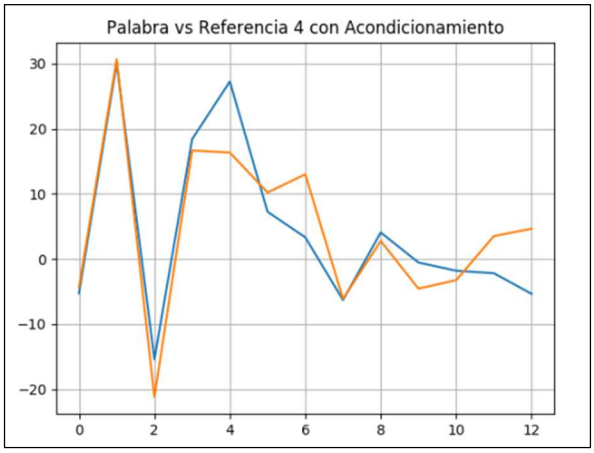

Figura 43. Muestra vs ref. 4 con acondicionamiento.

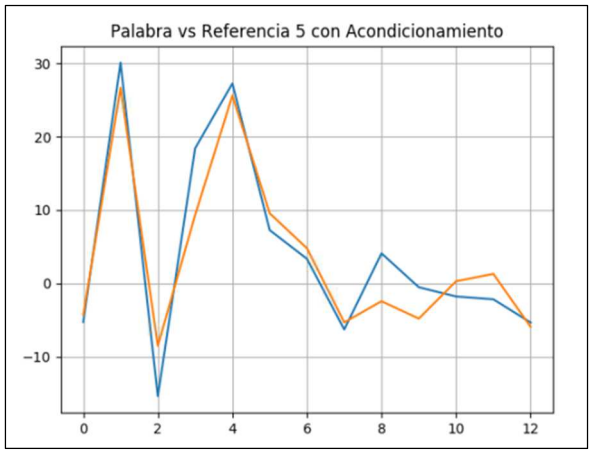

Figura 44. Muestra vs ref. 5 con acondicionamiento.

Palabra Vs Ref	Distancia Euclidiana	
	Con Acond	Sin Acond
Referencia 1	40.08602	80.67623
Referencia 2	41.74369	79.00918
Referencia 3	37.02274	82.4418
Referencia 4	42.13549	83.42319
Referencia 5	35.033942	80.94977
Promedio	39.2043764	81.300034

Tabla 3. Comparacion de distancias con y sin acondicionamiento.

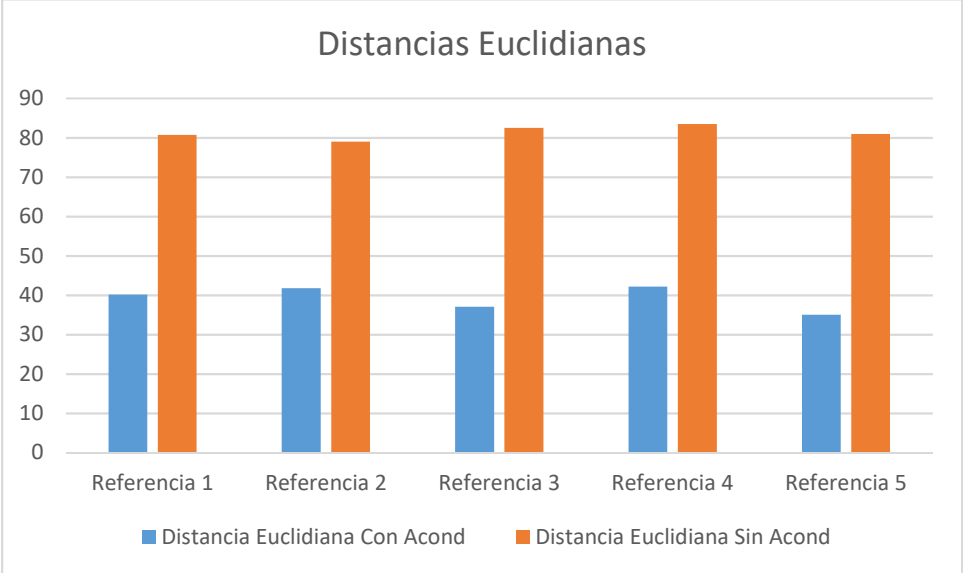

Figura 45. Distancia eucilidea de los parametros cepstrales entre muestra y referencias.

Como se muestra en la figura 45, el no realizar un acondicionamiento de las señales aumenta significativamente la distancia calculada entre los parametros cepstrales de la muestra y los de la referencia.

7.2 PRUEBAS DE ALGORITMO DE RECONOCIMIENTO

Para determinar la confiabilidad del algoritmo implementado, se comparó el algoritmo de Distancia Euclidiana (DE) con la funcion de correlación de la librería numpy como método de reconocimiento. Diez locutores se registraron en el sistema y solicitaron acceso al mismo con el algoritmo de DE y después con la funcion de correlación. Cada locutor solicito acceso al sistema 20 veces tanto con el algoritmo de DE como con la función de correlación, los resultados se observan en la tabla 1.

Locutor	**N° de Intentos**	**Aciertos DE**	**%Aciertos DE**	**Aciertos Correlación**	**%Aciertos correlación**
Locutor 1	20	18	90%	19	95%
Locutor 2	20	18	90%	19	95%
Locutor 3	20	20	100%	20	100%
Locutor 4	20	17	85%	19	95%
Locutor 5	20	18	90%	19	95%
Locutor 6	20	19	95%	20	100%
Locutor 7	20	17	85%	18	90%
Locutor 8	20	19	95%	19	95%
Locutor 9	20	18	90%	19	95%
Locutor 10	20	18	90%	20	100%
promedio	20	18.2	**91%**	19.2	**96%**

Tabla 4. Ingreso al sistema DE(distancia euclidiana) vs correlacion.

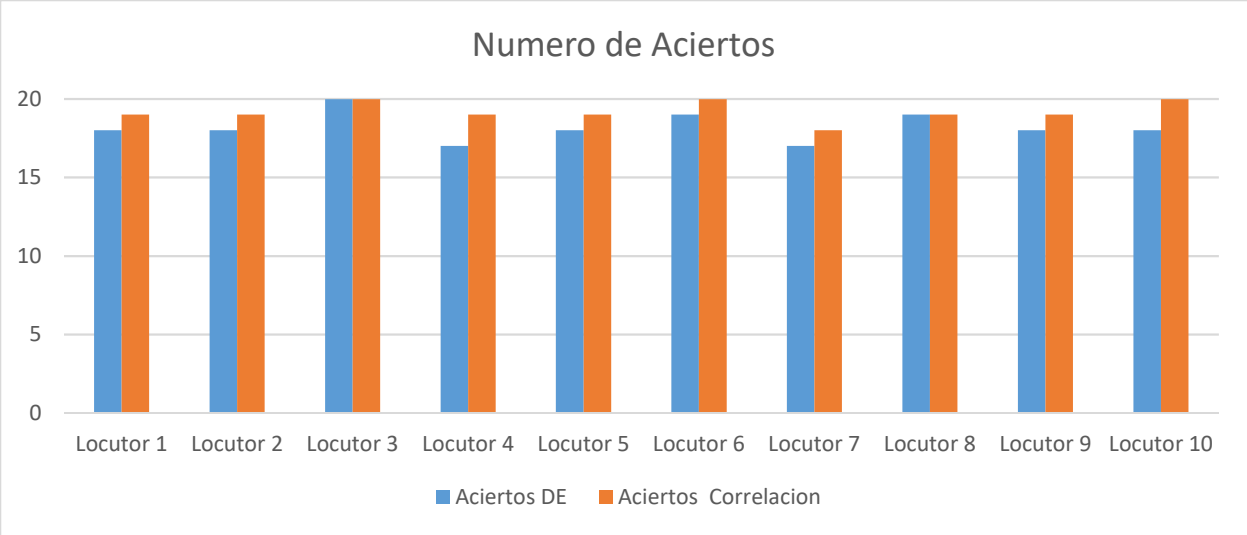

Figura 46. Numero de aciertos DE vs número de aciertos correlacion.

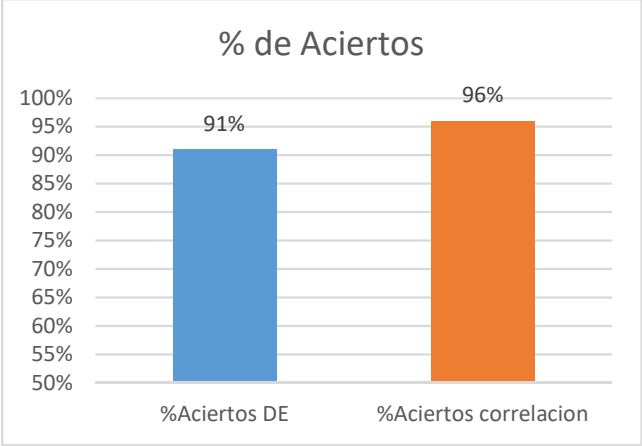

Figura 47. Porcentaje de aciertos DE vs porcentaje de aciertos Correlacion.

Se observa en las gráficas 46 y 47 que el porcentaje de aciertos del algoritmo DE es 5% menor a la función de correlación, sin embargo, es superior al 90% haciéndolo un algoritmo de reconocimiento confiable.

7.3 PRUEBAS FINALES

Una vez concluidas las validaciones mencionadas, se realizaron dos sesiones de pruebas a 20 locutores. Los primeros 10 con usuario registrado, solicitando acceso al sistema; los 10 locutores restantes solicitando acceso sin tener usuario en el sistema, con el fin de suplantar la identidad de los primeros 10 locutores. Cada sesión de pruebas se realizó con 20 iteraciones y una palabra especifica por locutor.

7.3.1 Locutor verdadero

En las tablas 5 y 6 se visualizan los resultados de cada sesión.

Locutor Verdadero	Sexo	Edad	N° de Intentos	Aciertos Verdaderos	Error (%)
Locutor 1	F	14	20	16	20
Locutor 2	F	14	20	17	15
Locutor 3	F	25	20	19	5
Locutor 4	F	31	20	20	0
Locutor 5	F	40	20	19	5
Locutor 6	M	15	20	18	10
Locutor 7	M	12	20	15	25
Locutor 8	M	46	20	20	0
Locutor 9	M	28	20	18	10
Locutor 10	M	34	20	20	0
				Promedio de error	**9%**

Tabla 5. Resultados aciertos verdaderos sesión 1.

Locutor Verdadero	Sexo	Edad	N° de Intentos	Aciertos Verdaderos	Error (%)
Locutor 1	F	14	20	16	20
Locutor 2	F	14	20	15	25
Locutor 3	F	25	20	20	0
Locutor 4	F	31	20	19	5
Locutor 5	F	40	20	19	5
Locutor 6	M	15	20	17	15
Locutor 7	M	12	20	16	20
Locutor 8	M	46	20	19	5
Locutor 9	M	28	20	18	10
Locutor 10	M	34	20	20	0
				Error Promedio	**10,5%**

Tabla 6. Resultados aciertos verdaderos sesión 2.

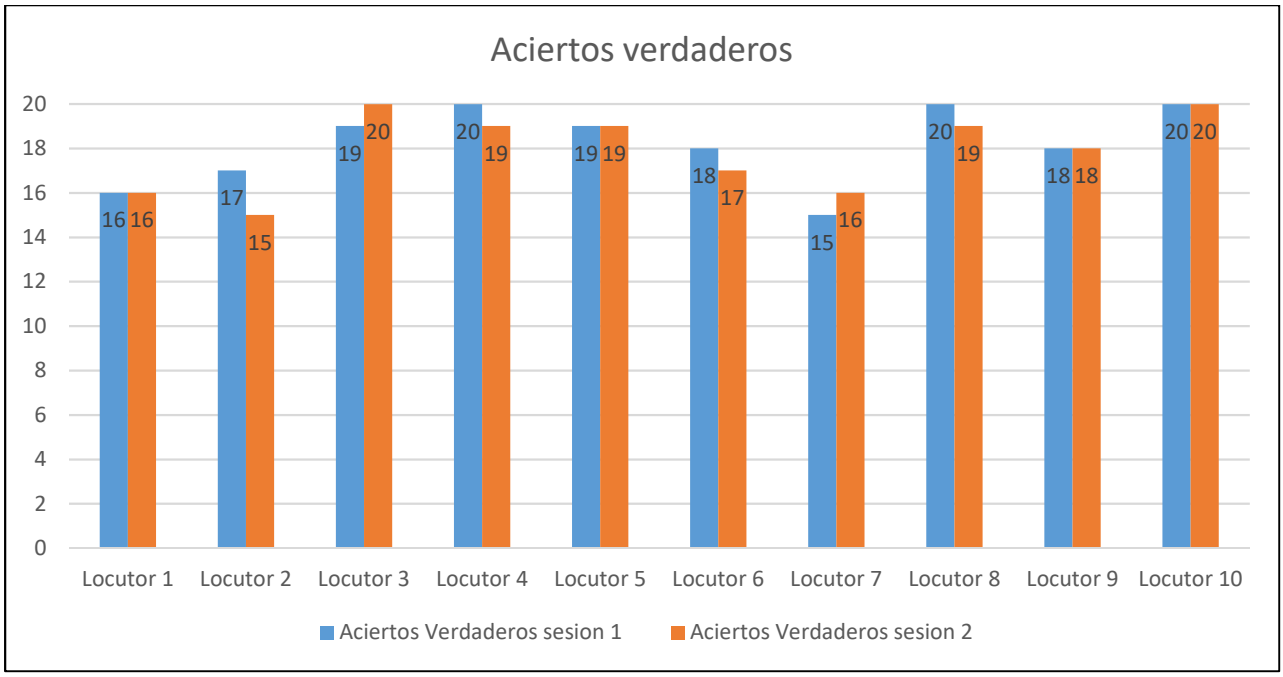

Figura 48. Aciertos verdaderos primera y segunda sesión.

Promediando los errores de la primera y la segunda sesión (9% y 10,5% respectivamente) el sistema presento un error del 9,75% permitiendo el acceso a los locutores registrados el 90,25% de las interacciones.

7.3.2 Locutor Falso

En las tablas 7 y 8 se muestran los resultados de las suplantaciones.

Locutor falso	Sexo	Edad	N° de Intentos	Rechazos Verdaderos	Error (%)
Locutor 11	F	11	20	18	10
Locutor 12	F	13	20	19	5
Locutor 13	F	22	20	20	0
Locutor 14	F	33	20	20	0
Locutor 15	F	35	20	20	0
Locutor 16	M	14	20	19	5
Locutor 17	M	15	20	19	5
Locutor 18	M	38	20	20	0
Locutor 19	M	25	20	20	0
Locutor 20	M	39	20	20	0
				Error Promedio	**2,5%**

Tabla 7. Resultados de Rechazos verdaderos sesión 1.

Locutor falso	Sexo	Edad	N° de Intentos	Rechazos Verdaderos	Error (%)
Locutor 11	F	11	20	17	15
Locutor 12	F	13	20	19	5
Locutor 13	F	22	20	19	5
Locutor 14	F	33	20	20	0
Locutor 15	F	35	20	20	0
Locutor 16	M	14	20	19	5
Locutor 17	M	15	20	18	10
Locutor 18	M	38	20	20	0
Locutor 19	M	25	20	20	0
Locutor 20	M	39	20	20	0
				Error Promedio	**4%**

Tabla 8. Resultados de Rechazos verdaderos sesión 2.

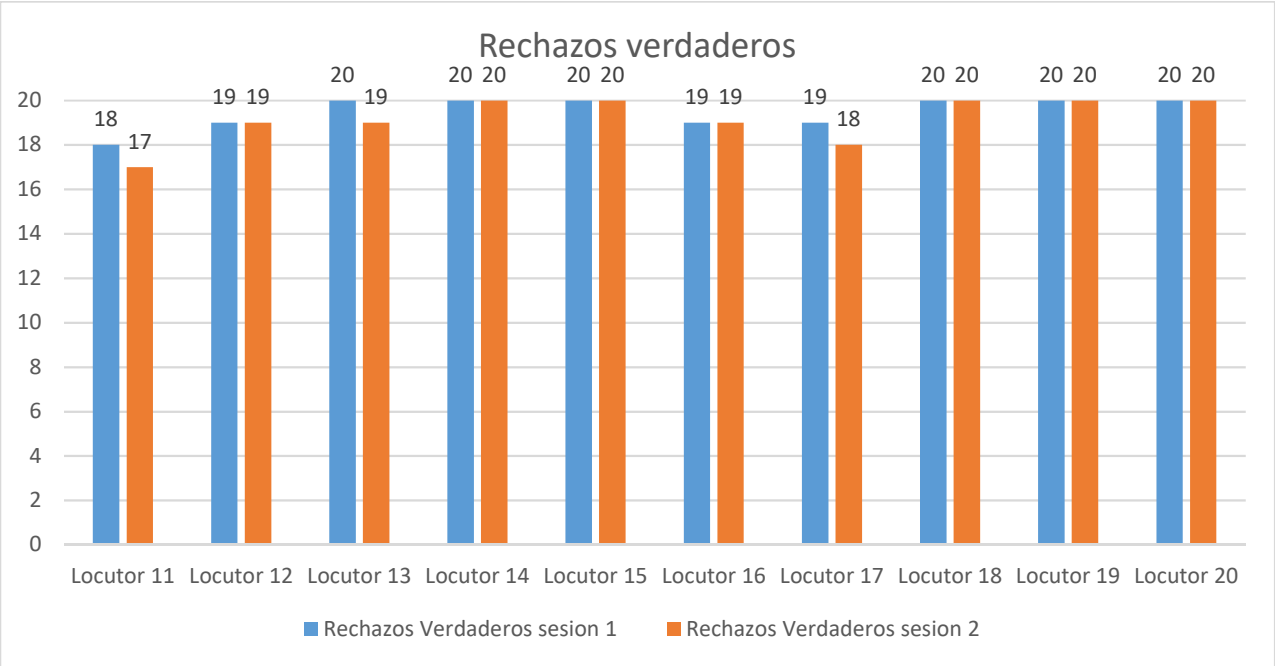

Figura 49. Rechazos verdaderos primera y segunda sesión.

Promediando los errores de la primera y la segunda sesión con locutores falsos (2,5% y 4% respectivamente) el sistema presento un error del 3,25 % negando el acceso a los locutores falsos el 96,75% de los intentos.

7.4 ANÁLISIS

Con base a los resultados presentados en las tablas 5, 6, 7 y 8, se puede identificar que el sistema presenta mayor porcentaje de error con los locutores 1, 2 6, 7, 11, 12, 16, 17; estos locutores tienen una edad que oscila entre los 11 y 15 años. La inestabilidad de la voz en estas edades cuando se está definiendo el tono, el timbre, el color y la intensidad de la voz de una persona afecta considerablemente el desempeño del sistema, el cual, se funciona mucho mejor con usuarios que ya han pasado por esta etapa de la vida y tienen las características de su voz definidas.

La confiabilidad del sistema es calculada teniendo en cuenta el error promedio en las dos sesiones de los locutores verdaderos y los locutores falsos. La tabla 9 muestra los errores promedio del sistema reconociendo al locutor verdadero y rechazando al locutor falso.

Locutor	**Error (%)**
Locutor verdadero	9,75
Locutor falso	3,25
Error total del sistema	**6,5**

Tabla 9. Error total del sistema.

De acuerdo a los datos obtenidos en las dos sesiones de prueba, el promedio de error total del sistema de reconocimiento de voz utilizando los parámetros cepstrales de la voz destinado al control de acceso implementado en la Raspberry Pi 3 B fue del 6,5%, por tanto, la confiabilidad del sistema es del **93.5%.**

8. CONCLUSIONES

- En los resultados obtenidos de las pruebas de acondicionamiento, se puede observar que es fundamental acondicionar la señal para su correcto procesamiento, esto permite obtener parámetros cepstrales de la voz altamente precisos en cada una de las muestras, logrando una tasa de reconocimiento más alta como se observa en la figura 45, otorgándole al sistema un alto grado de confiabilidad.
- Las pruebas realizadas en el numeral 7.2 demuestran que el algoritmo de distancia euclidiana desarrollado para el reconocimiento del locutor es altamente confiable; la diferencia respecto a la función de correlación de la librería numpy es de tan solo el 5% y su tasa de aciertos es superior al 90%.
- Luego de realizar las pruebas finales, se concluye que el sistema de reconocimiento de locutor texto dependiente basado en los parámetros cepstrales de la voz desarrollado en la Raspberry Pi 3 B es un sistema altamente confiable con tan solo un error total promedio del 6,5%, es decir, tiene una confiabilidad del 93,5% como se observa en la tabla 9.
- En los resultados de las pruebas finales se puede observar que la edad del locutor influye considerablemente en la confiabilidad del sistema; las pruebas muestran que el mayor porcentaje de error del sistema para reconocer o rechazar un usuario fue del 25% con los locutores cuya edad estuvo entre los 11 y 15 años.

9. BIBLIOGRAFÍA

[1] S. Ozaydin, "A Text- - - Independent Speaker Recognition System," *IEEE J. Sel. Top. Signal Process.*, pp. 1–5, 2017.

[2] R. Mukherjee, T. Islam, and R. Sankar, "Text dependent speaker recognition using shifted MFCC," *Conf. Proc. - IEEE SOUTHEASTCON*, pp. 1–4, 2013.

[3] Cucorent, "Sistemas de Control de Accesos y Presencia - Cucorent." [Online]. Available: https://www.cucorent.com/. [Accessed: 06-Nov-2018]

[4] D. D. T. Thu, L. T. Van, Q. N. Hong, and H. P. Ngoc, "Text-dependent speaker recognition for vietnamese," *2013 Int. Conf. Soft Comput. Pattern Recognition, SoCPaR 2013*, pp. 196–200, 2013.

[4] D. Scheips and A. Abloy, "Voice recognition – benefits and challenges of this biometric application for access control", 2010.

[5] N. Hammami, M. Bedda, N. Farah, and R. O. Lakehal-Ayat, "Spoken Arabic Digits Recognition Based on (GMM) for E-Quran Voice Browsing: Application for Blind Category," *Proc. - 2013 Taibah Univ. Int. Conf. Adv. Inf. Technol. Holy Quran Its Sci. NOORIC 2013*, pp. 123–127, 2015.

[6] F. Rehman, C. Kumar, and S. Kumar, "VQ Based Comparative Analysis of MFCC and BFCC Speaker Recognition System," pp. 28–32, 2017.

[7] K. V. K. Kishore, S. Sharrefaunnisa, and S. Venkatramaphanikumar, "An efficient text dependent speaker recognition using fusion of MFCC and SBC," *2015 1st Int. Conf. Futur. Trends Comput. Anal. Knowl. Manag. ABLAZE 2015*, no. Ablaze, pp. 18–22, 2015.

[8] W. Astuti and E. B. W. Riyandwita, "Intelligent automatic starting engine based on voice recognition system," *Proc. - 14th IEEE Student Conf. Res. Dev. Adv. Technol. Humanit. SCOReD 2016*, 2017.

[9] F. G. Barbosa, W. Lu, and S. Silva, "Automatic Voice Recognition System based on Multiple Support Vector Machines and Mel-Frequency Cepstral Coefficients," pp. 668–673, 2015.

[10] T. Barbu and M. Costin, "A Text-dependent Voice Recognition Approach Using the Spectral Distance," pp. 2–5, 2009.

[11] D. Hardt and K. Fellbaum, "Spectral subtraction and rasta-filtering in text-dependent hmm-based speaker verification," pp. 867–870, 1997.

[12] M. H. Martínez, A. Lorena, A. Blanco, A. María, and G. Palacios, "Reconocimiento de patrones de voz para fines acústicos forenses," vol. 9, no. 17, pp. 37–44, 2014.

[13] S. M. Doubert G., "Segmentación y Realce de Señales de Voz Usando la Transformada Wavelet y DSP's," p. 78, 2004.

[14] G. Fant, “Acoustic Theory of Speech Production,” *Slav. East Eur. J.*, vol. 5, no. 3, p. 285, 1960.

[15] “software libre.” [Online]. Available: https://www.gnu.org/philosophy/free-sw.es.html. [Accessed: 26-May-2019].

[16] “FrontPage - Raspbian.” [Online]. Available: https://www.raspbian.org/. [Accessed: 26-May-2019].

[17] “About Python™ | Python.org.” [Online]. Available: https://www.python.org/about/. [Accessed: 26-May-2019].

[18] A. Larios, “Reconocimiento y síntesis de voz,” pp. 1–9, 1999.

[19] C. Esteve, “Reconocimiento de locutor dependiente de texto mediante apatación de modelos ocultos de Markov fonéticos,” 2007.

[20] A. T. Rusli, M. I. Ahmad, and M. Z. Ilyas, “Improving speaker verification using MFCC order,” *Proc. 2016 Int. Conf. Robot. Autom. Sci. ICORAS 2016*, 2017.

[21] R. Gutierrez-Osuna, “Introduction to Speech Processing, L9: Cepstral analysis,” *{…} Handb. Speech Process.*, 2008.

[22] H. D. Barrob and M. R. Costa-juss, “Reconocimiento automático del habla.”

[23] G. A. Martínez Mascorro and G. Aguilar Torres, “Reconocimiento de voz basado en MFCC, SBC y Espectrogramas,” *Ingenius*, no. 10, pp. 12–20, 2013.

[24] G. Mascorro and G. Torres, “Sistema para identificación de hablantes robusto a cambios en la voz,” *Ingenius.Ups.Edu.Ec*, pp. 45–53, 2012.

[25] INTERNATIONAL ORGANIZATION FOR STANDARDIZATION), “NORMA TÉCNICA NTC-ISO/IEC COLOMBIANA 27001 Requisitos Ntc-Iso/Iec 27001,” p. 37, 2013.

[26] Manhattan-products, “Adaptador de sonido 3-D 7.1 USB de alta velocidad.” pp. 1–3, 2006.

[27] A. Prentice, “Raspberry pi,” *Annu. Rev. Nutr.*, vol. 20, no. 1, pp. 249–272, 2000.

10. ANEXOS

Coeficientes cepstrales palabra 1												
-4.9279406	41.3512699	-25.977916	27.1443917	13.9719788	20.316928	6.37599635	1.03914087	-4.7558293	-5.1386027	4.00613539	-2.6167969	-0.913002
-4.3843479	52.2479048	-36.200272	34.196888	4.26431742	28.205261	-5.6370923	4.83011077	-14.181034	-0.41458	-0.6340742	1.88990289	-9.6679212
-3.6085011	50.1965999	-51.58355	45.8764088	3.47947328	23.2131794	0.4149522	2.61187919	-9.3579415	-7.4632949	-1.2736427	5.58662367	-11.212748
-1.4741808	45.583352	-55.177105	24.3414217	14.2937573	22.7547201	-8.5364628	-27.44461	14.162363	-17.205768	13.5836858	20.3787978	-1.5103969
-0.3477364	52.0953345	-65.35427	26.3994426	1.49446051	44.9091747	-31.379033	-35.923866	28.3328372	-21.97202	14.7980479	33.1923142	-6.8416681
-0.0304727	49.7857066	-64.773057	22.7853232	4.49972787	41.6937872	-33.690678	-39.114781	32.4895809	-20.998676	18.4898237	31.7376196	-11.569923
0.15792663	43.7094107	-58.899694	17.1714856	11.4681774	37.138669	-32.668797	-48.903532	36.0942323	-24.040928	21.4409617	35.7535367	-13.552363
0.25198981	30.6315804	-40.504126	-3.6315062	30.0183315	24.6064306	-35.541166	-36.046191	19.2127403	-8.7109397	19.9814892	29.2113296	-11.821736
0.26436679	42.2306061	-53.595269	8.89691417	22.7169665	28.9879868	-36.453913	-41.140447	25.744757	-13.235565	24.2415541	29.2649294	-13.725029
0.21997763	42.6627168	-52.159416	7.33594861	27.5776392	28.5555218	-39.139664	-42.533387	28.1337031	-14.225177	28.1526935	33.748679	-21.516101
0.12667268	50.549735	-62.612779	22.0458457	15.6660642	38.6219193	-46.183397	-39.867126	28.6726223	-18.15228	30.5984846	28.7890488	-24.59416
0.06935465	47.2746418	-56.904422	12.7568906	25.5840478	31.4747856	-39.93093	-39.518125	24.4926262	-17.134234	25.6454673	24.256366	-19.376017
-0.1313815	48.0048254	-58.940991	21.5077977	23.8287781	31.6801867	-37.273674	-44.841217	20.2019922	-11.160049	19.7340158	18.2558847	-15.13707
-0.3618801	38.491193	-47.755059	8.87448732	34.6391037	24.0026948	-25.212968	-36.667911	15.6135343	-15.835951	2.2458788	16.4099122	-13.926107
-0.7979656	45.2458871	-53.539891	19.6363597	28.6832524	27.346667	-19.288637	-40.432747	13.4229885	-21.403886	-4.1256846	7.6023873	-10.688028
-1.7568274	44.2566283	-56.088112	28.7984727	22.3716978	23.3540096	-10.368536	-31.707069	1.61877899	-21.273782	-0.3743645	5.54298963	-7.3237442
-2.6135938	30.6612321	-49.005103	33.0859089	30.8685614	-6.8937412	10.9010216	-10.008528	-18.115822	-5.7055505	14.7057039	0.19998882	-17.988785
-3.1941636	40.946037	-57.398858	41.9513577	29.522544	16.3468359	8.59496903	-14.633906	-22.540445	-14.092062	0.92819081	-0.736703	-22.493063
-3.708139	41.8538242	-59.85812	47.1622574	20.0677922	14.3977629	6.30066602	3.62225791	-19.706786	-11.627959	-1.6335315	-3.9466447	-16.095544
-4.3228688	39.9972067	-51.853926	51.1226985	10.9519357	8.45813263	12.7948017	-0.3831505	-13.940327	-18.126361	2.85678622	4.72579229	-17.03864
-4.8300681	31.9451209	-39.196997	47.3278141	11.4849977	5.87271587	18.3922559	-7.0301836	-11.35881	-23.906108	1.47701255	8.02643498	-10.286506
-4.856327	33.3639699	-52.010828	54.2228427	-2.2790349	15.0390558	17.713643	8.15828458	-10.958255	-23.949688	-2.2167751	14.032256	-23.390341
-5.3329134	25.7600814	-48.062261	45.2794169	-2.7791075	3.07794334	19.6749836	-3.6175197	1.287067	-31.688951	-10.555041	11.1011084	-23.908642
-5.0816139	24.6985279	-49.746447	37.0213121	9.02186507	-0.0076833	-4.8560976	-5.1342216	10.1297301	-30.351177	-20.499985	9.8956781	-15.47867
-3.2360986	19.5845208	-60.520036	50.0271432	6.09422152	2.80098059	-13.482858	8.14604999	20.0977021	-15.394801	-23.143796	13.7487035	-16.521226
-2.9385403	4.69935522	-51.923101	53.0913264	-13.512849	-0.8485225	-8.147694	-11.044639	3.08126704	-22.870596	-22.435787	13.9974365	-7.9205387
-2.4106648	-0.4736421	-53.930589	57.6703402	-9.6954008	0.25965283	-3.4683672	-5.0010793	12.3970539	-16.428445	-27.120923	9.85458143	-2.7682428
-1.7671012	-2.1951767	-61.043796	51.791879	-8.9168815	-2.5666489	-7.2084338	-19.642622	8.53721904	-13.933215	-19.048785	11.9801414	-6.9814327
-2.1067441	8.72138604	-66.515399	47.0417828	-17.135221	-0.5771691	-8.9022075	-14.453123	7.22742067	-17.49189	-20.08161	22.0449151	2.78798427
-3.4448002	6.06672641	-54.711258	34.6088742	-0.40544	-1.6228532	9.27742594	-14.415128	12.0315893	-20.083331	-11.770005	12.9722088	1.52281216
-4.5231007	19.6801997	-62.018318	34.7523668	-11.615162	6.50332441	-1.1279322	-20.787656	4.41236676	-25.528394	-13.631377	6.45207064	-1.0891735

Anexo 1. Coeficientes cepstrales Referencia 1.

Coeficientes cepstrales palabra 2												
-4.2590718	34.6141821	-24.276376	24.2325893	15.3294501	9.95805582	10.6129192	-0.9998486	8.93253137	-0.17177	0.72828694	-3.6044754	-3.4936049
-3.7631745	42.7124791	-34.029112	35.3731686	6.5234845	21.3208439	-4.061395	3.90156084	0.34328603	1.46987103	-3.2840402	-0.673613	-7.5439777
-3.4730588	44.3143819	-35.658296	35.6781288	3.46377721	24.0380638	1.12324599	11.6939289	-2.2330892	6.13112417	-3.8310341	4.78721193	-4.4255602
-3.1760654	44.6342091	-39.683169	38.6719566	2.77147778	4.75302186	3.36374826	19.4398878	0.87435396	-10.242443	-2.9251491	13.50863	-5.555346
-1.8511649	42.3265198	-49.479115	28.5051929	6.66318511	10.7908167	0.97967965	-9.4199714	3.9739262	-16.6374	6.22038463	25.4718515	-1.104012
-0.9980426	40.6877041	-54.107044	26.7343374	4.56194291	14.1528941	-4.3024377	-18.90667	7.55239238	-14.156773	13.8041777	24.6552073	5.833459
-0.8950622	44.9819349	-58.08277	32.032846	-3.7769872	24.4772411	-7.736302	-17.819365	7.75141673	-17.177752	11.0703619	25.5769753	2.71777304
-0.943531	40.3899732	-51.716797	25.7610786	3.98610903	23.0062846	-2.5750589	-24.184497	10.7429013	-18.478955	16.2018664	43.9625659	2.86816928
-0.8592931	45.6870508	-54.36858	33.0210109	-8.2486553	27.0832132	-12.012969	-24.574024	7.95087835	-18.733078	10.222235	43.7355351	-0.798332
0.21568374	46.9453412	-62.48607	25.6453262	2.00808239	28.0093045	-17.956735	-16.884155	12.418451	-27.0136	11.447772	44.5909608	-3.7687096
0.21655662	29.7007063	-38.941265	1.96964432	26.3046992	14.2453917	-14.206445	-15.463992	6.99889019	-22.213174	6.81741749	39.7065433	3.95600074
-0.0124759	39.443348	-51.334475	18.1393687	18.6828558	20.9523436	-13.851138	-22.533755	18.1133093	-32.967972	5.60967163	41.4126585	-1.2420927
-0.573886	39.9022516	-49.173333	22.5398159	21.6554835	25.9062048	-16.29591	-25.444136	16.4714225	-35.191686	5.86587872	32.2909384	-9.121205
-1.7157043	42.4540417	-47.117604	29.5097746	16.2357317	34.0169146	-11.31658	-17.427675	8.10072536	-36.561084	0.8855736	29.2530869	-6.893452
-3.6375408	36.9857692	-40.684993	32.4710475	9.97103764	13.7938897	8.48232237	-0.0205545	0.36480253	-13.222065	-2.3778606	7.52286354	2.85652324
-3.4503958	33.9118346	-54.966667	41.4651954	-4.6559131	-0.1499796	4.13108117	-3.4651269	-7.6863738	-6.449018	-2.1733422	0.69255753	-15.80332
-2.6877947	33.8675598	-56.169136	48.327784	-1.1954599	-0.1961852	7.19947901	4.53440588	-13.259596	-7.40064	1.26451012	0.08973688	-24.431493
-3.3809678	27.9636375	-47.076381	38.1048245	15.7710556	-3.0700504	13.9398774	2.52377036	-7.8943085	-7.1016738	6.06997496	-1.2548841	-17.028529
-3.8168548	28.3825803	-53.188784	35.0542499	4.78165699	-1.3058479	10.8002747	-9.1372241	-14.139166	-4.5497268	4.26320646	3.07513637	-14.406481
-3.7305578	27.3677238	-54.005341	42.6223324	-6.5378193	-1.5567287	-0.333841	1.612162	-9.6332362	-2.270614	0.17716213	4.96693925	-13.858743
-3.4660421	24.6447157	-56.632933	43.2680154	-5.5425336	-2.4500058	4.03339159	-2.0311328	0.80398795	-12.552665	1.83351616	17.8077674	-16.273785
-3.8910208	27.7665204	-56.864088	38.401255	-14.479271	2.6853019	-6.4811259	-2.2327029	-2.4106324	-15.227518	-10.623711	20.7323379	-14.599858
-3.911737	21.2876053	-46.414657	31.6264909	0.42514155	-3.0899043	-7.1231136	0.68292404	8.4328944	-18.997574	-9.183492	6.34157917	-2.3371878
-1.5646856	9.12642412	-56.83621	53.7367595	-18.192219	3.04017371	-21.479867	13.2217087	8.01952178	-16.451913	-17.840438	18.2545667	5.16240118
-1.0219993	2.03573703	-53.122246	45.8546432	-9.1610702	-13.397073	-10.059912	-6.8234972	9.03345569	-23.636471	-16.163003	7.77396972	5.29561113
0.75396209	5.92176423	-67.4452	46.8536359	0.31922928	0.99714744	-20.984929	-6.8179153	18.4052974	-18.061469	-29.670547	2.00249889	13.7444943
-0.3397631	12.1708141	-66.327117	52.0492843	-12.72389	4.19862129	-23.331877	-16.503594	18.3920465	-17.22661	-23.901804	3.87721312	7.3187613
-3.0249517	12.734414	-46.828353	29.2511218	-1.0526874	-2.5193251	3.16862215	-16.944043	7.74349892	-18.700448	-6.8865446	8.48884111	0.5941298
-3.7207559	24.2544375	-48.696014	27.0409357	-3.9057939	-0.7729299	-7.5660216	-16.969896	11.4547875	-24.01733	-14.016301	10.258272	7.27924489
-3.7861644	33.9338139	-46.143395	29.6646788	-10.194051	18.7928868	-0.7813406	-6.8968673	17.1054598	-27.400387	-9.4215247	7.07657782	-1.7768521
-3.3619148	41.4438481	-42.450011	29.1511039	-21.087341	13.0992338	-11.631938	-8.273747	8.59536194	-22.091609	-11.997843	9.73080419	-0.5559395

Anexo 2. Coeficientes cepstrales Referencia 2.

Coeficientes cepstrales palabra 3												
-5.064011	36.8321361	-28.222866	29.3888725	7.73138404	26.6074651	7.80889233	1.5848219	-3.889763	-6.9576579	-5.7896284	-4.1675569	-0.1618771
-4.7115044	42.3163987	-34.615299	31.1081419	4.83315406	29.4099983	0.23698568	2.19264043	2.12136612	-2.6408226	0.15431533	-5.0461	-13.747502
-4.434366	38.9861936	-31.869918	30.4229573	10.8840875	29.6163024	10.4315364	-4.7998034	-0.4685164	-6.289924	-0.1881256	-3.7846206	-6.9117204
-4.1875123	45.1087848	-39.664227	36.7579759	-11.206236	34.8690162	-0.6767371	11.5748693	4.71233305	-7.270906	-1.9737081	2.90836878	-20.570195
-4.2697793	46.2361055	-36.648315	32.0194828	-3.5724414	31.4971846	3.24024373	1.85281244	-3.9096847	-1.0007894	-5.6982745	-0.7720791	-9.4330022
-3.427201	45.4435535	-47.127764	32.8545555	-2.762645	14.8338781	2.67206332	0.86032669	-8.520655	-19.015981	2.70068385	11.9316079	-6.9941801
-2.3433503	40.9194382	-42.125385	17.7657589	4.52441809	10.2898783	7.5507645	-19.074395	-0.8582158	-20.603876	6.3057014	25.2127363	2.56429028
-1.7968101	40.3029471	-39.487057	11.4157939	0.11635911	22.9751024	-5.2052246	-32.80267	17.2499124	-16.216094	10.6241153	34.6218462	1.14106626
-1.8291616	45.1697644	-47.658177	21.2851549	-8.7352634	28.78899	-11.998605	-28.899193	23.7596979	-20.001169	14.3407456	31.1814132	0.45923399
-1.8874793	44.4444157	-42.02479	13.9975806	-6.4240523	35.6071601	-14.948659	-34.604294	27.2552252	-16.359732	10.9443878	21.4691668	-2.478812
-1.7103307	39.6572265	-37.477776	6.89425732	-1.932355	30.7750299	-5.9012195	-39.901048	24.2507505	-4.4127692	4.24553726	22.6643041	5.14695052
-1.8657305	39.0729214	-37.919321	10.1356967	-0.9825553	29.702619	-7.6663892	-39.909882	22.4490592	-9.6522984	10.318772	24.2867697	2.06405645
-1.9350357	46.7841819	-47.874944	29.2932136	-11.810074	41.1475337	-18.072634	-39.221973	14.9526266	-1.5783386	-4.0257522	30.178663	-2.3685471
-2.2486432	43.9460068	-41.024712	24.1040787	-7.9494952	35.9867768	-5.7447387	-45.11444	22.0513027	-9.3899888	-0.4584626	17.6177347	-0.2294322
-3.0444732	41.2682753	-35.978152	26.1681602	5.01841563	29.8543682	-7.028357	-31.348513	17.2583497	-19.977568	4.59325901	19.2099229	-6.5758343
-4.1629238	43.2217232	-33.900018	29.626151	2.96571569	27.0236343	3.35703333	-5.4446976	-6.2306608	-17.476067	4.80854538	15.1375971	-7.8839874
-4.9263447	29.9439261	-24.701904	19.0511559	13.2409301	1.0003756	12.2718775	-5.1505768	-11.406758	-6.0495829	4.10514507	2.23273489	-3.1680713
-2.8285304	28.4821623	-49.610943	36.322798	15.0265653	-8.3581168	14.9864536	-4.8736932	-20.048156	-9.5984232	14.1376558	-6.6762685	-21.325531
-2.8240488	29.7729006	-53.713251	53.795796	7.77790019	1.76438105	22.8572798	-7.2713111	-12.726694	-10.818687	16.4803946	-1.3455791	-26.352964
-3.4681904	33.2406233	-53.007871	55.7632457	8.19393991	9.85263823	15.9408884	-8.3048195	-15.790356	-6.2596515	6.42121536	-1.3716892	-24.389073
-3.8060057	30.3013113	-54.305798	52.0994054	4.16394844	8.70232757	14.0649267	1.98469504	-14.684812	-7.9628838	9.33317606	3.51205448	-11.906246
-3.9861974	26.6389456	-58.958132	43.6045531	1.91432849	1.71652076	10.5458221	-12.073093	-9.9881305	1.65886659	8.31591051	2.4839932	-4.1943361
-3.4565803	25.2313158	-70.874502	47.8885765	-12.823617	4.9900521	-8.842681	-5.2779451	-17.91337	-20.409987	-5.6822632	17.3214288	-10.814215
-3.5377274	18.8526335	-58.455369	39.5810182	-4.3018629	-9.37743	2.8302295	-2.1513605	6.40224909	-31.89922	-1.757583	15.9027673	-3.219987
-4.6159827	17.6435853	-46.933264	32.0969068	-6.7978251	-4.1358515	1.37376942	-15.679112	8.31442615	-21.332579	-13.213047	16.5979502	-6.6092799
-2.7359557	7.38668246	-52.269335	55.1142452	-18.001817	3.52332114	-2.1971219	-5.6508857	-3.3658245	-10.369052	-18.714607	15.2673222	1.10291307
-1.1625686	-0.241201	-59.489608	62.3168792	-16.513126	4.24857769	-13.161099	-3.1079613	6.22295244	-2.2050751	-29.924458	14.3929484	-3.8667539
-0.3175029	-3.2773706	-61.437956	58.7581283	-13.219811	0.61303203	-11.623218	-15.029228	12.0528062	-17.638626	-43.987554	14.0258326	4.42305594
-0.2381267	-1.5790183	-62.905988	51.97221	-5.048746	0.78756914	-3.8977677	-17.527682	18.4256957	-13.345104	-25.597624	19.05031	9.89878214
0.03502796	5.64721522	-71.739182	44.6828401	-16.864313	-4.0140187	-8.8123152	-19.88894	18.4082012	-12.072505	-20.941206	15.2734241	1.55455338
-1.9477196	11.5890639	-65.259899	41.231027	-14.569421	-0.6353515	-3.9092081	-17.48454	10.3141581	-17.298363	-15.753347	-0.089826	-3.7754247

Anexo 3. Coeficientes cepstrales Referencia 3.

Coeficientes cepstrales palabra 4												
-4.3017938	30.6922886	-21.180691	16.6704479	16.3592443	10.2197466	13.0156443	-6.093686	2.77494877	-4.5342794	-3.246966	3.52300871	4.65756604
-3.8131581	42.4650361	-35.982503	35.4147982	-0.8967863	32.7070608	-4.6939538	-0.5515438	-1.8080285	0.18059418	-8.4959533	2.96791838	-6.2073137
-3.1640362	42.2354568	-38.249405	29.9017218	7.4289043	29.5886943	-2.491071	1.40765757	8.89160937	-4.4633249	-3.6651416	2.42837214	-9.1065198
-3.0175999	43.8933359	-43.980778	36.5211522	4.55357819	33.0371025	1.36478524	6.43302874	4.78026209	-6.1750618	-12.334227	2.46890786	-14.42529
-2.7870044	40.5901443	-42.883109	31.4568679	4.74439499	17.9239779	11.4462503	-1.7646442	14.7067674	-19.798414	-10.105099	3.25690742	-9.6147502
-1.8285232	45.2204005	-54.426979	33.5077275	-1.4319014	23.9755323	-13.781118	10.6994735	-3.4761956	-10.611022	-13.966133	26.6621331	-4.5963761
-1.3076181	42.6235633	-52.54705	28.3627439	5.92438525	23.4729754	-8.5294573	-1.0406743	4.29015336	-12.771695	-10.058823	25.0249661	-5.6086012
-1.2638847	39.0485132	-48.659964	30.305381	4.8635964	24.2644399	-7.0970677	-9.7069714	6.19631275	-13.912087	-9.8350212	22.6119119	-2.3968005
-1.4285616	44.7705153	-52.052396	35.5824782	-2.9725765	32.2622784	-15.486767	-5.9344129	-3.6346671	-6.2648441	-11.975966	33.3451612	-11.273919
-1.3861683	42.3056284	-49.293199	32.3426095	1.37993864	28.3201201	-11.656451	-14.056738	8.06679293	-19.296579	-2.32091	24.9983889	-11.330772
-1.3879246	40.0684993	-46.432377	30.5138451	5.98346294	28.7120555	-10.968352	-18.336562	13.2320016	-25.001279	-2.6386641	20.6245381	-14.194011
-1.8230063	44.2256046	-46.444305	35.758289	5.53620144	26.8676867	-15.706757	-9.8150912	5.48895505	-26.691939	-1.6776934	21.6736543	-13.913107
-2.4993458	36.2391161	-32.873399	29.1026266	16.9261415	17.0341897	-1.6231573	-13.033911	6.27313643	-31.476394	-5.9231773	19.6967243	-11.798622
-3.1942285	42.596679	-44.777742	38.2566083	7.84800571	23.9914458	4.62293619	15.810059	-12.232132	-12.224183	-6.3552936	13.7636355	-15.39491
-3.0804058	42.7285531	-48.709365	38.5802717	1.07168584	18.98647	11.5331155	-8.3437834	-23.832412	14.0437323	4.09666557	-7.8768833	-16.918386
-3.6975978	40.5718698	-42.928389	36.5679612	1.97414689	19.3846834	0.36119663	-1.3028499	-19.238473	-3.4593474	-7.4185858	-3.5283871	-18.062929
-4.2905919	36.3026007	-43.771889	36.2512336	7.87862767	9.01553666	1.03925913	4.25992684	-12.743658	-7.4214398	6.77394958	-3.0040082	-11.820012
-3.9123679	32.6197119	-51.579947	40.8701834	1.85019123	6.43296511	-3.6113233	5.17515805	-8.4827087	-12.252685	-4.6002044	5.68582956	-10.77184
-3.7889312	28.2795404	-54.957711	44.8963816	-8.4008468	5.09691765	-3.5552596	-0.7191335	-5.9895363	-23.694189	-3.3596456	5.39966254	-7.9983955
-3.1016835	22.8331996	-53.946983	46.1935707	-7.0806909	-3.2377601	-5.2283473	-7.506727	11.8204255	-29.759488	-16.403549	12.748321	-4.8426392
-3.6761292	26.2079098	-56.820633	40.5753048	-10.568024	8.30961877	-12.833405	-19.581715	-0.3812418	-20.593977	-23.793699	7.37477569	3.93070857
-2.4049275	25.7374674	-60.676594	39.9148033	3.27672366	10.8104302	-10.511256	-16.447886	7.52389612	-1.4345585	-23.829403	10.6057514	10.9409777
-1.7738098	9.60433026	-55.96348	47.5818741	-11.83109	-4.516807	-4.1624384	-12.942277	3.21483753	-6.3899227	-22.649514	5.99558525	11.6001211
-1.2228669	9.16917938	-61.411706	53.7235346	-11.360487	3.67697005	-11.537959	-14.672105	0.63570573	-4.3848083	-26.333748	5.40709071	10.3170128
-0.5861215	12.8057213	-68.24565	54.1662374	-20.031469	1.24842289	-14.612027	-3.8984399	9.27910062	2.09302422	-23.753651	20.1790909	11.0962579
-1.7983409	14.7233621	-60.780351	33.0981676	-8.4285959	2.24640415	-13.432797	-20.479965	18.2424906	-18.353682	-11.768447	12.1848819	2.97152796
-3.571886	27.3019665	-57.097727	37.5154276	-16.4373	11.9061475	-5.5513007	-15.213667	12.5977193	-21.117121	-12.718639	8.14160153	-5.2002886
-4.3306521	31.8430877	-50.376904	27.6885098	-14.648658	16.5513402	-5.3332609	-15.108126	6.4228537	-10.564082	-5.308484	4.6055673	-6.9397797
-4.3764923	35.2353801	-36.550673	25.8311713	-6.3355332	16.4216103	-13.041396	-16.330582	1.82444194	-5.5694467	-4.1068308	5.78044332	6.98769346
-3.3793638	44.9644502	-36.021103	26.9970666	-18.422292	13.3379006	-13.424913	-10.794519	5.24587669	-13.814558	-3.4804867	5.77081194	-0.7465397
-2.3246981	46.490044	-36.366444	14.0372607	-14.928431	11.0099652	-12.127343	-19.746529	14.6638974	-15.311332	-10.866221	0.8075004	-0.6896146

Anexo 4. Coeficientes cepstrales Referencia 4.

Coeficientes cepstrales palabra 5												
-3.9070316	26.6925689	-8.523474	9.3254578	25.6183929	9.55846358	4.78854603	-5.3251017	-2.427344	-4.8049596	0.29099886	1.2956196	-5.9352526
-3.7392453	40.1548855	-27.545317	23.8589789	9.87941708	21.5938921	5.12785572	-2.4151426	7.67682199	-13.588356	5.60537268	1.5419823	-10.498395
-3.6362288	43.106496	-36.224901	22.5853777	6.55100591	24.0399123	2.57773928	-1.4611156	9.31180159	-3.5485722	13.0480576	1.5669358	-15.423431
-3.5052123	48.3041415	-42.566408	29.050364	-4.1968606	28.7625868	-12.424423	1.01334356	4.59024628	-0.3797448	-1.1436008	-1.7628334	-19.729324
-3.4393944	42.0940848	-41.417203	29.6605755	5.41762144	15.0187426	9.25023674	0.67262406	0.58920238	-9.7366636	4.37910246	0.74840725	-19.189503
-1.819276	43.1109376	-47.418609	26.8802598	13.6836175	6.79427832	-0.4383359	-9.6981661	1.87698429	-22.106732	7.07484684	26.907591	-5.9435615
-0.6427315	44.1710047	-52.768418	20.1423311	6.13916193	23.7662058	-6.225904	-30.539225	10.0225384	-21.710627	11.5224372	35.4510832	-3.8129031
-1.3878957	45.6874962	-49.798015	35.2354465	-15.17505	30.4314312	-11.154389	-37.460373	9.12450567	-11.04902	7.8162116	29.186574	-2.2450798
-0.9879086	36.3076076	-37.118084	17.754898	2.08140138	19.5705591	-2.4276646	-49.976362	5.09795801	-3.1153449	11.8731913	27.2005786	-1.355541
-0.9919347	40.8904518	-44.812881	25.1977027	0.6779989	22.9195263	-4.7778878	-46.338411	4.96356335	-9.7795767	13.0345037	25.6734579	3.28807479
-0.8952062	34.9799556	-34.736314	13.4340326	16.1252446	21.2002209	-13.797976	-47.548904	9.21136933	-8.5875825	0.80836688	33.7663282	5.12463953
-0.8336073	44.3633326	-53.120559	32.760791	-0.2157632	31.6122007	-14.250091	-38.198484	4.8880258	-26.277758	4.33715317	34.8474501	-1.1619945
-1.0315405	46.2961675	-50.558758	30.9054833	5.75225966	37.1368227	-25.931775	-34.889351	4.22133778	-29.20143	0.23803414	35.4039787	-0.2051604
-1.8990178	46.2482662	-52.772074	42.2303432	-3.439619	32.2220252	-16.815991	-17.770608	-15.130805	-8.8888323	-9.9301678	19.7264394	0.66093943
-2.9480774	31.8066397	-42.754215	40.181764	9.80851003	1.68417617	15.382938	-26.784788	-14.15373	-6.5264047	3.99461272	6.60565768	-3.951793
-3.0045428	38.4693261	-47.429779	39.5188761	7.88676607	12.1567757	5.08258127	-20.197832	-8.1076095	-6.6741226	2.97119688	-6.2758025	-16.541136
-3.2522006	33.8289156	-42.98845	32.2746379	18.0292978	11.6071314	12.0853994	-14.089519	-9.7707831	-10.355088	0.260667	-2.3479912	-15.38503
-3.7328153	37.9101039	-48.524724	45.2101009	5.52797273	19.4865902	10.8294996	-8.7146666	-13.79135	-11.291211	-0.0695677	4.11840746	-15.054904
-3.615289	35.0625578	-52.470721	49.6334927	3.44973902	12.6682053	10.3875353	-3.4510558	-26.736373	-5.1675615	4.68970059	-0.446311	-12.378564
-3.4082501	28.1974226	-58.433978	44.5514606	5.01752518	10.9169636	10.7192881	-13.02443	-8.3085971	5.93919422	17.5549158	12.2598602	-12.848612
-3.3215847	27.5592771	-62.128052	45.646406	-8.9269064	14.89574	-0.7050303	-2.5786711	-14.689162	-9.7602503	7.64355609	20.7540036	-15.406255
-3.4406986	21.5583787	-56.574368	36.1271813	-5.188552	3.86595115	2.3732514	-6.7697163	-1.7673119	-28.081967	-1.1791211	9.87198022	-6.7867136
-4.3831907	26.6424565	-53.427463	40.1410808	-13.265812	9.57454378	-4.8931257	0.59178274	-0.3667508	-22.78456	-4.2240111	15.3527721	-8.9627171
-3.2157347	21.4796576	-48.19133	21.8481156	6.07650068	-10.369271	-7.2884982	-2.3000877	13.8465135	-28.461187	-12.591082	13.4506448	-5.457031
-2.2096128	23.5687001	-62.451754	29.2137742	7.90820468	-4.3710947	-11.662719	-5.6294718	14.4076373	-17.588884	-17.91273	6.72666821	-0.40028
-2.2458239	11.3723847	-57.759647	46.7240643	-6.2177333	-9.6286776	-10.136082	-19.814007	4.13700462	-15.081391	-32.487526	12.2719691	1.70026214
-0.975805	0.79564114	-54.686272	60.8450601	-12.662224	3.51357258	-8.0651179	-15.375996	6.20002062	-7.2405246	-31.66777	16.8720461	0.50926897
-0.5061668	-0.8605965	-56.657264	58.3211892	-16.798403	3.37242289	-14.776623	-10.113412	12.7772743	-4.9009562	-34.29725	14.6920457	1.0269103
-0.1862393	8.64099183	-66.718729	54.8227441	-30.348402	3.79737665	-28.282048	-8.351413	5.48401237	-0.3527182	-27.198389	11.2129575	-2.391836
-0.4561369	4.85559205	-62.407416	45.3629662	-7.3576197	1.61710887	-9.0935986	-22.124767	23.0229054	-7.6294527	-15.098484	5.05490119	9.65093614
-2.2329752	11.7403736	-54.247387	31.292371	-2.7159178	-4.964288	-1.6069801	-23.453789	8.07811233	-25.537221	-16.727712	16.8869813	10.228624

Anexo 5. Coeficientes cepstrales Referencia 5.

Coeficientes cepstrales palabra 6												
-4.2911318	30.0745307	-15.339518	18.3674616	27.2904292	7.25078901	3.33419578	-6.2783753	4.07240944	-0.5162225	-1.7974674	-2.1724304	-5.340939
-3.5524643	41.4694571	-32.279201	20.6828267	18.2295338	14.8001336	0.34892282	-3.0517932	10.8887697	-2.5882429	3.92296555	-1.8806941	-12.45875
-3.277209	45.0355327	-43.186127	30.8672113	12.5415061	31.4664284	-3.4926189	6.18974616	5.32645562	-6.1435303	-3.7502024	-13.800221	-23.422824
-2.8334629	41.8449924	-42.225642	28.7246126	13.5348656	25.9562782	-4.9504662	-8.9427075	10.8773949	-21.637484	1.94168706	-2.2213225	-23.398211
-2.6189779	45.7933237	-47.602591	33.2973677	7.76171713	30.6148626	-15.346317	2.36459276	4.55109099	-11.261494	1.7677393	0.16604422	-24.364108
-2.5696019	38.9353939	-37.073508	17.1012937	18.904409	18.8210622	5.19980885	-8.5198145	7.30967012	-15.207249	4.61598207	8.98671071	-11.861454
-0.8469043	39.377729	-50.683744	18.1118105	19.1231551	14.6599395	-0.2609083	-17.100807	9.09266324	-25.543515	1.79606856	31.0873428	-3.3277518
0.09733175	36.5391781	-55.269915	22.4886692	23.4942344	13.5155574	-6.7436512	-28.607184	6.2296862	-21.83292	6.79469834	34.2067257	-3.5592893
-0.1068703	34.8325139	-48.116926	17.208624	24.7656802	20.0002947	-12.366475	-30.091963	11.1257319	-22.925273	5.86298915	35.7947187	-5.5795959
0.27592356	26.6407688	-41.269849	11.7751548	34.5802858	15.0492134	-14.357713	-27.374536	1.89953336	-15.611795	3.09520804	34.3983215	-2.5832843
0.62440223	38.0289298	-58.915532	29.6224623	23.4085158	23.0715114	-22.862249	-35.808342	11.9193128	-29.88732	13.2309719	35.417166	-5.1566724
0.70442836	43.9594726	-69.964961	44.35657	11.0501516	35.7204935	-33.526283	-28.260314	6.9389801	-25.903051	7.85384692	37.9783015	-10.755594
0.17345975	36.9996352	-55.332239	28.806625	34.5423908	21.3706973	-22.231491	-33.495666	7.14799126	-29.214752	8.4944233	36.7120845	-3.7402886
-0.3752405	45.2145485	-65.294135	44.2122493	17.3725387	33.9953873	-31.173611	-19.241925	1.81191115	-28.357438	-3.2716681	43.4413794	-11.225834
-2.5517967	42.9698479	-50.184045	37.1153849	11.1063819	33.8070572	-15.754386	0.94979496	2.38580828	-26.860259	-7.3786398	34.6749083	-3.6247818
-3.7320861	36.7525213	-39.997994	31.3958776	10.4678631	12.9904586	11.5570242	-12.977086	-17.736105	-13.762124	9.95201273	0.8448342	-9.6977505
-3.0522756	30.3908712	-49.695144	38.4594453	8.8032261	11.3780874	13.7850309	-21.469583	-12.626098	-12.248455	16.0244258	-5.0058246	-25.019402
-3.5474325	28.0412657	-45.997207	35.896064	14.4731026	2.73078397	11.7356662	-8.6618843	-8.5028735	0.01539046	18.9662401	-6.1336894	-21.188712
-3.152912	30.4330154	-57.635915	45.2998571	7.65231512	7.65180452	7.59040819	-1.2318779	-7.7704424	5.51492367	24.9861339	4.57993585	-26.932521
-3.4556662	24.4749554	-56.026212	35.870308	3.7679757	4.17176866	10.9188246	-7.2336762	-5.8896071	-10.144365	12.3794929	11.7053839	-15.674157
-3.777218	26.8972602	-57.346585	39.2377354	-8.5704007	3.74634258	0.27651209	2.45591798	-13.560572	-15.402231	-8.308632	14.8032117	-11.640659
-4.7385183	24.4793572	-49.45273	37.4901056	-9.6315773	-1.4393469	-5.5325146	-7.0947589	-4.6129172	-9.6790778	-4.0201798	11.761491	-14.887923
-5.1691035	26.3677201	-45.495814	32.1102227	-12.655582	2.32612082	-4.8842197	-10.559654	-3.5265934	-11.367907	-5.4691454	7.43211655	-9.8580665
-3.89623	21.5663493	-50.349753	22.5923347	3.80897253	-8.2129219	-4.0052554	-11.348732	9.12075864	-14.99871	-16.771424	6.85141721	-6.1637416
-1.4708309	13.9238557	-65.035954	31.2576157	-8.2994872	-8.1839128	-1.7206046	-18.476276	7.72213028	-14.777887	-30.283878	8.17120238	-4.697598
-0.8840497	2.2657824	-58.901654	59.3720513	-13.187666	-8.9934794	-9.2755512	-6.9207945	12.7119322	-3.5092195	-25.927019	12.366338	4.73030873
-0.376138	0.09164975	-61.287185	52.3966678	-11.607567	-8.9537096	-4.7831839	-11.185395	20.8588573	-4.5942997	-18.795524	12.0643179	0.85215037
-1.003366	6.66332818	-64.450984	39.3947018	-18.25556	-10.421256	-6.4950611	-13.579152	7.89613779	-10.044021	-3.0053508	1.09877909	1.7964002
-2.1867531	20.1856829	-69.502565	41.7980512	-8.5635838	13.8531815	-0.1115586	-9.2291509	3.89907192	-21.119416	-11.747332	11.4528895	1.29385366
-3.8990848	31.3965877	-49.196907	32.1970114	-6.739508	9.10072129	4.07650703	-16.299027	8.60910045	-17.780999	-7.577619	15.3483265	3.6121516
-2.7662778	48.1095952	-42.209995	28.6865311	-22.117479	22.728649	-13.57164	-10.549751	10.3982159	-9.7482785	-17.69128	7.71876705	1.56141767

Anexo 6. Coeficientes cepstrales palabra identificar.

```
from tkinter import *
from tkinter import messagebox
import pyaudio
import wave
import numpy as np
import scipy.io.wavfile as wavfile
import scipy.signal.windows as windows
import matplotlib.pylab as plt
from scipy import signal
from scipy.fftpack import fft
from python_speech_features import mfcc
import csv
import mysql.connector as mysql
from mysql.connector import Error
import math
import time

n = 0
##u = 0
##----------PyAudio----Settings-----------------------------##
form_1 = pyaudio.paInt16 # 16-bit resolution
chans = 1 # 1 channel
samp_rate = 44100 # 44.1kHz sampling rate
chunk = 4096 # 2^12 samples for buffer 4096
record_secs = 3 # seconds to record
dev_index = 2 # device index found by p.get_device_info_by_index(ii)

try:
    connection = mysql.connect(host = 'localhost',
                               database = 'usuarios',
                               user = 'root')

    sql_select_query = 'select id from users'

    cursor = connection.cursor(prepared=True)
    res = cursor.execute(sql_select_query)
    rec = cursor.fetchall()
    rec = np.asarray(rec)
    connection.commit()
    cursor.close()
```

Anexo 7. Código en Python parte 1.

```
except Error as e:
    connection.rollback()
    print("Error connecting: ",e)
finally:
    if connection.is_connected():
        connection.close()

nid = len(rec)
print(nid)

##-----Registro de usuario y almacenamiento en base de datos-----##
def addUser():
    global nid
    nid = nid + 1
    dataU = [[nid,userName.get()]]
    dataU = np.asarray(dataU)
    dataU = dataU.tolist()
    try:
        connection = mysql.connect(host = 'localhost',
                                   database = 'usuarios',
                                   user = 'root')

        sql_insert_query = 'INSERT INTO users(id,name) VALUES (%s,%s)'

        cursor = connection.cursor(prepared=True)
        res = cursor.executemany(sql_insert_query,dataU)
        connection.commit()
        cursor.close()

    except Error as e:
        connection.rollback()
        print("Error connecting: ",e)
    finally:
        if connection.is_connected():
            connection.close()
```

Anexo 8. Código en Python parte 2.

```
##----------Grabacion de usuario-----------------------####
    for u in range(5):
        audio = pyaudio.PyAudio() # create pyaudio instantiation
        wav_output_filename = (userName.get()+str(u)+'.wav')
        stream = audio.open(format = form_1,rate = samp_rate,channels = chans,\
                            input_device_index = dev_index,input = True,\
                            frames_per_buffer=chunk)

        lb3 = Label(project,text="Listening",fg="blue",font=("Agency FB",24)).place(x=180,y=70)
        project.update()
        frames = []

        # loop through stream and append audio chunks to frame array
        for ii in range(0,int((samp_rate/chunk)*record_secs)):
            data = stream.read(chunk)
            frames.append(data)


        lb3 = Label(project,text="Stand By",font=("Agency FB",24)).place(x=180,y=70)
        project.update()

        # stop the stream, close it, and terminate the pyaudio instantiation
        stream.stop_stream()
        stream.close()
        audio.terminate()

        # save the audio frames as .wav file
        wavefile = wave.open(wav_output_filename,'wb')
        wavefile.setnchannels(chans)
        wavefile.setsampwidth(audio.get_sample_size(form_1))
        wavefile.setframerate(samp_rate)
        wavefile.writeframes(b''.join(frames))
        wavefile.close()
```

Anexo 9. Código en Python parte 3.

```
##----------------------Almacenamiento de los archivos de voz-----------###
    rate,data = wavfile.read(wav_output_filename)
    data = data/abs(data.max())

    b,a = signal.butter(3,[50,65],'stop',False,'ba',rate)
    sf = signal.lfilter(b,a,data)
    b,a = signal.butter(3,[20,4000],'pass',False,'ba',rate)
    Data = signal.lfilter(b,a,sf)

    data1=data**2
    em=data1[0:len(data)].sum()/len(data)

    th=em*1.5       ##1.5-0.5
    step=250        ##250-100

##-----------------inicio de palabra---------------##
    for ji in range(0,len(data),step):
        t=(data1[ji:ji+step].sum())/step
        if t>th:
            x=ji-step
            break
##----------------final de palabra----------------##
    x0=x+15000
    for ji in range(x0,len(data),step):
        t=(data1[ji:ji+step].sum())/step
        if t<th*0.2:
            x1=ji-step
            break
##---------palabra---------------------------
    data = data[x:x1]
```

Anexo 10. Código en Python parte 4.

```
        coefs = mfcc(signal=data,samplerate=rate,nfft=1103,winfunc=windows.hamming)
        coefs1 = coefs.tolist()

        nameT = (userName.get()+str(u))
        try:
            connection = mysql.connect(host = 'localhost',
                                       database = 'usuarios',
                                       user = 'root')

            sql_select_query = 'create table '+nameT+'(F1 double,F2 double,F3 double,F4 double,F5
            cursor = connection.cursor()
            cursor.execute(sql_select_query)

            sql_insert_query = 'INSERT INTO '+nameT+'(F1,F2,F3,F4,F5,F6,F7,F8,F9,F10,F11,F12,F13)

            cursor = connection.cursor(prepared=True)
            res = cursor.executemany(sql_insert_query,coefs1)
            connection.commit()
        except Error as e:
            connection.rollback()
        finally:
            if connection.is_connected():
                cursor.close()
                connection.close()

        messagebox.showinfo("ACCESS","REMAINING RECORDINGS"+"\n"+str(5-(u+1)))

    return()
```

Anexo 11. Código en Python parte 5.

```
##-----Comparacion de grabacion con la base de datos------##
def access():
    global n
##----------Grabacion de usuario-----------------------####
    audio = pyaudio.PyAudio() # create pyaudio instantiation
    wav_output_filename = ('test.wav')
    stream = audio.open(format = form_1,rate = samp_rate,channels = chans,\
                        input_device_index = dev_index,input = True,\
                        frames_per_buffer=chunk)

    lb3 = Label(project,text="Listening",fg="blue",font=("Agency FB",24)).place(x=180,y=70)
    project.update()
    frames = []

    # loop through stream and append audio chunks to frame array
    for ii in range(0,int((samp_rate/chunk)*record_secs)):
        data = stream.read(chunk)
        frames.append(data)

    lb3 = Label(project,text="Stand By",font=("Agency FB",24)).place(x=180,y=70)
    project.update()

    # stop the stream, close it, and terminate the pyaudio instantiation
    stream.stop_stream()
    stream.close()
    audio.terminate()

    # save the audio frames as .wav file
    wavefile = wave.open(wav_output_filename,'wb')
    wavefile.setnchannels(chans)
    wavefile.setsampwidth(audio.get_sample_size(form_1))
    wavefile.setframerate(samp_rate)
    wavefile.writeframes(b''.join(frames))
    wavefile.close()
```

Anexo 12. Código en Python parte 6.

```
##----------------------Almacenamiento de los archivos de voz-----------###
    rate,data = wavfile.read('test.wav')
    data = data/abs(data.max())

    b,a = signal.butter(3,[50,65],'stop',False,'ba',rate)
    sf = signal.lfilter(b,a,data)
    b,a = signal.butter(3,[20,4000],'pass',False,'ba',rate)
    data = signal.lfilter(b,a,sf)

    data1=data**2
    em=data1[0:len(data)].sum()/len(data)

    th=em*1.5       ##1.5-0.5
    step=250        ##250-100

##-----------------inicio de palabra--------------##
    for ji in range(0,len(data),step):
        t=(data1[ji:ji+step].sum())/step
        if t>th:
            x=ji-step
            break
##----------------final de palabra---------------##
    x0=x+15000
    for ji in range(x0,len(data),step):
        t=(data1[ji:ji+step].sum())/step
        if t<th*0.2:
            x1=ji-step
            break
##---------palabra---------------------------
    data = data[x:x1]
```

Anexo 13. Código en Python parte 7.

```
    coefs1 = mfcc(signal=data,samplerate=rate,nfft=1103,winfunc=windows.hamming)

    try:
        connection = mysql.connect(host = 'localhost',
                                   database = 'usuarios',
                                   user = 'root')

        sql_select_query = "SELECT name FROM users"
        cursor = connection.cursor()
        cursor.execute(sql_select_query)
        rec = cursor.fetchall()
        users = np.asarray(rec)

        for k in range(len(users)):
            users[k,0] = str(users[k,0])

        for m in range(len(users)):
            for u in range(5):
                sql_select_query = 'SELECT * FROM '+users[m,0]+str(u)
                cursor = connection.cursor()
                cursor.execute(sql_select_query)
                rec = cursor.fetchall()
                coefs2 = np.asarray(rec)

                l1=len(coefs1)
                l2=len(coefs2)
```

Anexo 14. Código en Python parte 8.

```
if l1>l2:
    l = l1
    results1=np.zeros([l1,13],np.float)
    results2=np.zeros([l1,13],np.float)
    results1=coefs1
    results2[0:l2]=coefs2

if l2>l1:
    l = l2
    results1=np.zeros([l2,13],np.float)
    results2=np.zeros([l2,13],np.float)
    results2=coefs2
    results1[0:l1]=coefs1
if l1==l2:
    l=l1
    results1=coefs1
    results2=coefs2

dist=np.zeros([l],np.float)

for f in range(l):
    p = 0
    r = results1[f]-results2[f]
    for q in range(13):
        p = p + (r[q]*r[q])

    dist[f] = math.sqrt(p)
suma = 0
for n in range(l):
    suma = suma + dist[n]

prom = suma/l
aa=0
)
```

Anexo 15. Código en Python parte 9.

```
                if prom<40:
                    messagebox.showinfo("ACCESS",users[m,0]+"\nGRANTED")
                    fecha = time.strftime("%d/%m/%y")
                    hora = time.strftime("%H:%M:%S")
                    access = [[users[m,0],hora,fecha]]
                    access = np.asarray(access)
                    access = access.tolist()

                    sql_insert_query = 'INSERT INTO access(name,hour,date) VALUES (%s,%s,%s)'

                    cursor = connection.cursor(prepared=True)
                    res = cursor.executemany(sql_insert_query,access)
                    connection.commit()
                    aa = 1
                    print(users[m,0]+str(u))

                    break
            if aa==1:
                break
        if aa==0:
            messagebox.showerror("ACCESS","\nDENIED")
        print(prom,aa)
        cursor.close()
    except Error as e:
        connection.rollback()
        print(e)
    finally:
        if connection.is_connected():
            connection.close()

    return()

def out():
    exit()
```

Anexo 16. Código en Python parte 10.

```
def regedit():
    try:
        connection = mysql.connect(host = 'localhost',
                                   database = 'usuarios',
                                   user = 'root')

        sql_select_query = 'select * from access'

        cursor = connection.cursor(prepared=True)
        res = cursor.execute(sql_select_query)
        rec = cursor.fetchall()
        rec = np.asarray(rec)
        connection.commit()
        cursor.close()

    except Error as e:
        connection.rollback()
        print("Error connecting: ",e)
    finally:
        if connection.is_connected():
            connection.close()

    myFile = open('registro.csv', 'w')
    with myFile:
        writer = csv.writer(myFile)
        writer.writerows(rec)

    messagebox.showinfo(' ','Regedit imported')
```

Anexo 17. Código en Python parte 11.

```
project = Tk()
project.geometry("350x150")
project.title("CONTROL ACCESS")

userName = StringVar()
boxName = Entry(project,textvariable=userName,width=15).place(x=25,y=40)

sw1 = Button(project,text="Add User",font=("Agency FB",12),\
             command=addUser).place(x=40,y=70)
sw2 = Button(project,text="Request Access",font=("Agency FB",12),\
             command=access).place(x=180,y=25)
sw3 = Button(project,text="EXIT",fg="red",font=("Agency FB",12),\
             command=out).place(x=290,y=120)
sw4 = Button(project,text="Access Record",fg="blue",font=("Agency FB",12),\
             command=regedit).place(x=25,y=110)

lb1 = Label(project,text="New User",font=("Agency FB",12)).place(x=50,y=10)
lb3 = Label(project,text="Stand By",font=("Agency FB",24)).place(x=180,y=70)

project.mainloop()
```

Anexo 18. Código en Python parte 12.