ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

IMPLEMENTACIÓN DE UN MÓDULO DE RECONOCIMIENTO DE VOZ PARA NIÑOS MEDIANTE EL PROCESAMIENTO DE SEÑALES APLICADO EN UN CASO PRÁCTICO

Trabajo de titulación presentado para optar al grado académico de**:**

INGENIERA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

AUTORAS: KATTY BEATRIZ APOLO DIAZ
NELLY JACQUELINE COBA CASTILLO

TUTOR: ING. OSWALDO MARTÍNEZ, MSC.

Riobamba – Ecuador
2017

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMATICA Y ELECTRÓNICA

ESCUELA DE INGENIERIA EN ELECTRONICA TELECOMUNICACIONES Y REDES

El Tribunal del Proyecto de Titulación certifica que: El trabajo de investigación: IMPLEMENTACIÓN DE UN MÓDULO DE RECONOCIMIENTO DE VOZ PARA NIÑOS MEDIANTE EL PROCESAMIENTO DE SEÑALES APLICADO EN UN CASO PRÁCTICO, de responsabilidad de las señoritas Katty Beatriz Apolo Díaz y Nelly Jacqueline Coba Castillo, ha sido minuciosamente revisado por los Miembros del Tribunal del Proyecto de Titulación, quedando autorizada su presentación.

ING. WASHINGTON LUNA
DECANO DE LA FACULTA DE INFORMÁTICA Y ELECTRÓNICA _____ _____

ING. FLANKLIN MORENO
DIRECTOR DE LA ESCUELA DE INGENIERÍA ELECTRÓNICA, TELECOMUNICACIONES Y REDES _____ _____

ING. OSWALDO MARTÍNEZ
DIRECTOR DEL TRABAJO DE TITULACIÓN _____ _____

ING. EDWIN ALTAMIRANO
MIEMBRO DEL TRIBUNAL _____ _____

Nosotras, Katty Beatriz Apolo Díaz y Nelly Jacqueline Coba Castillo somos las responsables de las ideas, doctrinas y resultados expuestos en esta tesis y el patrimonio intelectual del Proyecto de Titulación pertenece a la Escuela Superior Politécnica de Chimborazo.

_____

Katty Beatriz Apolo Díaz

_____

Nelly Jacqueline Coba Castillo

DEDICATORIA

A Dios, por haberme dado la fortaleza para seguir adelante cuando me sentí derrotada y por haber puesto en mi vida a mis dos grandes ángeles, mi madre Beatriz y mi hijo Derek quienes con su ejemplo de lucha constante son mi principal fuente de inspiración para alcanzar las metas que me propongo.

A mi padre quien a base de esfuerzos logró que este sueño se hiciera realidad.

A mi esposo quien siempre ha estado presente con consejos y palabras de aliento en los momentos difíciles y con quien hemos logrado vencer grandes adversidades que nos han fortalecido como personas.

A mi familia, amigos, docentes y a todas las personas que creyeron en mí y me brindaron su apoyo incondicional.

Katty

Este trabajo de titulación se lo dedico a Dios quien me guio en este arduo camino dándome la fuerza necesaria para superar los obstáculos y dificultades que se me presentaron.

A mi madre Sandra Castillo que ha sido un ejemplo de lucha en todo momento quien con su amor, paciencia y cariño incondicional me ha dado su apoyo para seguir adelante en mis estudios.

A mi hermano Luis Coba, mis tíos Flavio Echeverría, Susana Castillo y mi primo Javier Echeverría por sus consejos, enseñándome que con amor, dedicación, humildad y perseverancia se puede alcanzar las metas que me proponga.

A mis profesores y amigos que estuvieron a lo largo de esta etapa brindándome sus experiencias y conocimientos.

Nelly

AGRADECIMIENTO

A Dios por no permitirme jamás desmayar y por haber puesto pruebas en mi camino que me ayudaron a crecer como persona, a mi padre Segundo y a mi madre Beatriz, a mis hermanas Karen, Mery y Alexandra y a mis sobrinos por todo el cariño y apoyo brindado.

A mi familia política, por brindarme su afecto y acogerme en su hogar, en especial a mis suegros por su la confianza y apoyo brindado, a mi esposo quien con su amor, paciencia y dedicación me ayudo a enfrentar los momentos difíciles.

A mi amigo y compañero Luis por su paciencia y por estar conmigo en los buenos y malos momentos a lo largo de este camino.

De manera especial, a nuestro tutor el Ing. Oswaldo Martínez, al Ing. David Moreno y al Ing. Edwin Altamirano por su tiempo, confianza y apoyo incondicional para la realización de este trabajo.

Katty

Agradezco primero a Dios por permitirme estar con vida y disfrutar de este momento, a mi madre Sandra Castillo, mi hermano Luis, mis tíos Flavio, Susana, primos Javier y Edison quienes depositaron toda su confianza en mí en esta etapa tan importante de mi vida.

A mi amigo Christian por su amistad, paciencia, por todos los logros y cosas buenas que compartimos durante estos años de formación académica. Asi también a todas las personas que me apoyaron y me brindaron su amistad en todo momento.

Un sincero agradecimiento al Ing. Oswaldo Martínez, Ing. David Moreno y al Ing. Edwin Altamirano por sus enseñanzas impartidas, su colaboración y apoyo incondicional para este trabajo de titulación.

Nelly

TABLA DE CONTENIDOS

Páginas

INDICE DE TABLAS ..... **x**
INDICE DE FIGURAS ..... **xi**
RESUMEN ..... **xv**
SUMMARY ..... **xvi**
INTRODUCCIÓN **1**
ANTECEDENTES **1**
CAPÍTULO I
1. **MARCO TEÓRICO** **5**
1.1 Fundamentos de la Voz **5**
1.1.1 Características de la voz ***5***
1.1.2 Análisis del habla ***5***
1.1.2.1 Articulación *6*
1.1.2.2 Percepción Auditiva *7*
1.1.2.3 Señal acústica *7*
1.1.3 Modelo general de producción de voz ***8***
1.1.4 Clasificación de los sonidos ***9***
1.2 Procesamiento Digital de Señales (PDS) **10**
1.2.1 Software utilizados para el PDS ***10***
1.2.1.1 MATLAB *10*
1.2.1.2 LabVIEW *11*
1.2.1.3 SciPy *11*
1.2.2 Procesamiento digital de voz ***12***
1.2.2.1 Reconocimiento de voz *12*
1.2.2.2 Proceso del reconocimiento de voz *13*
1.3 Métodos y Etapas para el Reconocimiento de Voz **14**
1.3.1 Representación de la señal de voz en el dominio del tiempo y la frecuencia ***14***
1.3.1.1 La transformada de Fourier *15*
1.3.1.2 La transformada de Z *15*
1.3.2 Adquisición de la voz ***17***
1.3.2.1 Características esenciales de un micrófono *17*

1.3.2.2 Clasificación de los micrófonos ... *18*

1.3.3 Digitalización de la señal de voz ... ***20***

1.3.3.1 Muestreo ... *20*

1.3.3.2 Cuantificación ... *22*

1.3.3.3 Codificación ... *24*

1.3.4 Pre-procesamiento ... ***24***

1.3.4.1 Segmentación y enventanado de la señal ... *24*

1.3.4.2 Segmentado de tramas ... *26*

1.3.5 Filtrado de preénfasis ... ***28***

1.3.6 Reconocimiento: Extracción de características ... ***29***

1.3.6.1 LPC (Codificación Predictiva Lineal) ... *29*

1.3.6.1.1 Envolvente LPC ... *29*

1.3.6.1.2 Predicción de Error ... *32*

1.3.6.1.3 Número de Coeficientes LPC ... *36*

1.3.7 Reconocimiento: Comparación y decisión ... ***36***

1.4 Tarjetas de Adquisición de Datos ... **37**

1.4.1 Placa Arduino ... ***37***

1.4.2 Tarjeta Galileo ... ***38***

1.4.3 Tarjeta Raspberry Pi ... ***39***

1.5 Módulos de Comunicación Inalámbrica ... **40**

1.5.1 Modulo Bluetooth ... ***40***

1.5.2 Módulos RF 433Mhz ... ***40***

1.5.3 Módulo NRF24L01 ... ***41***

1.6 Elementos Electrónicos ... **41**

1.6.1 Batería Lipo 7.4v ... ***41***

1.6.2 Modulo Regulador de voltaje ... ***42***

1.6.3 Módulo L298N ... ***42***

CAPÍTULO II

2 MARCO METODOLÓGICO ... **43**

2.1 Algoritmo para el Procesamiento de voz (Reconocimiento de comandos de voz) **44**

2.2 Adquisición de la voz ... **47**

2.2.1 Base de Datos ... ***47***

2.3 Pre procesamiento ... **48**

2.3.1 Normalización y determinación del umbral ... ***48***

2.3.2 Segmentación *49*
2.3.3 Calculo de energía *49*
2.3.4 Eliminación de silencios *49*
2.4 Filtrado preénfasis **50**
2.5 Extracción de características **50**
2.5.1 Análisis LPC **53**
2.5.1.1 Error de Predicción: *53*
2.5.1.2 Modelo AR o de todos polos *54*
2.5.1.3 Formantes de la Señal *54*
2.5.1.3.1 Método Peak-Picking *54*
2.5.1.3.2 Método de las Raíces *55*
2.6 Cálculo de Distancias **56**
2.7 Comparación de Distancias **56**
2.8 Caso Práctico **57**
2.8.1 Selección del Módulos de Comunicación Inalámbrica *57*
2.8.2 Selección de la Tarjeta de Adquisición *58*
2.8.3 Enlace Matlab Arduino *58*
2.9 Diseño e Implementación del Módulo de Reconocimiento de Voz **59**
2.9.1 Módulo Transmisor *59*
2.9.2 Módulo Receptor *63*
CAPÍTULO III
3 MARCO DE RESULTADOS Y DISCUSIÓN **65**
3.1 Programa de Reconocimiento de voz para Niños **65**
3.1.1 Interfaz Base de Datos *65*
3.1.2 Interfaz Reconocimiento de voz para Niños *66*
3.1.2.1 Procesamiento de Señal *66*
3.1.2.2 Reconocimiento *67*
3.2 Resultado de la Implementación del Módulo de Reconocimiento de Voz **68**
3.3 Pruebas y Resultados **69**
3.3.1 Prueba-1 ..... *69*
3.3.2 Resultado Prueba-1 ..... *71*
3.3.2.1 Coeficientes LPC *72*
3.3.2.2 Coeficientes LPC Señales Patrón *75*
3.3.2.3 Distancias Totales *78*
3.3.3 Prueba-2 ..... *79*

3.3.4 Resultado Prueba-2 ..... *81*
3.3.5 Prueba 3 (Caso Práctico) *81*
3.3.6 Resultado Final *81*
3.4 Análisis de Resultados **83**
3.4.1 Derecha *83*
3.4.2 Izquierda *88*
3.4.3 Adelante *90*
3.4.4 Atrás *93*
3.4.5 Para *95*
3.5 Sistematización **99**
CONCLUSIONES **101**
RECOMENDACIONES **103**
BIBLIOGRAFÍA
ANEXOS

INDICE DE TABLAS

Páginas

Tabla 1-1 Tipos de ventanas 26
Tabla 1-2 Cálculo del umbral..... 48
Tabla 2-2 Descripción de Entradas y Salida. 58
Tabla 3-2 Salida del Primer Arduino. 62
Tabla 4-2 E/S del segundo Arduino. 62
Tabla 5-2 E/S Arduino Rx. 63
Tabla 6-2 Entradas Driver de motores DC. 64
Tabla 1-3 Prueba-1 Usuario-1 69
Tabla 2-3 Prueba-1 Usuario-2 70
Tabla 3-3 Prueba1 Usuario-3 71
Tabla 4-3 Coeficientes LPC Usuario-1 72
Tabla 5-3 Coeficientes LPC Usuario-2 73
Tabla 6-3 Coeficientes LPC Usuario-3 74
Tabla 7-3 Coeficientes LPC 75
Tabla 8-3 Coeficientes LPC Señales Patrón 76
Tabla 9-3 Coeficientes LPC Señales Patrón 77
Tabla 10-3 Distancias Totales Usuario-1 78
Tabla 11-3 Distancias Totales Usuario-2 78
Tabla 12-3 Distancias Totales Usuario-3 79
Tabla 13-3 Prueba2 Usuario-1 79
Tabla 14-3 Prueba2 Usuario-2 80
Tabla 15-3 Prueba2 Usuario-3 80
Tabla 16-3 Prueba-3..... 81
Tabla 17-3 Pruebas Totales..... 81
Tabla 18-3 Comparación en función al número de muestras 85
Tabla 19-3 Comparación entre la Señal Original y Silencios 88
Tabla 20-3 Comparación en función al número de muestras 88
Tabla 21-3 Comparación entre la Señal original y la Señal Silencios 90
Tabla 22-3 Comparación entre la Señal Original y Silencios 91
Tabla 23-3 Comparación entre la Señal original y Señal Silencios 93
Tabla 24-3 Comparación entre la Señal Original y Señal Silencios 93
Tabla 25-3 Comparación entre la Señal original y Señal Silencios 95
Tabla 26-3 Comparación entre la Señal Original y Silencios 96

INDICE DE FIGURAS

Páginas

Figura 1-1 Aparato Fonador. 6
Figura 2-1 Modelo acústico del tracto vocal. 9
Figura 3-1 Proceso del reconocimiento de voz. 13
Figura 4-1 Gráfica de una señal de voz en el dominio del tiempo. 14
Figura 5-1 Gráfica de una señal de voz en el dominio del tiempo. 14
Figura 6-1 Clasificación de los micrófonos según la Directividad. 18
Figura 7-1 Clasificación de los micrófonos según el Transductor. 19
Figura 8-1 Clasificación de los micrófonos según la utilidad. 19
Figura 9-1 Proceso de conversión A/D. 20
Figura 10-1 Cuantificación. 22
Figura 11-1 Cuantificación Uniforme. 23
Figura 12-1 Cuantificación No-Uniforme. 23
Figura 13-1 Enventanado. 25
Figura 14-1 Solapamiento de una señal. 25
Figura 15-1 Filtro Preénfasis 29
Figura 16-1 Modelo AR 30
Figura 17-1 Método de Raíces y Método Peak-Picking 32
Figura 18-1 Modelo AR y Predicción de Error 32
Figura 19-1 Predicción de Error 33
Figura 20-1 Partes de la placa Arduino Uno. 38
Figura 21-1 Tarjeta Galileo 39
Figura 22-1 Tarjeta Raspberry Pi. 39
Figura 23-1 Modulo Bluetooth. 40
Figura 24-1 Módulos RF 433Mhz. 40
Figura 25-1 Módulos NRF24L01. 41
Figura 26-1 Batería Lipo 7.4v 2000mah. 42
Figura 27-1 Lm2596. 42
Figura 1-2 Algoritmo procesamiento de voz (proceso de reconocimiento) 45
Figura 2-2 Diagrama de bloques de proceso de reconocimiento. 46
Figura 3-2 Stereo Headset 662862 47
Figura 4-2 Código para adquirir la voz. 47
Figura 5-2 Código para realizar la normalización y cálculo energía. 49
Figura 6-2 Código para segmentar la voz. 49
Figura 7-2 Código para cálculo de energía de cada segmento. 49
Figura 8-2 Código para eliminar silencios. 50
Figura 9-2 Código para filtrado preénfasis 50

Figura 10-2 Código para calcular coeficientes lpc de la base de datos. 51
Figura 11-2 Código para calcular lpc de la señal ingresada por un usuario. 52
Figura 12-2 Código para el Error de Predicción 53
Figura 13-2 Error de Predicción 53
Figura 14-2 Código para Modelo AR 54
Figura 15-2 Código Peak Picking 55
Figura 16-2 Código Método de las Raíces 55
Figura 17-2 Interfaz Gráfica Formantes 55
Figura 18-2 Código para calcular distancias. 56
Figura 19-2 Código para elegir la menor distancia total. 57
Figura 20-2 Enlace Matlab Arduino. 58
Figura 21-2 Módulo de Reconocimiento de Voz Tx. 59
Figura 22-2 Comandos en Matlab enviar datos a Arduino 60
Figura 23-2 Datos enviados desde Matlab a Arduino 61
Figura 24-2 Conexión LCD y bus I2C 62
Figura 25-2 Módulo de Reconocimiento de Voz Rx. 63
Figura 1-3 Interfaz Gráfica Base de Datos. 66
Figura 2-3 Interfaz Gráfica Reconocimiento 68
Figura 3-3 Módulo de voz y caso práctico 68
Figura 4-3 Foto de Usuario-1 69
Figura 5-3 Foto de Usuario-2 70
Figura 6-3 Foto de Usuario-3 71
Figura 7-3 Interfaz Gráfica Adquisición 83
Figura 8-3 Señal original en función del tiempo 83
Figura 9-3 Señal de voz eliminada silencios del comando derecha 84
Figura 10-3 Señal eliminada los Silencios en función del tiempo 84
Figura 11-3 Señal original y Señal eliminada los Silencios en función del número de muestras 85
Figura 12-3 Señales de voz con Filtro Preénfasis 85
Figura 13-3 Señal Silencios sin filtro y Señal Silencios con filtro 86
Figura 14-3 Análisis de LPC comando Derecha 86
Figura 15-3 Señal Limpia_Usuario enventanada y Error de Predicción 87
Figura 16-3 Distancias Totales 87
Figura 17-3 Señal original (izquierda) y señal Silencios en función del tiempo 88
Figura 18-3 Señal original (izquierda) y señal Silencios en función de número de muestras 88
Figura 19-3 Señal de voz filtrada 89
Figura 20-3 Análisis LPC Izquierda 89
Figura 21-3 Distancias Totales 90
Figura 22-3 Señal original (adelante) y señal Silencios en función del tiempo 90
Figura 23-3 Señal original (adelante) y señal Silencios en función de número de muestras 91
Figura 24-3 Interfaz Gráfica Preénfasis 91

Figura 25-3 Interfaz Gráfica LPC (adelante) 92
Figura 26-3 Distancias Totales 92
Figura 27-3 Señal original (atrás) y señal Silencios en función del tiempo 93
Figura 28-3 Señal original (atrás) y señal Silencios en función de número de muestras 93
Figura 29-3 Interfaz Gráfica Preénfasis 94
Figura 30-3 Interfaz Gráfica LPC (atrás) 94
Figura 31-3 Distancias Totales 95
Figura 32-3 Señal original (para) y señal Silencios en función del tiempo 95
Figura 33-3 Señal original (para) y señal Silencios en función de numero de muestras 96
Figura 34-3 Interfaz Gráfica Preénfasis 96
Figura 35-3 Interfaz Gráfica LPC (para) 97
Figura 36-3 Distancias Totales 97
Figura 37-3 Envolvente de las señales (Formantes) 98

ÍNDICE DE ABREVIATURAS

A/D	Analógico/Digital
AR	Autorregresivo
DPC	Detección de Palabras Clave
GND	Ground
DC	Corriente Directa
FPGA	Field Programmable Gate Array
IDE	Integrated Development Environment (Entorno de desarrollo integrado)
LiPo	Litio y Polímero
LPC	Codificación Predictiva Lineal
LTI	Linear Time-Invariant (Sistema lineal e invariante en el tiempo)
PAP	Motores Paso a Paso
PBC	Printed Circuit Board (Plaqueta de circuito impreso)
PDS	Procesamiento de Señal Digital (Digital Signal Processing)
PWM	Pulse Width Modulation **(**Modulación por ancho de pulsos)
RAHC	Reconocimiento Automático del Habla Continua
RPA	Reconocimiento de Palabras Aisladas
RPC	Reconocimiento de Palabras Conectadas
ZCR	Zero-Crossing Rate (Tasa de cruces por Cero)
SBC	Sistema de Bajo Costo
SCL	System Clock
SD	Secure Digital **(**Tarjetas digitales seguras)
SDA	System Data
SDHC	Secure Digital High Capacity **(**Tarjetas digitales seguras de alta capacidad)
Vcc	Voltaje corriente continúa

RESUMEN

Se realizó un Módulo de reconocimiento de voz para niños mediante el procesamiento de señales aplicado en el control de un carro robot 4wd a través de comandos de voz: derecha, izquierda, adelante, atrás y para. El programa consta de dos interfaces gráficas desarrolladas en Matlab denominadas base de datos y reconocimiento de voz para niños. En la interfaz base de datos se realizaron los procesos correspondientes a la adquisición de la voz utilizando un micrófono Stereo Headset, una vez adquirida la señal de voz se realizó el preprocesamiento cuyo objetivo fue eliminar silencios indeseados de la señal para lo que se empleó el método de cálculo de energía el que engloba procesos de normalización, determinación del umbral y segmentación. A esta nueva señal se empleó un filtro preénfasis para la acentuación de las frecuencias altas y suavizar el espectro, de esta señal filtrada se extrajo las características de voz a través del Cálculo de Coeficientes de Predicción Lineal (LPC), para obtener la señal patrón se realizó un promedio entre los coeficientes de cada grabación. La segunda interfaz gráfica realiza los mismos procesos descritos anteriormente para la señal de voz a reconocer, luego se realiza una comparación de distancias entre la señal de voz a reconocer y cada una de las señales patrón siendo reconocida el comando de voz cuya distancia sea mínima. De acuerdo a las pruebas realizadas con el Módulo de reconocimiento de voz para niños se determinó que tiene una efectividad total de 84,70% estos resultados fueron comparados con el Módulo existente en el mercado SR-07 utilizado en la tesis "Diseño e implementación del sistema de movimiento direccional de una silla de ruedas para ser controlada por reconocimiento de un patrón de voz mediante electrónica de potencia y motores DC como actuadores", por lo que concluye que se cumplió con cada uno de los objetivos propuestos en este trabajo de titulación debido a que el módulo implementado proporcionó mejores resultados, sin embargo se recomienda probar los métodos actuales utilizados en el reconocimiento de voz con la finalidad de optimizar aún más los resultados.

Palabras Claves: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <TECNOLOGÍA DE LAS COMUNICACIONES>, <PROCESAMIENTO DE SEÑALES >, <VOZ HUMANA>, <ENERGÍA>, <UMBRAL>, < CÁLCULO DE COEFICIENTES DE PREDICCIÓN LINEAL (LPC) >, <DISTANCIA EUCLIDIANA>.

SUMMARY

A Child voice recognition was developed by signal processing applied to the control of 4wd robot car througth voice commands: right, left, forward, backward and halt. The program consists of two graphical interfaces developed in Matlab called database and voice recognition for children. In the database interface processes were performed corresponding to the acquisition of the voice using a Stereo Headset microphone, once acquired the speech signal processing was performed whose objective was to eliminate unwanted silences of the signal for which the method was used of energy calculation which encompasses normalization, threshold determination and segmentation processes. To this new signal, we used a pre-emphasis filter to accentuate the high frequencies and smooth the spectrum, from this filtered signal we extracted the voice characteristics through the Calculation of Linear Prediction Coefficients (LPC), to obtain the signal was performed an average between the coefficients of each recording. The second graphical interface performs the same processes described above for the speech signal to be recognized, then a comparison of distances between the speech signal to be recognized and each of the pattern signals is performed, the speech command having a minimum distance being recognized. According to the tests carried out with the Voice Recognition Module for children, it was determined that it has a total effectiveness of 84.70%.These results were compared with the existing Module in the SR-07 market used in the thesis "Design and implementation of the System of directional movement of a wheelchair to be controlled by recognition of a voice pattern by means of power electronics and DC motors as actuators ", thus concluding that it fulfilled each of the objectives proposed in this titration work due to that the implemented module provided better results, however it is recommended to test the current methods used in voice recognition in order to further optimize the results.

Keywords: <TECHNOLOGY AND ENGINEERING SCIENCES>, <COMMUNICATIONS TECHNOLOGY>, <SIGNAL PROCESSING>, <HUMAN VOICE>, <ENERGY>, <THRESHOLD> <CALCULATION OF LINEAR PREDICTION COEFFICIENTS (LPC)>, <EUCLIDIAN DISTANCE>.

INTRODUCCIÓN

ANTECEDENTES

El procesamiento de la señal digital DSP se practicó durante varios años e incluso antes que apareciera el computador ,en el año de 1928 Harry Nyquist realizó uno de los avances de DSP con el artículo "Certain topics in Telegraph Transmission Theory" ,en el cual especificó el efecto producido en el espectro de frecuencias de una señal analógica al ser discreta en el tiempo y se planteó que para conservar la información original, la tasa de muestreo tendría que ser mayor que el doble de la componente de frecuencia que contenía la señal analógica . En 1949 se publicó el artículo "Communications in the Presence of Noise" realizado por Claude Shannon en el cual demostró que es posible reconstruir una señal analógica partiendo de sus muestras con la ayuda de filtros pasabajos ideales.

El procesamiento de la señal digital en las últimas décadas ha incrementado en un avance tecnológico, desarrollando microcontroladores cada vez más potentes que ofrecen diferentes aplicaciones como es el procesamiento de la señal de voz que permite interactuar los diferentes usuarios mediante el reconocimiento de la señal de voz. Hoy en día existen diferentes empresas fabricantes de DSP como Texas Instrumen, Analog Devices, Miicrochip, MathWorks que con la ayuda de hardware y software como es LabVIEW, FPGA, VHDL, Matlab se puede llegar a realizar el análisis y procesamiento de la señal.

Kurzwei indica que el reconocimiento de voz es el proceso de convertir, por medio de una computadora, una señal acústica a una secuencia de palabras representadas en texto, estas palabras pueden servir de entrada a otros dispositivos que los requieren para realizar alguna acción, como activar dispositivos , estas técnicas se han logrado incorporar en diferentes aplicaciones como es el caso del control automático de sillas de ruedas a través de comandos de voz que han ayudado a las personas que presentan dificultades físicas para poder movilizarse .

En la tesis con el tema: "Implementación de un sistema de control para el manejo automático de una silla de ruedas" realizada en la Escuela Superior Politécnica de Chimborazo por la Ing. Estefany Gabriela Cujano, se especifica que para el desarrollo del proyecto se utilizó el módulo de reconocimiento de voz VRBOT el mismo que presentó inconvenientes dado que se debía repetir varias veces el comando específico para que el módulo realice las diferentes acciones solicitadas.

La implementación de un módulo de reconocimiento de voz para niños mediante el procesamiento de señales aplicado en un caso práctico contribuirá en estudios futuros concernientes al procesamiento de la voz.

FORMULACIÓN DEL PROBLEMA

Debido al avance tecnológico se han desarrollarlo diversas herramientas para el procesamiento de la voz que han sido utilizadas para desarrollar un sin número de aplicaciones. Actualmente existen diferentes módulos de reconocimiento de voz los cuales presentan fallas que se relacionan con el ruido y la tonalidad de la persona.

En el mercado ecuatoriano se puede encontrar diferentes módulos de reconocimiento de voz, los cuales no son muy óptimos dado que se necesita repetir una instrucción varias veces para que pueda ser reconocida, es importante considerar que si el módulo es utilizado por un niño se requiere que el mismo aumente su tono de voz ya que el modulo ha sido diseñado para que trabaje en la banda de frecuencias de voz, es decir ha sido modelado para un rango de frecuencias abiertas.

SISTEMATIZACIÓN DEL PROBLEMA

¿Cuáles son las aplicaciones del Procesamiento de la Señal Digital?

¿Qué software se puede utilizar para el Procesamiento de la Señal de Voz?

¿Qué problemas presentan los módulos DSP existentes en el mercado?

¿Qué problemas implica procesar una señal y como se puede solucionar?

JUSTIFICACIÓN

JUSTIFICACIÓN TEÓRICA

En la actualidad existen módulos de reconocimiento de voz que trabajan en la banda de frecuencias de 20Hz a 20KHz razón por la cual presentan inconvenientes al momento de ser utilizados ya que trabajan en un rango de frecuencias abiertas.

Hay diferentes factores que provocan el funcionamiento erróneo de los módulos DSP de voz entre los cuales se puede considerar que tienen mayor importancia el ruido y la tonalidad de la voz, es por esta razón que se considera al procesamiento de las señales de voz como una alternativa eficaz para modelar las señales acorde a lo requerido y así poder tener señales nítidas que pueden ser usadas en múltiples aplicaciones.

JUSTIFICACIÓN APLICATIVA

El análisis de una señal de voz tiene sus inicios desde el momento en que las cuerdas bucales generan dicha señal que hasta ese instante es considerada como señal sonora. Las señales sonoras generadas por niños se caracterizan por estar en frecuencias que suelen oscilar entre 250 Hz a 300 Hz.

La señal de voz básicamente está constituida por ondas sonoras, por lo cual para la obtención de este tipo de señales se requerirá un micrófono, el cual se encargará de convertir la onda sonora en una señal eléctrica.

La señal analógica obtenida por medio del micrófono se deberá convertir en formato digital para poder así realizar su procesamiento con la ayuda del software. El procesamiento se lo realizará mediante los procesos de muestreo, cuantificación y codificación, los cuales agrupan además varios procesos que permitirán obtener la señal deseada.

Una vez que se ha realizado todo el modelamiento del procesamiento de la señal se incorpora dicho software en una tarjeta que permita ejecutarlo y así comprobar su funcionamiento.

OBJETIVOS

OBJETIVO GENERAL

- Implementar un módulo de reconocimiento de voz para niños mediante el procesamiento de señales aplicado en un caso práctico.

OBJETIVOS ESPECIFICOS

- Investigar el procesamiento de señales de voz en niños y adultos.
- Seleccionar el software que proporcione las herramientas necesarias para desarrollar el procesamiento de la Señal de Voz.
- Diseñar e implementar en el software seleccionado el algoritmo que permita realizar el procesamiento de la voz generada por niños.
- Implementar el módulo de reconocimiento de voz que pueda ser utilizado en un caso práctico.
- Comparar el módulo de reconocimiento de voz con un módulo existente el mercado.

CAPÍTULO I

1. MARCO TEÓRICO

1.1 Fundamentos de la Voz

La palabra voz proviene del latín vox, es utilizada para nombrar al sonido que se produce cuando las cuerdas vocales vibran, dicha vibración se da cuando el aire sale desde los pulmones para dirigirse a la laringe.

1.1.1 Características de la voz

Para describir las características de la voz se hace referencia a los siguientes elementos:

Intensidad: es la potencia con la que un sonido es emitido, normalmente se asocia al volumen de la voz, se mide en decibelios y por medio de esta se determina los umbrales auditivos.

Duración: es el tiempo que se tarda en producir un sonido, por medio de esta se determina la velocidad con la que se emiten las palabras.

Timbre: al ser considerado como el espectro específico de la voz permite diferenciar de aquellas voces que tengan similitud de intensidad y tono, por lo que es considerado la característica más importante al momento de distinguir voces.

Tono: es la frecuencia del sonido, es decir la cantidad de vibraciones que produce una onda acústica.

1.1.2 Análisis del habla

Para el análisis del habla se toma en cuenta diferentes puntos de vista de los cuales en cuanto al reconocimiento se refiere se consideran los siguientes enfoques debido a su eficiencia:

Articulación: Análisis de la producción de sonidos que generan el habla humana.

Percepción Auditiva: Análisis de la manera en la cual el hombre procesa el habla.

Señal Acústica: Análisis de las ondas sonoras que produce el ser humano.

1.1.2.1 Articulación

Para emitir la voz se necesita del trabajo en conjunto de varios órganos de nuestro cuerpo, los cuales poseen funciones independientes. Estos órganos forman parte del Aparato Fonador humado y se los ha dividido según sus funciones en tres mecanismos:

En la Figura 1-1 se puede observar de manera detallada los órganos que forman parte del Aparto Fonador Humano.

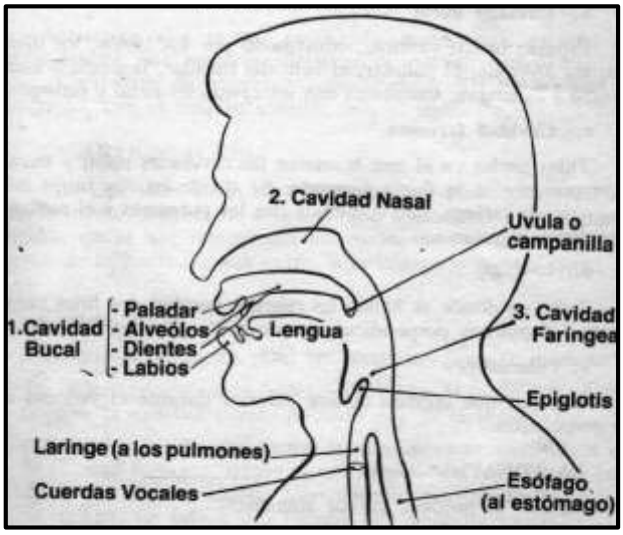


Figura 1-1 Aparato Fonador.
Fuente: A. Alonso, 2011, El Aparato Fonador.

Mecanismo Respiratorio: es donde se da origen a la voz debido a que es aquí donde se recoge el aire, el mismo que ingresa por la nariz para pasar a la faringe y laringe hasta llegar a la tráquea la cual está formada por los bronquios los cuales desembocan en los pulmones y se ramifican formando los bronquiolos los cuales terminan en unos pequeños sacos de aire donde se realiza el intercambio de gases con la sangre conocidos como los alveolos. A este mecanismo lo forman las fosas nasales, tráquea., receptáculo aéreo (pulmones. bronquios y caja torácica) y los músculos de la respiración (diafragma, músculos intercostales y abdominales) (Alonso, 2011, p.16).

Mecanismo Fonador: Este mecanismo formado por la laringe y las cuerdas vocales es el encargado de producir el sonido y determinar el tono. Cuando el aire que ha ingresado a los

pulmones es expulsado se dirige hacia las cuerdas vocales y las hace vibrar generando así ondas sonoras en la columna de aire que fluye por la faringe nariz y boca (Alonso, 2011, p.24).

Mecanismo Resonador: su función principal es convertir en habla comprensible los sonidos producidos por la vibración de las cuerdas vocales. Está formado por la faringe, la boca, las fosas nasales y los senos cráneo-faciales (Alonso, 2011, pp.27-28).

1.1.2.2 Percepción Auditiva

La percepción auditiva permite al ser humano interpretar los sonidos provenientes de su alrededor gracias al trabajo de los órganos auditivos. Para poder distinguir los diferentes tipos de habla humana se analiza el tono la intensidad y el timbre de la voz.

El margen de frecuencias del sonido es bastante inferior aproximadamente está entre los 20 Hz y 20 KHz, por lo cual todo dispositivo de audio debe ser diseñado de tal manera que responda al margen de frecuencias antes mencionado (Pohlmann, 2002, p.2).

Es importante considerar que el ser humano procesa las frecuencias por grupos mas no individualmente lo que permite que pueda distinguir ruidos presentes a su alrededor. La frecuencia fundamental del habla varia durante la comunicación, se puede decir que la frecuencia fundamental oscila los 125 Hz para la voz masculina, 250 Hz para la voz femenina y 350 Hz para la voz emitida por niños (Jackson, 1992, p.54).

1.1.2.3 Señal acústica

Un reconocedor de voz no se basa en el análisis del movimiento de la boca, se basa en el análisis de la señal de la voz (Villalobos et al., 2013, p.8). La ciencia que se encarga de este tipo de análisis es la acústica la cual define las características específicas de la señal del habla necesarias para realizar un óptimo reconocimiento de voz, de las cuales se puede destacar:

Coarticulación: Se le da este nombre al proceso de producir un sonido o fonema casi al mismo tiempo que el siguiente, normalmente se utiliza este término para distinguir entre las características de un sonido de habla es decir un fonema y su forma de pronunciación es decir el tono (Villalobos et al., 2013, p.10).

Resonancia: Este término se asocia con la vibración de un sonido y su efecto vibrante en otro objeto. La resonancia permite que cada persona tenga una voz diferente debido a que las frecuencias de habla son diferentes ya que cada individuo posea diferente fisiología.

Frecuencia y amplitud: La frecuencia está relacionada directamente con el tono, está determinada por el número de vibraciones de la onda de sonido en un determinado tiempo, dependiendo de esta se pueden producir tonos graves o agudos. La amplitud se relaciona con el volumen que tiene un sonido es decir con la intensidad.

Estructura Armónica y Ruido: Un sonido armónico está formado por la mezcla de varios sonidos superpuestos, en los cuales se debe cumplir que las frecuencias de dichos sonidos sean múltiplos de la frecuencia fundamental, la cual permite determinar la periodicidad y altura tonal del sonido resultante. Las vocales internamente están formadas de ondas cíclicas las cuales poseen patrones repetitivos y a cíclicas que no poseen patrones repetitivos a los cuales se les da el nombre de ruido, estas ondas a cíclicas forman parte de las consonantes, semivocales y fonemas en general (Villalobos et al., 2013, p.9).

Para poder visualizar de mejor manera los patrones armónicos y de ruido se utilizan espectrogramas los cuales hacen uso de la transformada de Fourier permitiendo de esta manera identificar palabras y fonemas.

1.1.3 Modelo general de producción de voz

Para modelar el tracto vocal se usa el modelo de tubos mostrado en la Figura 2-1, el cual representa al tracto vocal como una cavidad resonante, es decir un tubo acústico no uniforme sin perdidas en el cual un extremo representa la glotis y el otro extremo representa los labios. Este tipo de representación se usa debido a que las ondas acústicas se propagan a lo largo de un eje en una única dirección.

Cuando un sonido atraviesa esta cavidad resonante se podrá distinguir diversas frecuencias a las que se les conoce como formantes, normalmente el número de formantes es proporcional al número de resonadores que posea el tracto vocal aunque para diferenciar los distintos tipos de sonidos se consideran solo los tres primeros formantes (F1, F2, F3) que cubren un rango de frecuencias entre 100 y 3500 Hz. Esto debido a que las resonancias de alta frecuencia son

atenuadas por la característica frecuencial del tracto que tiende a actuar como un filtro pasa bajo con una caída de aproximadamente -12 dB por octava.

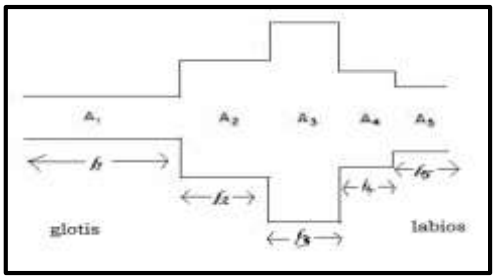


Figura 2-1 Modelo acústico del tracto vocal.

Fuente: G. Velásquez, 2008, Modelo acústico del tracto vocal.

1.1.4 Clasificación de los sonidos

De acuerdo a como se generan a los sonidos se los puede clasificar en (Duque y Morales, 2007, p.7):

Sonidos Sonoros: Las cuerdas vocales vibran debido al paso del aire por el tracto vocal sin interferencias por lo cual generan un tren de pulsos cuasi-periódicos, se caracterizan por tener niveles altos de energía generalmente se encuentran en el rango de frecuencias 300Hz a 4kHz.

Sonidos No-Sonoros: también conocidos como sordos o no tonales, se diferencian de los sonoros porque las cuerdas vocales no vibran debido a que existe dificultad del paso de aire por el tracto vocal, se caracterizan por tener uniformidad de componentes frecuenciales ,baja energía y amplitudes pequeñas. Matemáticamente este tipo de sonidos son modelados como ruido blanco.

Sonidos oclusivos: este tipo de sonido se genera debido al cierre momentáneo del tracto vocal con lo que se genera una excitación temporal, es decir se impide el paso de aire por un momento y luego se lo libera.

A la imagen mental o modelo que el ser humano tiene de un sonido se le denomina fonema, su clasificación de acuerdo al modo de articulación se muestra en el ANEXO A. Es importante considerar que los fonemas vocálicos tienen frecuencias bajas con respecto a las consonantes.

1.2 Procesamiento Digital de Señales (PDS)

El procesamiento digital de señales (Digital Signal Processing) es el conjunto de técnicas y herramientas para tratar señales en el dominio discreto o digital, para lo cual las manipula matemáticamente obteniendo así una señal mejorada la cual pude utilizarse en múltiples aplicaciones. Este procesamiento se lo realiza utilizando hardware digital el cual debe proveer el software correspondiente que brinde las herramientas necesarias para el correcto tratamiento de la señal.

Existen diferentes tipos de procesamiento digital los cuales se mencionan a continuación:

- Procesamiento digital de audio
- Procesamiento digital de datos
- Procesamiento digital de imágenes
- Procesamiento digital de vídeo
- Procesamiento digital de voz

1.2.1 Software utilizados para el PDS

Existen diversos software que brindan las herramientas necesarias para realizar el procesamiento digital de señales, de los cuales se puede destacar:

1.2.1.1 MATLAB

Es un software licenciado desarrollado por Mathworks que utiliza lenguaje de alto nivel propio (lenguaje m), el cual proporciona las herramientas necesarias para la creación de aplicaciones científicas y de ingeniería.

Características (MATHWORKS, 2016, https://www.mathworks.com/products/matlab/features.html):

- Se basa en el cálculo con matrices y vectores lo cual permite ahorrar tiempo de ejecución.
- Programación sencilla.
- Integra análisis numérico, cálculo matricial, proceso de señal y visualización gráfica (2D y 3D).

- Se usa en áreas de computación y cálculo numérico tradicional, prototipaje algorítmico, teoría de control automático, estadística, procesamiento digital de señales.
- Cuenta con las herramientas Simulink y GUIDE.
- Permite crear y manipular señales para su procesamiento.
- Compatible con hardware Arduino, Raspberry Pi, Altera, National Instrument, ARM, Keysight, ST Microelectronics, Xilinx, Android.

1.2.1.2 LabVIEW

Es un software licenciado desarrollado por National Instrument que utiliza lenguaje gráfico (lenguaje G), el cual facilita la creación y visualización de sistemas de ingeniería.

Características (NATIONAL INSTRUMENTS, 2016, http://www.ni.com/labview/esa/):

- LabVIEW es un entorno diseñado específicamente desarrollar sistemas de control y medidas.
- Lenguaje de programación gráfico, parecido a Simulink
- Permite crear aplicaciones y generar prototipos con tecnología FPGA.
- Permite visualizar y crear graficas con datos dinámicos
- Es compatible con MATLAB
- Cuenta con librerías necesarias para crear diversos sistemas como los de procesamiento digital de señales.
- La programación se realiza por bloques en lugar de líneas de código
- Compatible con hardware GPIB, VXI, PXI, RS-232, RS-485.

1.2.1.3 SciPy

Es una biblioteca que agrupa software de código abierto que utiliza lenguaje Python el cual permite al usuario manipular y visualizar datos utilizando comandos de alto nivel. Está compuesto de una serie de paquetes que permiten desarrollar aplicaciones científicas y de ingeniería entre los que se puede destacar (SciPy, 2016, https://www.scipy.org/about.html):

- NumPy: permite realizar cálculo numérico entre matrices,
- Biblioteca SciP: Agrupa algoritmos de cálculo numérico y herramientas utilizadas en el procesamiento de señales, estadística, optimización, etc.
- Matplotlib: Permite graficar en 2D y 3D.

1.2.2 Procesamiento digital de voz

Se encarga específicamente del estudio de las técnicas aplicadas en el procesamiento de la voz, este tipo de procesamiento engloba las siguientes categorías:

- *Reconocimiento de locutor:* Tiene como fin identificar a una persona por medio de su voz.
- *Reconocimiento de voz:* Analiza el contenido lingüístico de la señal de voz.
- *Análisis de voz:* Análisis con fines médicos de disfunciones vocales.
- *Codificación de voz:* Técnicas de compresión de datos y transmisión de voz.
- *Síntesis de voz:* Permite crear voz artificial por medio de un ordenador.
- *Mejoramiento de la señal de voz:* Técnicas para el mejoramiento.

1.2.2.1 Reconocimiento de voz

El reconocimiento de la voz permite la comunicación hombre máquina, tomando en cuenta una serie de parámetros de la voz que permitan realizar esta tarea, entre los que se destacan: estilo y modalidad de habla, entrenamiento, dimensión del vocabulario, tipo de lenguaje, etc.

El fin de los sistemas de reconocimiento de voz es reconocer al hablante en todo momento dejando atrás factores externos que dificultan esta tarea, como por ejemplo el ruido. Estos sistemas tienen algunas modalidades:

- *Reconocimiento de Palabras Aisladas (RPA):* Permite reconocer palabras específicas por medio de un entrenamiento que consiste en la comparación de la palabra a reconocer con una base de datos previamente ingresada.
- *Detección de Palabras Clave (DPC):* permite reconocer palabras claves dentro de un grupo de fonemas.
- *Reconocimiento de Palabras Conectadas (RPC):* Busca realzar el reconocimiento de un grupo pequeño de secuencias de palabras como por ejemplo oraciones.
- *Reconocimiento Automático del Habla Continua (RAHC):* Este tipo de reconocimiento tiene como fin reconocer un flujo continuo de palabras es decir una conversación humana normal, por lo cual este es el más complicado.

Todos los sistemas de reconocimiento mencionados anteriormente pueden clasificarse además acorde a la dependencia o no del locutor en:

- *Sistemas independientes del locutor:* Como su nombre lo indica este tipo de sistema se enfoca en el reconocimiento de una población en general no de un locutor en específico.
- *Sistemas dependientes del locutor:* Este tipo de sistema permite realizar el reconociendo de una persona en específico.
- *Sistemas adaptables:* Se van adaptando a un locutor determinado mientras el sistema está funcionando.

1.2.2.2 Proceso del reconocimiento de voz

El reconocimiento de voz tiene como objetivo crear una interfaz hombre-máquina por medio de la utilización de un software que realice el procesamiento respectivo de la señal, para lo cual debe cumplir con tres tareas esenciales:

1. Pre-procesamiento: Agrupa los procesos que permiten convertir la señal de entrada de la voz de tal manera que el reconocer pueda realizar su procesamiento.
2. Reconocimiento: Identifica lo que el locutor dijo.
3. Comunicación: Trasmite el patrón reconocido al sistema de hardware o software que lo requiera para alguna aplicación.

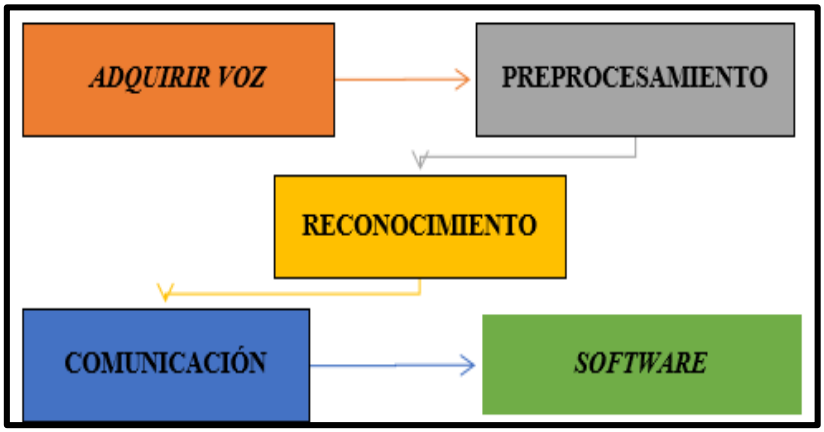


Figura 3-1 Proceso del reconocimiento de voz.

Fuente: Apolo K, Coba N, 2016.

Es importante recalcar que cada uno de las tareas que se pueden observar en la Figura 3-1 engloba además una serie de procesos que permite que el reconocedor mejore su fiabilidad.

1.3 Métodos y Etapas para el Reconocimiento de Voz

1.3.1 Representación de la señal de voz en el dominio del tiempo y la frecuencia

La representación de una señal en el dominio del tiempo permite visualizar la variación de la amplitud de la señal respecto al tiempo. Las características de la señal de la voz en este tipo de representa con son estacionarias.

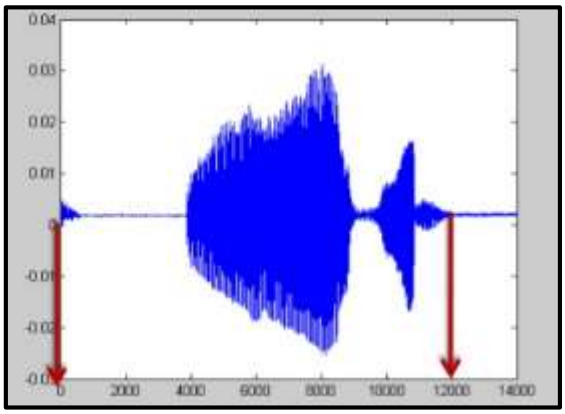


Figura 4-1 Gráfica de una señal de voz en el dominio del tiempo.

Fuente: http://es.slideshare.net/ABEL170/analisis-espectral-en-matlab-7757614

La representación de una señal en el dominio frecuencia permite visualizar las componentes de dicha señal según la frecuencia de oscilación. Esta representación espectral contiene además información referente al desplazamiento de fase aplicado a cada frecuencia de modo que se pueda recuperar la señal original. Para este tipo de representación se usan series de Fourier.

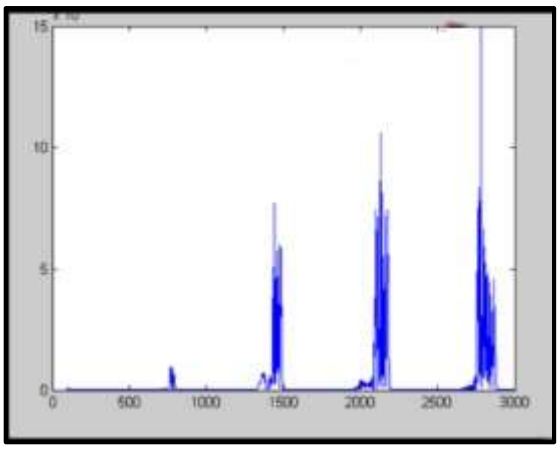


Figura 5-1 Gráfica de una señal de voz en el dominio del tiempo.

Fuente: http://es.slideshare.net/ABEL170/analisis-espectral-en-matlab-7757614

1.3.1.1 La transformada de Fourier

Es una herramienta matemática utilizada para representar una señal en el dominio frecuencia partiendo de un dominio temporal, para lo cual descompone la señal en frecuencias múltiplos (armónicos) de la frecuencia fundamental mediante la suma ponderada de funciones sinusoidales.

Se la representar como:

$$S(w) = \frac{1}{L} \sum_{n=´-\infty}^{n=\infty} s(n) e^{-jwn}(l) \quad (1\text{-}1)$$

$$s(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S(w)\, e^{jwn} \quad (1\text{-}2)$$

donde:

S(w): Representación de la respuesta en frecuencia
s(n): Representación de una secuencia en el tiempo

1.3.1.2 La transformada de Z

La transformada Z se enfoca el análisis de sistemas discretos invariantes en el tiempo (LTI), para lo cual caracteriza las señales por medio de la utilización de polos y ceros. Este tipo de transformada permite representar una señal en el dominio de frecuencia compleja (Soria et al., 2003, p.2).

Se la puede definir matemáticamente como:

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n} \quad (1\text{-}3)$$

donde:

x(n): Señal discreta
z: Variable compleja defina por:

$$X(z) = e^{-j\Omega} \quad (1\text{-}4)$$

donde:

r: Magnitud de z

Ω: Angulo de z

Otra forma de denotar esta esta transformada es la mostrada en la Ecuación 1-5.

$$X(z) = Z|x(n)| \qquad (1\text{-}5)$$

Donde la relación entre x(n) y X (z) puede denotarse como:

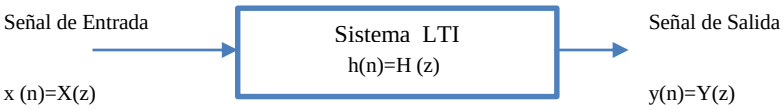
$$x(n) \overset{z}{\leftrightarrow} X(z) \qquad (1\text{-}6)$$

Dado a que la transformada z puede expresarse como las sumas de series de potencias negativas de z, solo está determinada para aquellos valores para los cuales dicha suma converge, por lo tanto la region de corvergencia esta determinada por:

$$X(z) = \left| \sum_{n=-\infty}^{\infty} x(n) z^{-n} \right| < \infty \qquad (1\text{-}7)$$

En la Ecuacion 1-7 X(z) es finita para todos el conjunto de valores de z.

Con la Transformada Z se puede establecer la función de transferencia de un sistema lineal e invariante en tiempo discreto que es la relación entre la transformada Z de la salida del sistema y la transformada Z de la entrada del sistema con condiciones iniciales nulas.



$$H(z) = \frac{Y(z)}{X(z)} \qquad (1\text{-}8)$$

donde:

Y(z): transformada Z salida del sistema

X(z): transformada Z entrada del sistema

1.3.2 Adquisición de la voz

La adquisición de la señal de voz para realizar el reconocimiento se la realiza generalmente por medio de un micrófono el cual tiene la función de convertir las ondas sonaras en energía eléctrica para que puedan ser procesadas digitalmente, la calidad y fiabilidad de un micrófono se miden acorde con sus características.

1.3.2.1 Características esenciales de un micrófono

Sensibilidad: Determina técnicamente la eficiencia del micrófono por medio de la relación del voltaje de salida del micrófono en función de la presión acústica. La sensibilidad es directamente proporcional al voltaje del micrófono es decir cuanto mayor sea el voltaje mayor será la sensibilidad.

Fidelidad: Es la respuesta en frecuencia expresa en dB que proporciona un micrófono, la cual está determinada por la variación de la sensibilidad con respecto a la frecuencia.

Directividad: Determina el ángulo de cobertura que el micrófono es capaz de captar de una fuente de sonido.

Ruido de fondo o equivalente: Es la tensión base que existe en un micrófono cuando ningún sonido incide sobre él. La calidad del micrófono es inversamente proporcional al ruido de fondo es decir que un buen micrófono tiene bajo ruido de fondo. Este ruido es medido en dB y está alrededor de los 60dB.

Rango dinámico: Se determina a partir de la relación S/N (señal a ruido) y el nivel máximo de presión acústica admisible por el micrófono. Este rango dinámico determina el margen en el cual el micrófono trabaja de forma óptima.

Impedancia interna: Es la resistencia al paso de la corriente que tiene el micrófono, según su valor puede denominarse como impedancia baja (entre 200 Ω), impedancia alta (600 Ω,1000 Ω o 2000 Ω) y muy alta impedancia (mayor a 3000 Ω). El valor de esta impedancia toma importancia al momento de evitar perdida de la señal para lo cual si el micrófono es de baja impedancia se debe usar un cable de mayor longitud y viceversa.

1.3.2.2 Clasificación de los micrófonos

Existen un sin número de clasificaciones de los micrófonos, en las Figuras 6-1, 7-1 y 8-1 se puede observar las clasificaciones más comunes y características de acuerdo a:

1. Directividad
2. Transductor
3. Utilidad

Dependiendo la utilidad que se le vaya a dar a un micrófono se elige es que más se adapte a los requerimientos.

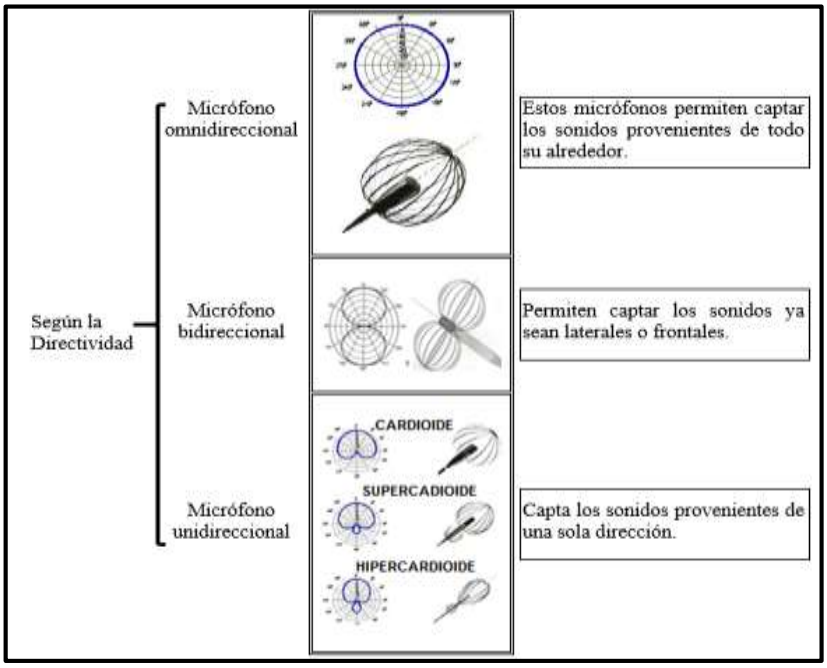


Figura 6-1 Clasificación de los micrófonos según la Directividad.

Fuente: Apolo K, Coba N, 2016.

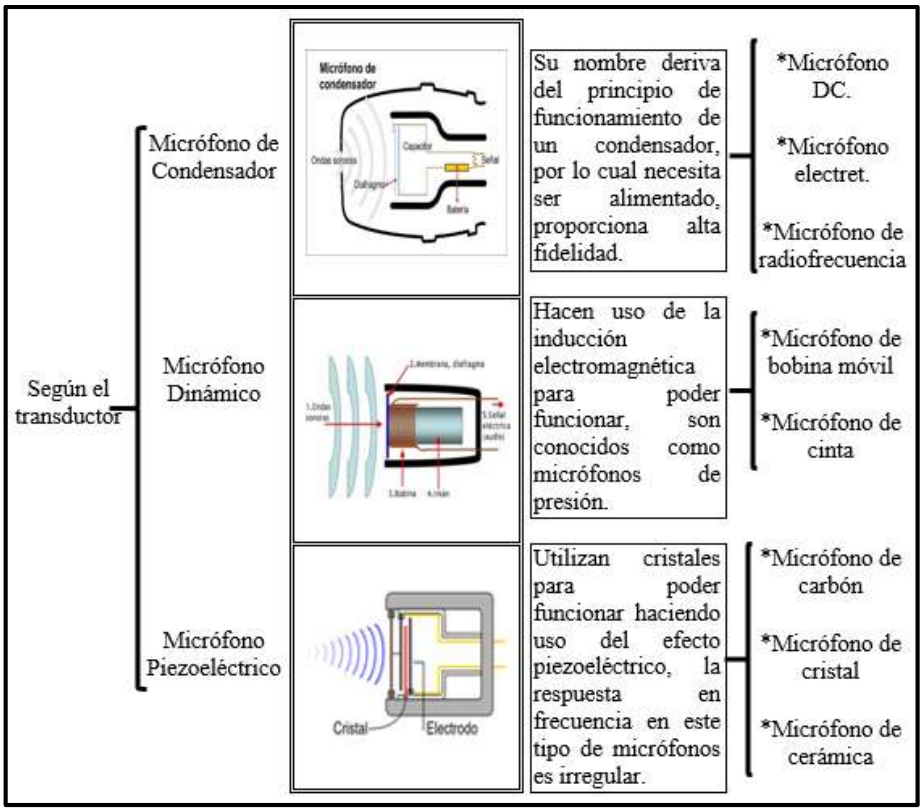


Figura 7-1 Clasificación de los micrófonos según el Transductor.

Fuente: Apolo K, Coba N, 2016.

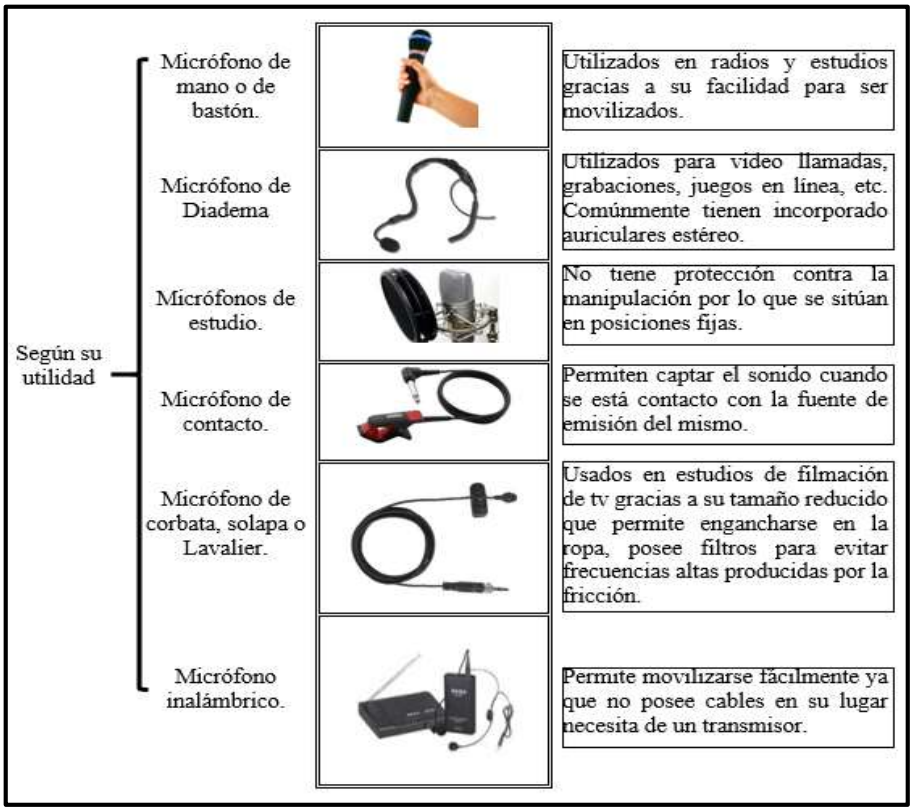


Figura 8-1 Clasificación de los micrófonos según la utilidad.

Fuente: Apolo K, Coba N, 2016.

1.3.3 Digitalización de la señal de voz

La voz es una señal analógica (señal continua) por lo cual para poder ser procesada por algún software debe ser digitalizada (señal discreta). En un reconocedor la voz es captada por medio de un micrófono el cual convierte las ondas sonoras en energía eléctrica para que esta pueda codificada por medio de una tarjeta de audio y digitaliza para de esta manera ser procesada en un ordenador (Soria et al., 2003, p.2).

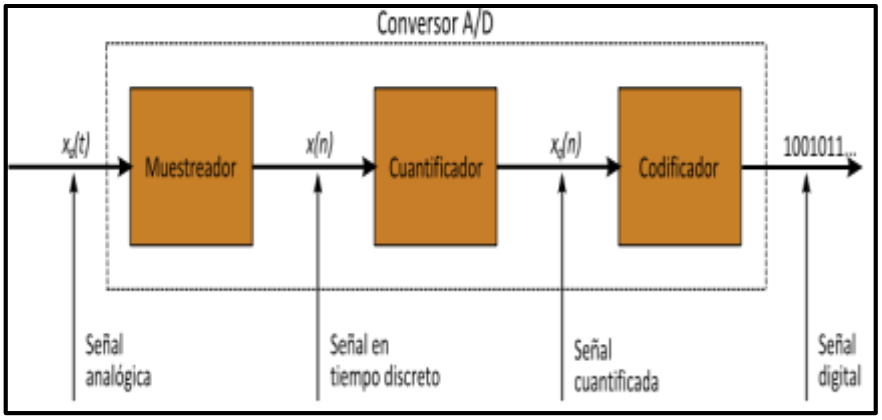


Figura 9-1 Proceso de conversión A/D.

Fuente: https://commons.wikimedia.org/wiki/File:Conversor_AD.svg.

Los tres procesos necesarios para transformar una señal continua en discreta son:

1. Muestreo
2. Cuantificación
3. Codificación

1.3.3.1 Muestreo

El muestreo de señales analógicas se debe dar bajo condiciones que aseguren que no existan perdidas de información, existen diferentes formas de muestrear una señal de las cuales la más usada es el muestreo uniforme en el cual la cadencia de toma de muestras es constante (Soria et al., 2003, p.2).

Para este caso se define:

$$x(n) = x_a(t) = x_a(nT), -\infty < n < \infty \qquad (1\text{-}9)$$

donde:

$x_a(t)$: Señal analógica de donde se toman las muestras cada T (s).
$x(n)$: Señal en tiempo discreto.
T: Intervalo de tiempo entre dos muestras (periodo)

La relación de muestreo se define como:

$$t = nT = \frac{n}{Fs} \quad (1\text{-}10)$$

donde:

F_s: Frecuencia de muestreo.
t: Variable de tiempo continuo.
n: Variable de tiempo discreto.

Es importante considerar que además podemos establecer una relación entre la frecuencia analógica (F o Ω) y la frecuencia digital (f o ω), para lo cual si tenemos una señal analógica de la forma:

$$x_a(t) = A \cdot \cos(2 \cdot \pi \cdot F \cdot t + \theta) \quad (1\text{-}11)$$

Y si muestreamos periódicamente a una frecuencia $F_s = \frac{1}{T}$, dará lugar a:

$$x_a(t) = x(n) = A \cdot \cos(2 \cdot \pi \cdot F \cdot n \cdot T + \theta) \quad (1\text{-}12)$$

$$x_a(t) = A \cdot \cos\left(\frac{2 \cdot \pi \cdot F \cdot n}{F_s} + \theta\right) = A \cdot \cos(2 \cdot \pi \cdot f \cdot n \cdot + \theta) \quad (1\text{-}13)$$

De la Ecuación 1-13 se puede deducir que $f = \frac{F}{f_s}$, esta relación es conocida como la frecuencia normalizada o relativa.

Para determinar cada que tiempo se debe tomar las muestras para digitalizar una señal se hace uso del **Teorema de muestreo de Nyquist** el cual establece que se debe muestrear una señal al menos al doble de la frecuencia máxima contenida en la misma para de esta forma evitar aliasing el cual crea falsas componentes de la señal (Soria et al., 2003, p.4).

$$f_s \geq 2f_m \qquad (1\text{-}14)$$

1.3.3.2 Cuantificación

La cuantificación es el proceso de conversión de una señal en tiempo discreto que posee valores continuos en una señal en tiempo discreto que posea valores discretos, este proceso es irreversible por lo cual al ser cuantificada una señal no se podrá obtener los valores originales debido a que se asigna un mismo valor discreto a distintos valores continuos.

El proceso de cuantificación mostrado en la Figura 10-1 esta denotado por la siguiente expresión:

$$x_q(n) = Q[x(n)] \qquad (1\text{-}15)$$

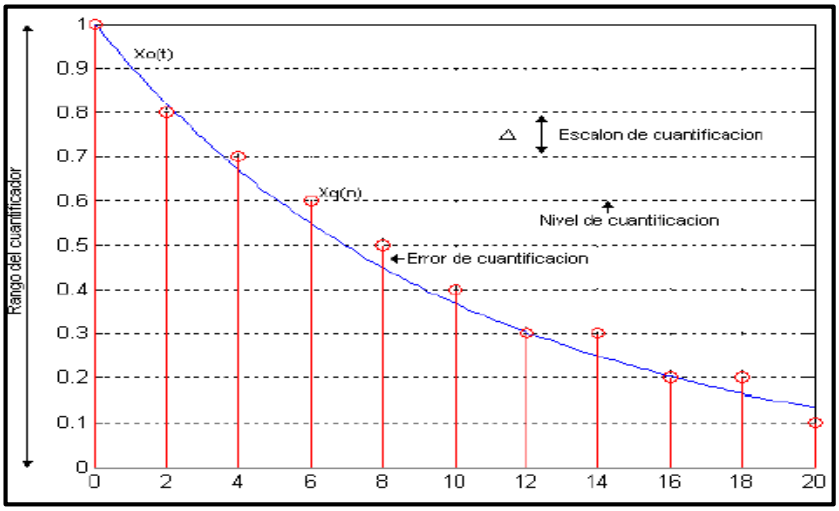


Figura 10-1 Cuantificación.

Fuente: J. Vivanco et al, 2010, Ilustración de la cuantificación, p.32.

El error de cuantificación está dado por:

$$e_q(n) = x_q(n) - x(n) \qquad (1\text{-}16)$$

donde:

$e_q(n)$: Error de cuantificación.

$x_q(n)$: Valor cuantificado.

$x(n)$: Valor de la muestra original.

Existen diversas técnicas de cuantificación (Velásquez, 2008, pp.34-35):

Cuantificación uniforme: Esta cuantificación lineal representada en la Figura 11-1 considera la misma distancia entre niveles de cuantificación. Debido a su efectividad no es muy usada, sin embargo son económicos y fáciles de implementar.

Cuantificación no uniforme: Esta cuantificación mostrada en la Figura 12-1 es también denominada como no lineal debido a que se utiliza cuando se tienen dos señales que no son homogéneas por lo cual la distancia entre niveles de cuantificación es variable, mientras más cercanos estén entre si los niveles de cuantificación la distancia será más pequeña.

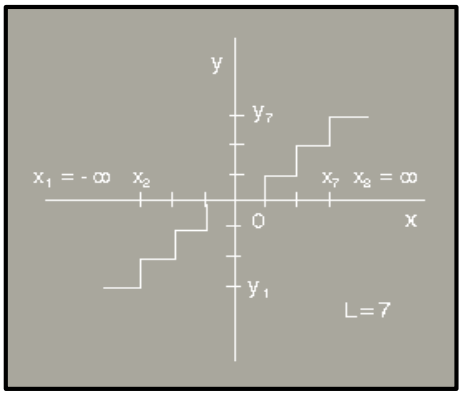


Figura 11-1 Cuantificación Uniforme.

Fuente: http://ceres.ugr.es/~alumnos/luis/mycuan.htm#logaritmica

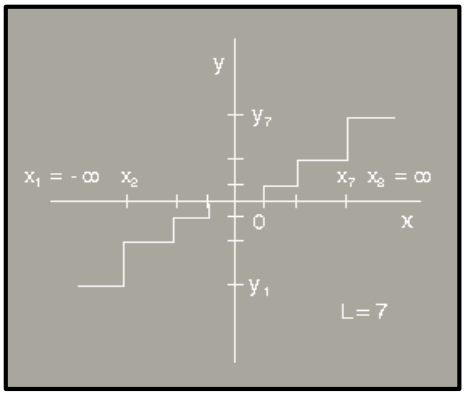


Figura 12-1 Cuantificación No-Uniforme.

Fuente: http://ceres.ugr.es/~alumnos/luis/mycuan.htm#logaritmica

Cuantificación logarítmica: Este tipo de cuantificación escalar digital antes de realizar el proceso de cuantificación pasa la señal por un compresor logarítmico con el fin de disminuir el error de cuantificación y después de realizar el proceso de cuantificación pasa la señal por un expansor, garantiza además que las distancias entre los niveles de cuantificación sea constante.

Cuantificación vectorial: Esta cuantificación vectorial digital es considerada la más eficiente, utiliza el mismo procedimiento que la cuantificación uniforme o no uniforme con la diferencia que la cuantificación se realiza para un número determinado de muestras (bloques).

1.3.3.3 Codificación

La codificación asigna a cada nivel de cuantificación diferente un código preestablecido, el código más usado para codificar es el binario (Vivanco et al., 2010, p.32).

Para poder realizar la codificación se debe cumplir que la cantidad de números binarios diferentes debe ser mayor o igual que el número de niveles diferentes requeridos para la cuantificación, lo cual se expresa en la Ecuación 1-17.

$$2^b \geq \mathrm{L} \qquad (1\text{-}17)$$

Por lo cual:

$$\mathrm{b} \geq \log_2 L \qquad (1\text{-}18)$$

donde:

L: Niveles
b: Número de bits

1.3.4 Pre-procesamiento

El objetivo del pre-procesamiento es eliminar parámetros de la señal de voz que no contribuyan en el procesamiento es decir la eliminación del ruido y de periodos en silencios permitiendo de esta manera obtener una señal que solo contenga información relevante.

1.3.4.1 Segmentación y enventanado de la señal

Al ser la señal de voz considerada como no estacionaria y aleatoria se necesita dividirla en pequeños segmentos dentro de los cuales se considera que la señal es estacionaria, cada segmento tiene por lo general un tiempo de duración de 20ms a 40ms, de esta manera se agrupan segmentos que tienen características similares lo que permite reducir el tiempo de análisis.

Al proceso de multiplicar un segmento por una función limitada en el tiempo para eliminar valores que no se encuentren en dicho intervalo se denomina ventaneo, generalmente se busca que las ventanas contiguas estén solapadas entre sí para de esta manera evitar pérdidas de información y garantizar continuidad.

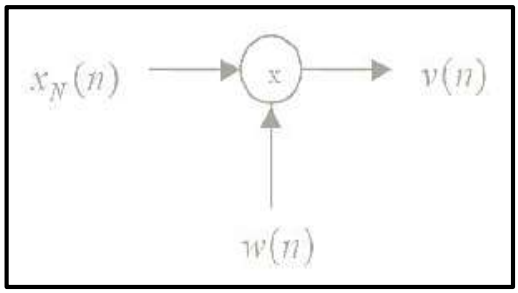


Figura 13-1 Enventanado.

Fuente: L. Rojo, 2011, Enventanado, p.47.

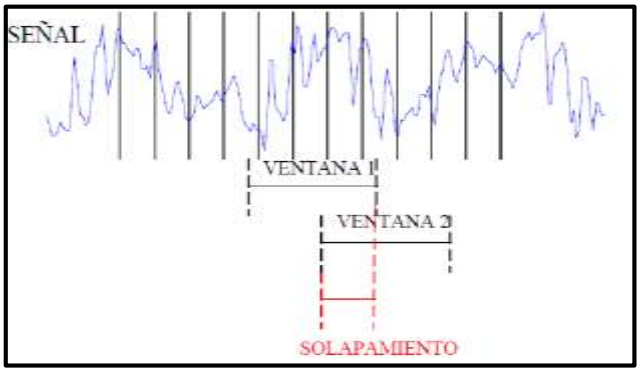


Figura 14-1 Solapamiento de una señal.

Fuente: L. Rojo, 2011, Ilustración del solapamiento de una señal, p.47.

En la Figura 14-1 se observa la agrupación de un determinado número de muestras (N) de una señal en segmentos $x_N(n)$ para su posterior multiplicación por una ventana w(n).

Es importar destacar que para que se mantenga la potencia de la señal antes y después del enventanado se debe normalizar. La normalización consiste en aplicar una ganancia a un audio de tal forma que la amplitud del pico promedio alance un valor equitativo.

En la Tabla 1-1 se especifican los diferentes tipos de ventanas y sus ecuaciones acorde al número de muestras.

Tabla 1-1 Tipos de ventanas

VENTANA	ECUACIÓN
Rectangular	w(n)=1
Hanning	$\mathrm{w(n)} = \frac{1}{2} - \frac{1}{2} \cdot \cos(\frac{2\pi n}{N})$
Hamming	$\mathrm{w(n)} = \frac{27}{50} - \frac{23}{50} \cdot \cos(\frac{2\pi n}{N})$
Barlett	$\begin{cases} \frac{2n}{N}, & 0 < n < \frac{N}{2} \\ 2 - \frac{2n}{N}, & \frac{N}{2} < n < N \end{cases}$
Blackman	$\mathrm{w(n)} = \frac{21}{50} - \frac{1}{2} \cdot \cos\left(\frac{2\pi n}{N}\right) + \frac{2}{25} \cdot \cos\left(\frac{4\pi n}{N}\right)$

Fuente: http://www.monografias.com/trabajos20/enventanado/enventanado.shtml.

1.3.4.2 Segmentado de tramas

El segmentado de tramas permite identificar las partes donde se tiene información relevante y eliminar los silencios producidos por el ruido.

Para realizar este segmentado se utilizan tres métodos (Moral et al., 2011, pp.26-28):

- Cálculo de energía
- Tasa de cruces por cero (ZCR)
- Cálculo del pitch.

1. Cálculo de energía

Este método nos permite eliminar silencios indeseados ya que por medio del cálculo de la energía de la señal se puede realizar la comparación entre segmentos sonoros y periodos de silencios, considerando que los segmentos sonoros tienen mayor energía.

La energía promedio de un segmento (E_s) está representada por:

$$E_s = \frac{1}{N} \sum_{n=1}^{N} s^2(n) \qquad (1\text{-}19)$$

donde:

N: Tamaño del segmento

La energía promedio de una señal se representa:

$$E_T = \frac{1}{L}\sum_{l=1}^{L} x^2(l) \quad (1\text{-}20)$$

donde:

L: Longitud de toda la señal

2. Tasa de cruces por cero (ZCR)

Una señal de audio esta compuestos de valores positivos y negativos previo a la eliminación de componentes DC. Esta forma de eliminación de silencios consiste en obtener la tasa de cruces por cero que tiene una señal en determinado segmento la cual va a ser un factor de decisión en la identificación de silencios ya que en segmentos sonoros esta tasa es alta mientras que en segmentos no sonoros es pequeña.

Esta tasa de cruces por cero (z) se puede calcular mediante:

$$z = \sum_{m=0}^{N-1} |sgn[x(m)] - sgn[x(m-1)]| \quad (1\text{-}21)$$

donde:

N: Numero de muestras

sgn: Función signo defina por:

$$sgn(x) = \begin{cases} +1, & x \geq 0 \\ -1, & x < 0 \end{cases} \quad (1\text{-}22)$$

3. Cálculo del pitch.

En este método se extrae la frecuencia fundamental de la señal esta será analizada utilizando métodos de detección de la misma, es decir donde se detecte esta frecuencia fundamental se asume que se tiene una trama sonora.

1.3.5 Filtrado de preénfasis

Se le da el nombre de preénfasis al proceso mediante el cual se garantiza que el espectro tenga un margen dinámico equivalente en toda la banda de frecuencias para lo cual se compensa la atenuación de -6dB/octava (-12dB/octava espectro de la señal glótica y +6dB/octava radiación de los labios en frecuencias bajas). Este filtro tiene como función principal acentuar las frecuencias altas (consonantes) y realiza una comprensión del rango dinámico para de esta manera compensar la manera en que el tracto vocal filtra estas frecuencias.

En el PDS se implementa pre-énfasis usando filtros:

- Analógicos pasa altos de primer orden con frecuencia de corte a 3dB
- Digitales pasa altos de primer orden

La implementación de filtros digitales pasa altos de primer orden usa la ecuación:

$$y(n)=x(n)-ax(n-1) \qquad (1\text{-}23)$$

donde:

y(n) : Muestra actual de salida del filtro
x(n): Muestra actual de entrada al filtro
x(n-1): Muestra anterior de entrada al filtro
a: Constante de amplificación entre $0.9 \leq a \leq 1$

Si evaluamos la transformada z de la Ecuación 1-23:

$$Y(z)=X(z)-az^{-1}X(z)=(1-az^{-1})X(z) \qquad (1\text{-}24)$$

La función de transferencia del filtro será entonces:

$$\mathrm{H(z)} = \frac{Y(z)}{X(z)} = 1\text{-}az^{-1} \qquad (1\text{-}25)$$

donde:

z^{-1}: Operador de retardo de muestra unitario

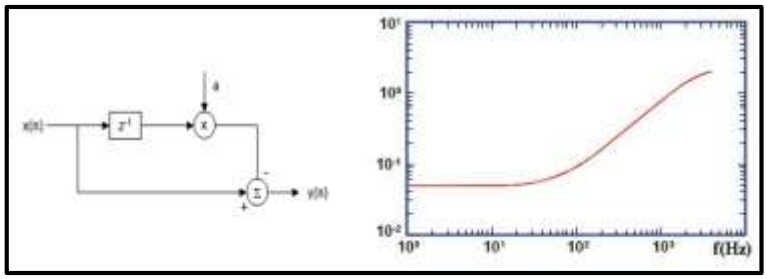


Figura 15-1 Filtro Preénfasis

Fuente: Apolo K, Coba N, 2016.

1.3.6 Reconocimiento: Extracción de características

La etapa de reconocimiento implica la extracción de parámetros que definan o caractericen a la señal, para lo cual se utilizan diversas técnicas (Oropeza y Suarez, 2006, p.273).

- Análisis de Fourier.
- Codificación Predictiva Lineal.
- Análisis de los coeficientes Cepstrales.
- Predicción Lineal Perceptiva.

1.3.6.1 LPC (Codificación Predictiva Lineal)

1.3.6.1.1 Envolvente LPC

Para determinar la envolvente de la señal LPC se utiliza el Modelo Autorregresivo (AR) o todo-polo que describe la función de transferencia de un tubo formado por secciones distintas, esta técnica es una de las más usadas en el análisis de la voz ya que permite representar en forma

comprimida la envolvente de una señal digital, para lo cual modela la fuente de sonido como un filtro con "n" polos. (Velásquez, 2008, p.38).

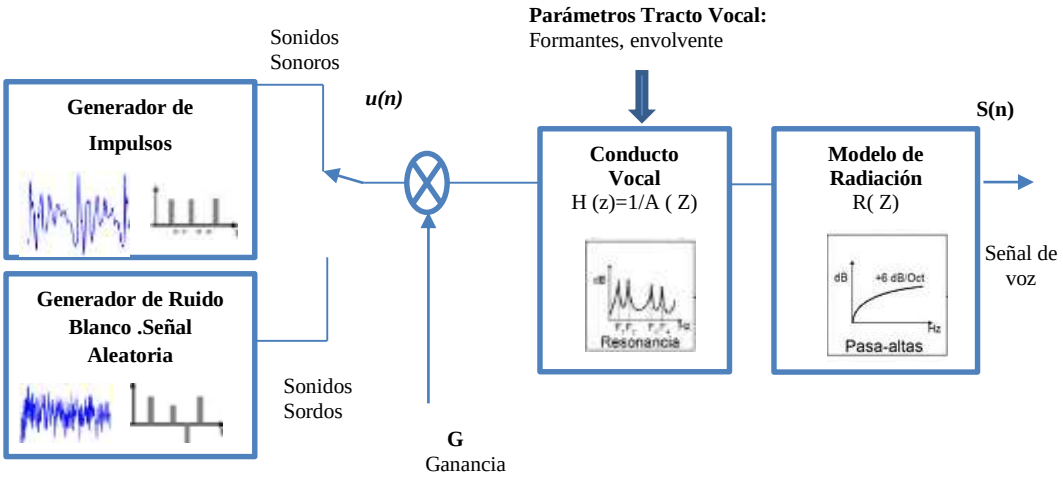


Figura 16-1 Modelo AR

Fuente: Apolo K, Coba N, 2016.

Matemáticamente esta técnica representa la señal de voz como la sumatoria de muestras pasadas, así se tiene que:

$$s(n) = -\sum_{k=1}^{p} \alpha_k \, s(n-k) \qquad (1\text{-}26)$$

donde:

s(n):	Señal de voz
$s(n-k)$:	Muestras pasadas de la señal de voz
k:	Número total de parámetros LPC
α_k:	Coeficientes del filtro
ρ:	Orden del filtro

Si a la Ecuación 1-26 le añadimos una ganancia de excitación G$u(n)$ que depende la naturaleza de la señal tendremos:

$$\mathrm{s(n)} = -\sum_{k=1}^{p} \alpha_k \, s(n-k) + \mathrm{G}^{*} u(n) \qquad (1\text{-}27)$$

A su vez podemos representar la Ecuación 1-27 en dominio Z lo que dará como resultado:

$$\mathrm{S(z)} = -\sum_{k=1}^{p} \alpha_k \, z^{-k} S(z) + G \, U(z) \tag{1-28}$$

La Ecuación 1-28 nos lleva a la función de transferencia:

$$\mathrm{H(z)} = \frac{S(z)}{U(z)} = \frac{G}{1 + \sum_{k=1}^{p} \alpha_k \, z^{-k}} \tag{1-29}$$

$$\mathrm{H(z)} = \frac{S(z)}{G \, U(z)} = \frac{1}{1 + \sum_{k=1}^{p} \alpha_k \, z^{-k}} = \frac{1}{A(z)} \tag{1-30}$$

En la Ecuación 1-30 podemos ver la función de transferencia $\mathrm{H(z)}$ la cual permite representar parámetros del tracto vocal.

Para la determinación de los formantes de la señal de voz se puede obtener a través de dos métodos: Método de Raíces y Método Peak-Picking.

En el método de Raíces hace referencia a la etapa de la función de transferencia $\mathrm{H(z)} = \frac{1}{A(z)}$ en el cual se puede distinguir diversas frecuencias, las cuales representan dichos formantes. Las raíces ocurren en pares complejos conjugados y cada par de polos complejos conjugados corresponde a un formante. Los formantes se obtiene a partir de las raíces de $A(z) = 0$.

$$A(z) = 1 + \sum_{k=1}^{p} \alpha_k \, z^{-k} = 0 \tag{1-31}$$

$$raiz = x + iy \quad r = \sqrt{x^2 + y^2} \quad , \Theta = tan^{-1}\left(\frac{y}{x}\right) \tag{1-32}$$

$$Fi = \frac{\theta_i F_s}{2\pi} \tag{1-33}$$

El Método Peak-Picking es una aproximación que hace referencia a los picos de la envolvente espectral de $H(z) = \frac{1}{A(z)}$.

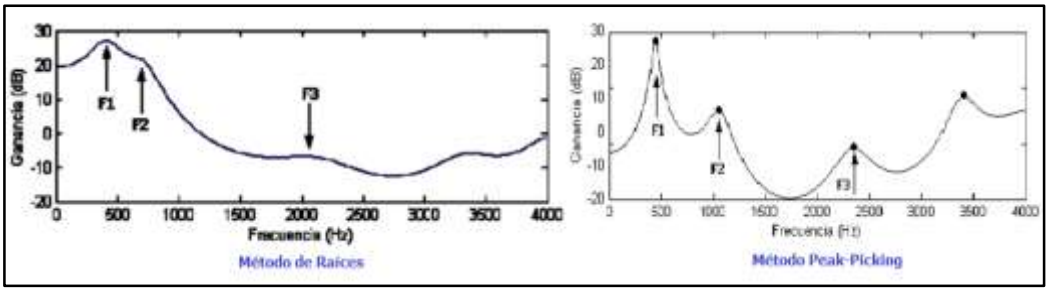


Figura 17-1 Método de Raíces y Método Peak-Picking

Fuente: Apolo K, Coba N, 2016.

El modelo de radiación describe la impedancia de radiación vista por la presión de aire cuando abandona los labios. Corresponde a un filtro pasa alto de primer orden (6dB/octava).

$$\mathrm{R}(Z) = (1 - \alpha_k z^{-1}) \qquad (1\text{-}34)$$

1.3.6.1.2 Predicción de Error

Se utiliza la Predicción de Error para encontrar las características de las señal que son de gran importancia para el reconocimiento de voz, debido que se tiene la señal de voz obtenida a través del micrófono, uno de los inconvenientes que se presenta es encontrar las característica que se forman en el tarto vocal para ello es necesario realizar un análisis inverso para encontrar dichas características corresponden a los coeficientes (α_k).

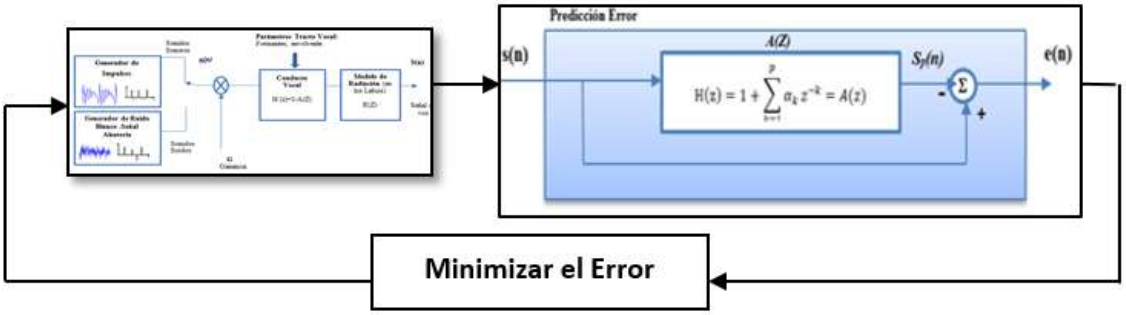


Figura 18-1 Modelo AR y Predicción de Error

Fuente: Apolo K, Coba N, 2016.

Se tiene la señal de voz s(n) para lo cual se puede establecer una señal de predicción o predicha $s_p(n)$ con valores previos es decir valores que se aproximen a la señal de voz en donde el error sea el menor posible.

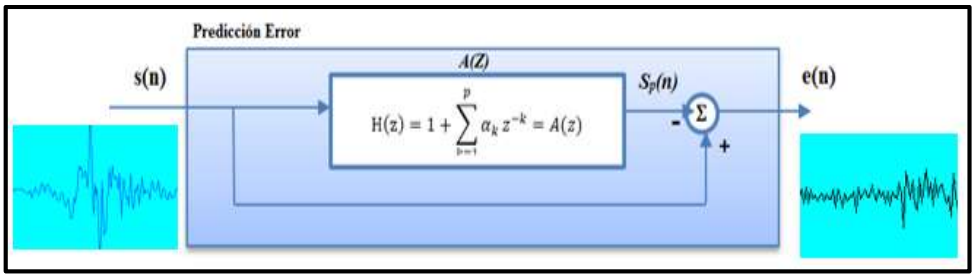


Figura 19-1 Predicción de Error

Fuente: Apolo K, Coba N, 2016.

H(z) corresponde al Filtro de Predicción que es el filtro Inverso de $H(z) = \frac{1}{A(z)}$, entonces este filtro corresponde a $H(z) = A(z)$.

$$s(n) = -\sum_{k=1}^{p} \alpha_k \, s(n-k) + Gu(n) \quad (1\text{-}35)$$

$$H(z) = 1 + \sum_{k=1}^{p} \alpha_k \, z^{-k} = A(z) \quad (1\text{-}36)$$

$$s_p(n) = -\sum_{k=1}^{p} \alpha_k \, s(n-k) \quad (1\text{-}37)$$

La señal s(n) pasa por el Filtro de Predicción $H(z) = A(z)$ y se obtiene la señal de predicción $s_p(n)$. El error de predicción de una señal se representa matemáticamente mediante la siguiente expresión, esta señal representa la excitación de la fuente del tracto vocal es decir **e(n) = G u(n)**.

$$e(n) = s(n) - s_p(n) = s(n) + \sum_{k=1}^{p} \alpha_k \, s(n-k) \quad (1\text{-}38)$$

donde:

s(n): Señal de voz

$s_p(n)$: Señal de voz predicha

Mediante el método de los mínimos cuadrados y la autocorrelación se puede determinar los coeficientes de predicción minimizando el error cuadrático medio (Agardoña, 2008, p.25).

$$\mathrm{E} = \sum_{n} e^2\,(\mathrm{n}) = \sum_{n} [s(n) + \sum_{k=1}^{p} \alpha_k\, s(n-k)]^2 \qquad (1\text{-}39)$$

Para minimizar el Error se deriva E para cada α_k y se iguala a cero.

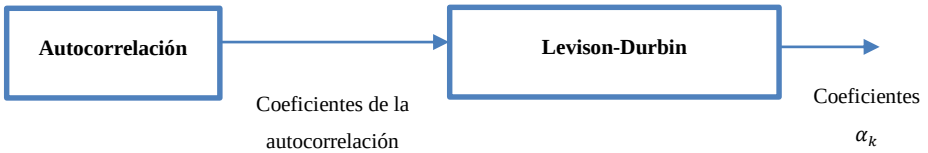
$$\frac{\partial E}{\partial\, \alpha_i} = 0 \qquad 1 \le i \le p \qquad (1\text{-}40)$$

$$2 \sum \{s(n) + \sum_{k=1}^{p} \alpha_k\, s(n-k)\}\, [s(n-i)] = 0 \qquad (1\text{-}41)$$

$$\sum \{s(n) + \sum_{k=1}^{p} \alpha_k\, s(n-k)\}\, [s(n-i)] = \frac{0}{2} \qquad (1\text{-}42)$$

$$\sum \{s(n)[s(n-i)]\} + \sum_{k=1}^{p} \sum \{\alpha_k s(n-k)[s(n-i)]\} = 0 \qquad (1\text{-}43)$$

$$-\sum \{s(n)[s(n-i)]\} = \sum_{k=1}^{p} \alpha_k \{\sum s(n-k)[s(n-i)]\} \qquad (1\text{-}44)$$



Realizamos la Autocorrelación.

$$\sum \{s(n)[s(n-i)]\} = R_s\,[\,|i|\,] \qquad (1\text{-}46)$$

$$\sum s(n-k)[s(n-i)]\} = R_s[\,|i-k|\,] \qquad (1\text{-}47)$$

Reemplazado en la ecuación (1-45) se tiene:

$$-R_s[\,|i|] = \sum_{k=1}^{p} \alpha_k\, R_s[\,|i-k|] \qquad para\ i = 1,2\,\ldots.p \qquad (1\text{-}48)$$

Se obtiene un sistema de ecuaciones:

$$-R_s[\,|1|] = \alpha_1\, R_s[\,|1-1|] + \alpha_2\, R_s[\,|1-2|] + \cdots \alpha_p\, R_s[\,|p-1|] \quad (1\text{-}49)$$

$$-R_s[\,1] = \alpha_1\, R_s[\,0] + \alpha_2\, R_s[\,1] + \cdots \alpha_p\, R_s[\,|p-1|] \quad (1\text{-}50)$$

$$-R_s[\,|2|] = \alpha_1\, R_s[\,1] + \alpha_2\, R_s[\,0] + \cdots \alpha_p\, R_s[\,|p-2|] \quad (1\text{-}51)$$

…… ……

$$-R_s[\,p] = \alpha_1\, R_s[\,p-1] + \alpha_2\, R_s[\,p-2] + \cdots \alpha_p\, R_s[0] \quad (1\text{-}52)$$

donde:

R: son los coeficientes de autocorrelación

α: son los coeficientes LPC

En forma matricial se tiene:

$$-\begin{bmatrix} R_s[1] \\ R_s[2] \\ \cdots \\ R_s[p] \end{bmatrix} = \begin{bmatrix} R_s[0] & R_s[1] & \ldots & R_s[p-1] \\ R_s[1] & R_s[0] & \ldots & R_s[p-2] \\ \cdots & \cdots & \cdots & \cdots \\ R_s[p-1] & R_s[p-2] & \ldots & R_s[0] \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \cdots \\ \alpha_p \end{bmatrix} \quad (1\text{-}53)$$

-r R α

$$-\mathrm{r} = \mathrm{R}.\alpha \quad (1\text{-}54)$$

$$\alpha = -R^{-1}.\mathrm{r} \quad (1\text{-}55)$$

El método de cálculo computacional de coeficientes de predicción (α_k) más usado es el algoritmo de Levinson Durbin.

Teniendo los parámetros anteriores se puede calcular la Ganancia.

$$G^2 = R(0) - \sum_{K=1}^{P} \alpha_k\, R(k) \quad (1\text{-}56)$$

Esta expresión muestra la dependencia de la ganancia con los coeficientes de autocorrelación y de los coeficientes α_k calculado por el método de Levinson Durbin.

1.3.6.1.3 Número de Coeficientes LPC

Para obtener el número de coeficientes LPC (p) de acuerdo a la frecuencia de muestreo (f_s) se toma se consideración que el espectro de voz puede representarse con una densidad media de dos polos (1 polo por kHz), por lo cual el tracto vocal cooperara con $\frac{f_s}{1000}$ polos para poder representar el espectro de voz, sin embargo se pueden consideran de 3 a 4 polos adicionales que representaran a la fuente de excitación y radiación, entonces lo dicho anteriormente estará representado por (Cobeta, 2013, p.108):

$$\mathrm{p} = \frac{f_s}{1000} + 3\ o\ 4 \qquad (1\text{-}57)$$

Del número de coeficientes LPC depende la calidad de la envolvente por lo cual se deben considerar valores de p elevados para minimizar el error de predicción ya que con valores de p bajos el error de predicción aumenta afectando la resolución de la envolvente.

1.3.7 Reconocimiento: Comparación y decisión

De la correcta comparación de la señal de voz de referencia con la señal ingresada depende la calidad del reconocedor, esta comparación de patrones se puede realizar mediante el cálculo de la distancia entre los vectores con características específicas de la señal patrón y la señal ingresada (Velásquez, 2008, p.46).

La distancia euclidiana resulta idónea al momento de comprar dos señales, esta puede representarse como:

$$\mathrm{d} = \sum_{i=1}^{D} |f_i - f_i'|^2 \qquad (1\text{-}58)$$

donde:

f_i: Valor característico señal patrón

f_i': Valor característico señal ingresada

D: Dimensión del vector

La distancia total se calcula utilizando la siguiente expresión:

$$D_{total} = \sum_{i=1}^{n} d_i \qquad (1\text{-}59)$$

donde:

D_{total}: Distancia total
n: Dimensión del vector donde se almacenan cada una de las distancias
d: Distancia

La Ecuación 1-59 permite obtener una sumatoria total de las distancias euclidianas resultantes de la Ecuación 1-58.

1.4 Tarjetas de Adquisición de Datos

1.4.1 Placa Arduino

Arduino es una plataforma de código abierto que está basada en hardware y software libre fácil de utilizar. La Placa hardware libre incorpora un microcontrolador reprogramable y una serie de pines que permiten conectar fácilmente sensores y actuadores. La Placa hardware se refiere a una placa PBC, es decir una placa de circuitos impresos, en el mercado existen varios tipos de placas con diferentes características y tamaños. Los microcontroladores que vienen incorporados en los diferentes tipos o modelos de las placas Arduino pertenecen a la misma familia tecnológica por lo que su funcionamiento es parecido, estos microcontroladores son de tipo AVR que es una arquitectura desarrollada y fabricada por la marca Atmel (Torrente, 2013, p.64) .

Arduino Software (IDE) proporciona la facilidad de escribir el código y subirlo a la placa, utiliza el lenguaje de programación C/C++ así también puede soportar varios lenguajes de programación de alto nivel derivados de C, al ser un software libre se puede descargar de la página web oficial de Arduino que se encuentra disponible para todo público en diferentes versiones, además este software puede ser utilizado en cualquier placa de Arduino. (Arduino,2016,http://arduinodhtics.weebly.com/iquestqueacute-es.html)

Arduino ofrece una serie de ventajas entre las más destacadas se tiene las siguientes (Arduino, 2016, https://www.arduino.cc/en/Guide/Introduction):

- *Asequible:* Tiene un bajo costo en comparación con otras plataformas de microcontroladores.
- *Multiplataforma:* Se puede ejecutar en Windows, Macintosh OS X, y Linux.
- *Simple:* El software de Arduino (IDE) es fácil de usar para personas que no tienen conocimiento en programación.
- *Código abierto y el software extensible:* El software de Arduino de basa en una herramienta de código abierto. El idioma se puede ampliar a través de bibliotecas de C ++, y la gente que quiere entender los detalles técnicos pueden dar el salto de Arduino para el lenguaje de programación C AVR en la que se basa.
- *Código abierto y hardware ampliable:* Las placas Arduino se publican bajo una licencia de Creative Commons, donde diseñadores de circuitos electrónicos pueden construir su propio módulo ampliarlo e incluso mejorarlo. Los usuarios que tienen poca experiencia pueden construir la placa con la finalidad de entender el funcionamiento.

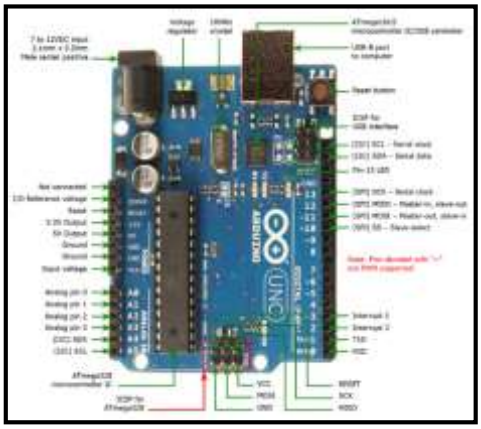


Figura 20-1 Partes de la placa Arduino Uno.

Fuente: (https://aprendiendoarduino.wordpress.com/)

1.4.2 Tarjeta Galileo

La tarjeta Galileo es una placa basada en el procesador Quark SoC X1000 de 32bits de Intel de desarrollo Open Hardware con una velocidad de 400MHz. El hardware y software de esta placa han sido diseñados para ser compatible con el IDE de Arduino y con Shields de Arduino para el Uno R3. Esta tarjeta tiene entrada para mini-PCI Express, conector Ethernet, un zócalo MicroSD, USB Host, puerto serie RS-232 y 8Mbyte de memoria NOR flash (Arduino,2016,http://arduino.cl/intel-galileo/).

Una de las diferencias entre la placa Galileo y Arduino es la compatibilidad de hardware y el software con el sistema operativo Linux, ya que se puede controlar sensores o motores con otros lenguajes de programación, crear un servidor y poder conectarlos a Internet, esta placa Galileo

está enfocada más a proyectos referente al Internet de las Cosas (DIYMakers, 2016, http://diymakers.es/intel-galileo-guia-inicial/).

Figura 21-1 Tarjeta Galileo.

Fuente: (https://www.arduino.cc/en/ArduinoCertified/IntelGalileo)

1.4.3 Tarjeta Raspberry Pi

Figura 22-1 Tarjeta Raspberry Pi.

Fuente: (www.diotronic.com/zmaker/raspberry_pi.html)

La tarjeta Raspberry Pi es ordenador de placa reducida de bajo costo (SBC) de tamaño de una tarjeta de crédito, que adopta la forma de una PCB, desarrollado por la Fundación Raspberry Pi en el Reino Unido esta fundación ha elegido el sistema operativo Linux, lo que permite a los usuarios beneficiarse de las diferentes aplicaciones y herramientas de código abierto .Es compatible con las distribuciones Debian y Fedora y, a través de GitHub2, que proporciona el código fuente necesario para crear un núcleo para la Raspberry Pi, la Fundación ha decidido destacarse por Python como el lenguaje preferido para la educación (eTech,2016,p.8,http://etech.designspark.info/ELE_0050_eTech%2010/ELE_0050_eTech_ES/pubData/source/ELE_0050_eTech_ES.pdf).

1.5 Módulos de Comunicación Inalámbrica

1.5.1 Modulo Bluetooth

Módulo Bluetooth es un dispositivo de comunicación inalámbrica el cual permite transmitir datos a través de radiofrecuencia en la banda de 2,4 GHZ con un alcance entre 5 a 10 metros.

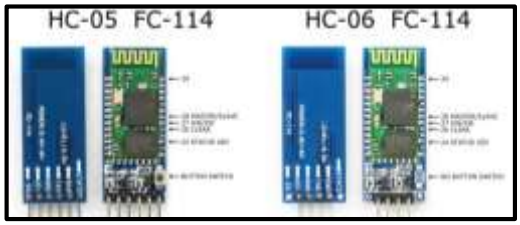


Figura 23-1 Modulo Bluetooth.

Fuente: (http://www.martyncurrey.com/author/admin/)

Existen dos tipos de módulos:

HC-05 este módulo puede desempeñar la función de maestro/esclavo, mientras que el HC-06 solo puede ser configurado como esclavo. Estos módulos son pequeños y fáciles de encontrar en el mercado, contienen un chip con una placa que contiene los pines necesarios para la comunicación serial (DIMakers, 2016, http://diymakers.es/arduino-bluetooth/).

1.5.2 Módulos RF 433Mhz

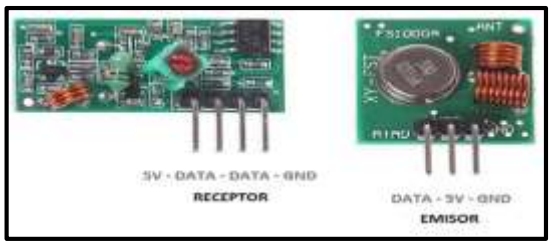


Figura 24-1 Módulos RF 433Mhz.

Fuente: (http://www.josehervas.es/sensorizados/wp-content/uploads/2014/04/rf.jpg)

Son módulos de radiofrecuencia que opera en la banda de frecuencia de 433Mhz, son muy utilizados por su bajo costo en el mercado, consta de un emisor FS1000A y receptor XY-MK-5V, estos módulos permiten establecer una comunicación inalámbrica de tipo simplex, los datos se transmiten por un solo canal y unidireccional es decir el transmisor envía y el receptor recibe pero

no puede enviar ningún dato al receptor, poseen una baja velocidad de transmisión y pueden dar un alcance con (5V) de 40m interior y 100m exterior (naylamp mechatronics, 2016, http://www.naylampmechatronics.com/).

1.5.3 Módulo NRF24L01

El módulo NRF24L01 es un transceptor puede enviar como recibir datos a la vez, opera en la banda de frecuencia de 2,4 GHz con una velocidad de datos en el aire de 1 Mbps y 2 Mbps, de se basan en el chip de Nordic Semiconductor NRF24L01, se basa en el soporte de interfaz SPI para acelerar la comunicación con el micro controlador, son muy económicos de tamaño pequeño se alimentan con 3,3voltios (OpenHardware,2016,http://openhardware.pe/transceptores-nrf24l01-2-4ghz-radio-wireless-how-to/).

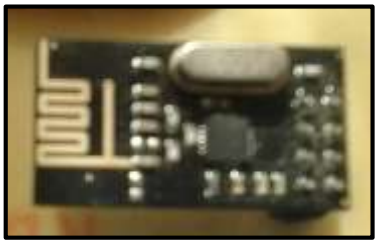


Figura 25-1 Módulos NRF24L01.

Fuente: Apolo K, Coba N, 2016.

Existen dos versiones del módulo NRF24L01 como transceptor de baja potencia con un alcance de 200 metros que tienen una antena integrada y el módulo con antena externa que pueden llegar a un alcance de 1km. (niple,2016, http://www.niplesoft.net/blog/2016/01/13/comunicacion-inalambrica-con-el-transceptor-nrf24l01/)

1.6 Elementos Electrónicos

1.6.1 Batería Lipo 7.4v

Las baterías LiPo es una abreviatura de Litio y polímero es un tipo de batería recargable que se pueden utilizan para los sistemas eléctricos de radiocontrol.

Características (Erle Robotics, 2016, https://erlerobotics.gitbooks.io/erle-robotics-erle-copter/content/es/safety/lipo.html):

- 7.4v
- Son ligeras.
- Tienen gran capacidad, es decir posee una gran cantidad de energía en un tamaño reducido.
- Tasa de descarga alta para alimentar los sistemas eléctricos o electrónicos más exigentes.
- 7.4v a 200mah 30C

Figura 26-1 Batería Lipo 7.4v 2000mah.

Fuente: Apolo K, Coba N, 2016.

1.6.2 Modulo Regulador de voltaje

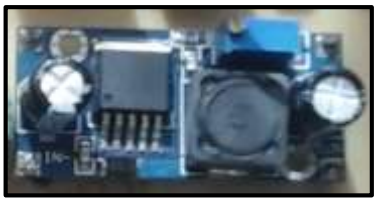


Figura 27-1 Lm2596.

Fuente: Apolo K, Coba N, 2016.

El módulo LM2596 es una fuente de alimentación conmutada basado en el Regulador DC-DC Step Down LM2596 que es un circuito integrado monolítico que permite regular el voltaje a partir de una fuente de alimentación que tenga un voltaje mayor (ELECTRONICLAB,2016, http://electronilab.co/tienda/modulo-lm2596-convertidor-de-voltaje-dc-dc-buck-1-25v-35v/).

1.6.3 Módulo L298N

El módulo L298N es un drive de motores se utiliza para el control de giro y velocidad de motores de corriente directa DC o motores paso a paso. Tiene cuatro terminales para la conexión de dos motores y 4 pines para las señales de control provenientes de un microcontrolador. Así también posee borneras para la alimentación de los motores (Vcc) y para la parte lógica (5V), un regulador de voltaje de 5V para la parte lógica, o dependiendo de las necesidades del usuario se puede añadir una fuente externa de 5V a 12V pero para ello es necesario deshabilitar el regulador de 5V (CtecmiKro, 2016, http://tecmikro.com/modulos-para-arduino-pic-avr/329-l298n-modulo-driver-para-motores.html).

CAPÍTULO II

2 MARCO METODOLÓGICO

El desarrollo del Módulo de Reconocimiento de Comandos de Voz para Niños se realizó básicamente en tres fases:

1. Desarrollo del programa de reconocimiento
2. Implementación del módulo de reconocimiento para aplicarlo en un caso practico
3. Pruebas de funcionamiento

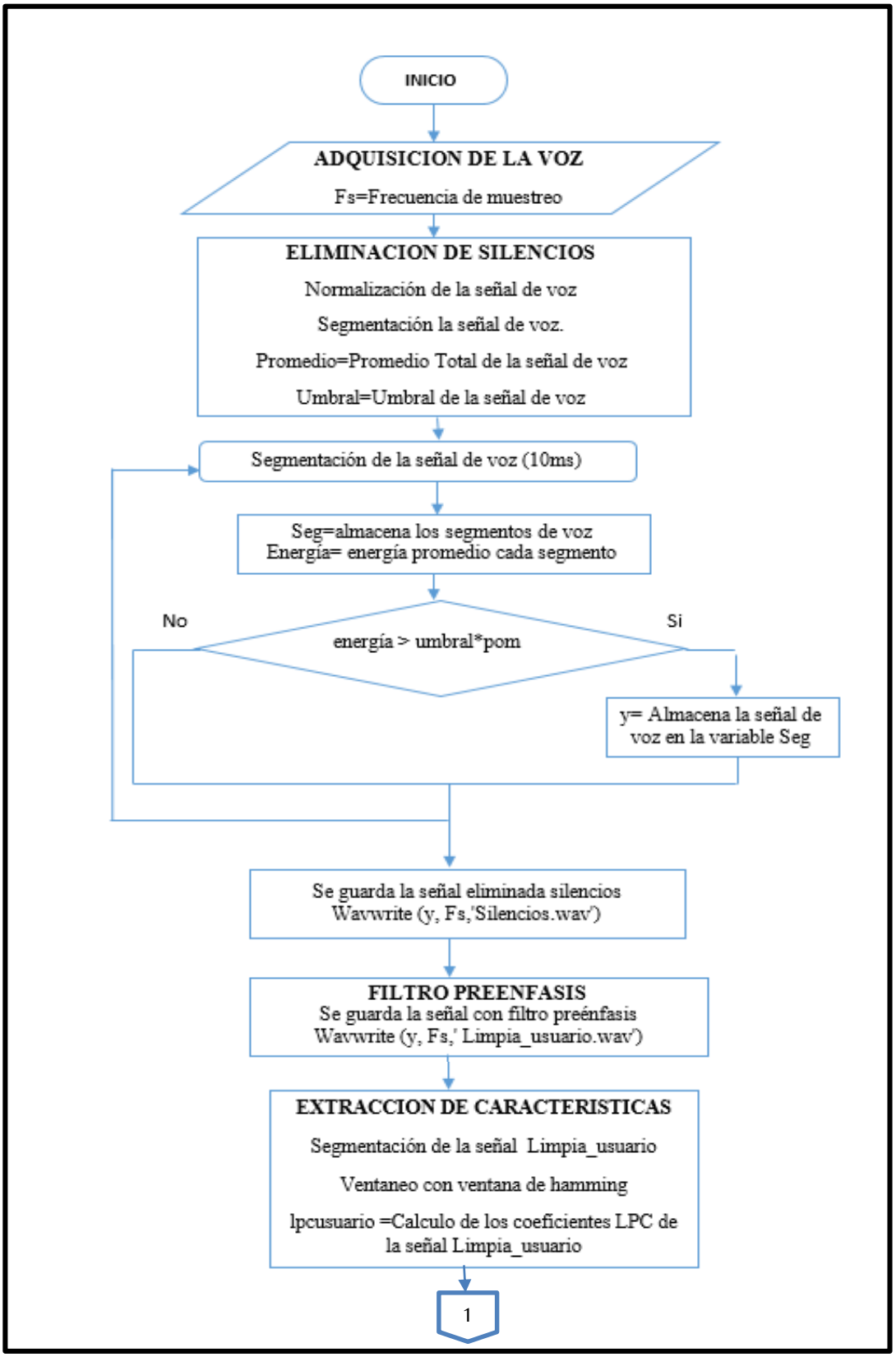
En la primera fase para realizar el procesamiento de la voz se eligió el Software MATLAB debido a que proporciona todas las herramientas necesarias para el análisis y procesamiento de señales digitales, además de ser un software cuyo costo de licencia es conveniente en referencia a otros existentes en el mercado. En el ANEXO B se muestra las características tomadas en consideración para la elección del software que permitió desarrollar el PDS aplicado al reconocimiento de voz.

Una vez elegido el software se procedió con la programación del algoritmo que permitió procesar la señal de voz de manera que se pueda reconocer comandos básicos que luego puedan ser usados en algún caso práctico. Cada uno de los pasos a seguir del algoritmo empleado se describe más adelante pero se sintetizan en el diagrama de bloques representado en la Figura 2-2.

Para hacer más amigable al usuario se crearon dos interfaces gráficas, la primera para grabar cada uno de los comandos a reconocer ,eliminar silencios y pasar por un filtro preénfasis cada señal de manera que se obtenga una señal limpia a la cual se le realizo la extracción de características para lo cual se creó un archivo.m denominado entrenar, proceso que permitió obtener una señal patrón con la que se realizó la comparación para el reconocimiento, la segunda utilizada para el proceso de reconocimiento y análisis por medio de gráficas.

Terminada la primera fase se procedió a la implementación del módulo para su aplicación en un caso práctico el cual posteriormente será descrito con detalles y se concluyó finalmente con la fase de pruebas para determinar la fiabilidad del módulo implementado.

2.1 Algoritmo para el Procesamiento de voz (Reconocimiento de comandos de voz)



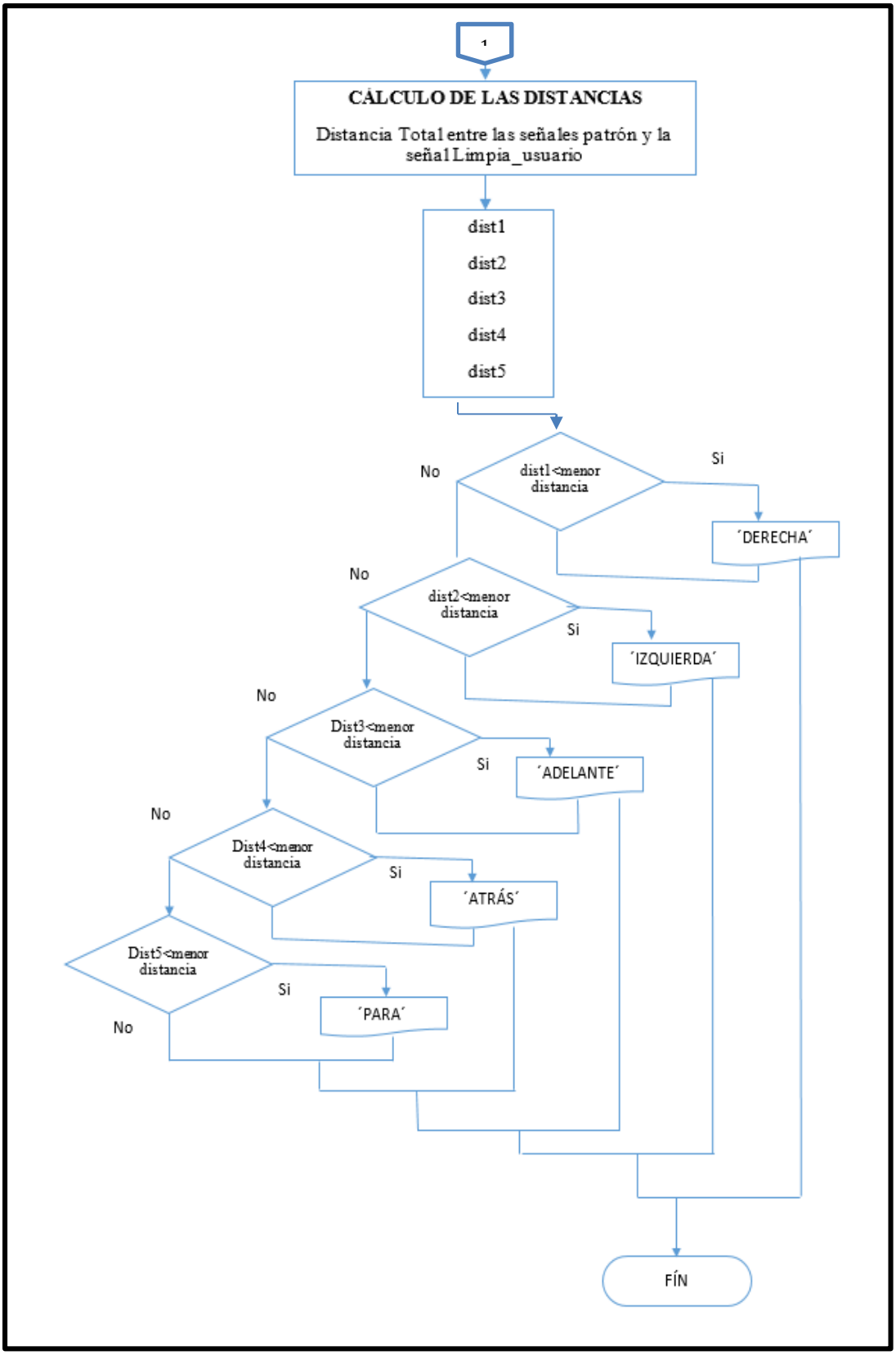


Figura 1-2 Algoritmo procesamiento de voz (proceso de reconocimiento)
Fuente: Apolo K, Coba N, 2016

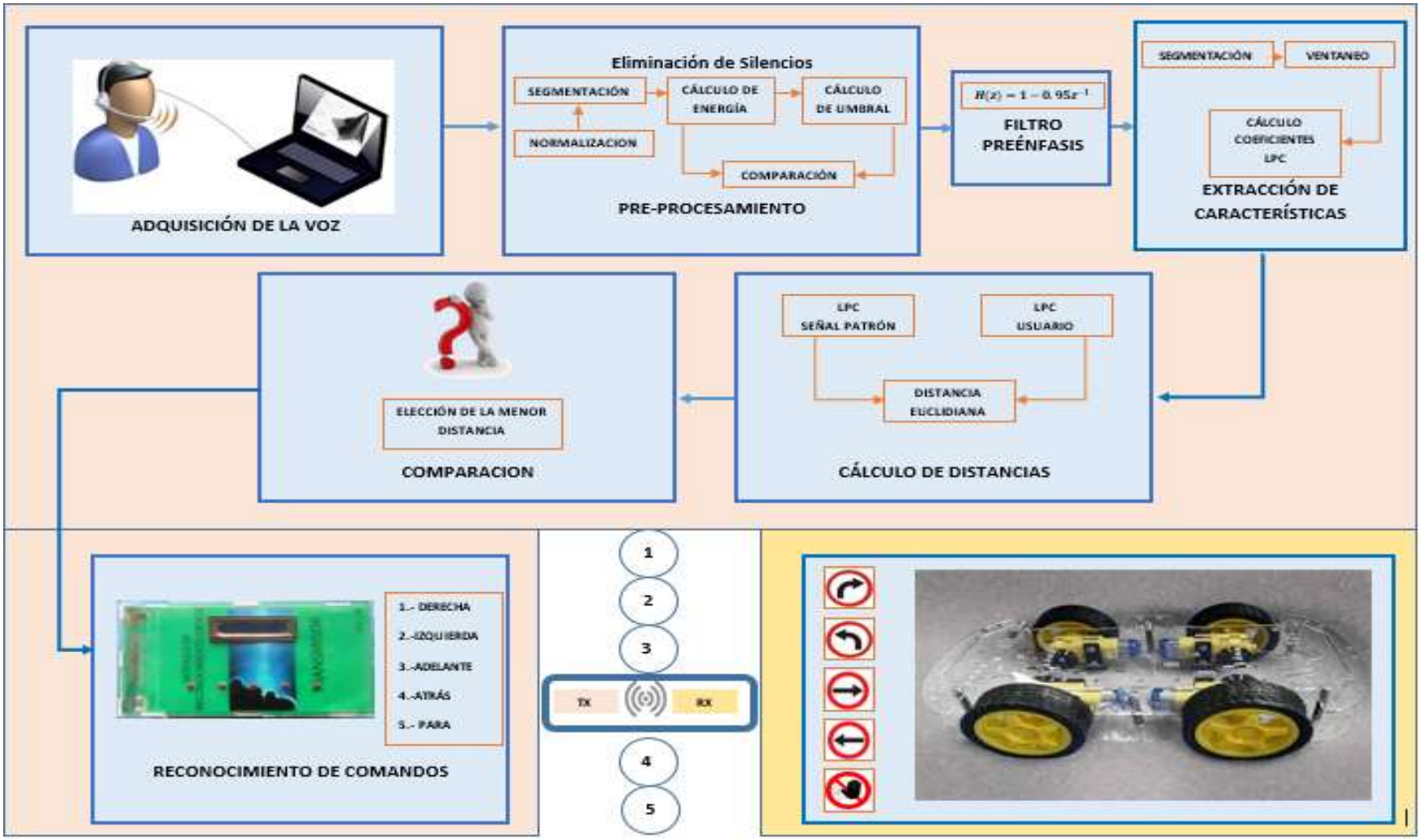


Figura 2-2 Diagrama de bloques de proceso de reconocimiento.
Fuente: Apolo K, Coba N,

2.2 Adquisición de la voz

Para la adquisición de la señal de voz se utilizó el Micrófono Stereo Headset modelo 662862, cuyas características especificadas el en ANEXO D se acoplaron al desarrollo proyecto.

Este micrófono fue elegido por su bajo costo y la forma de grabación ya que al ser unidireccional permite captar sonidos provenientes del locutor en una sola dirección que es lo que se necesita para este caso, además de ofrecer un ambiente cómodo al usuario por la forma de diadema que adopta.

Figura 3-2 Stereo Headset 662862.

Fuente: http://www.omegatechnology.com.

2.2.1 Base de Datos

Utilizando la interfaz gráfica del software MATLAB R2012a se creó una base de datos con 25 grabaciones de los comandos a reconocer izquierda, derecha, adelante, atrás y para, correspondientes a cada usuario. El objetivo de realizar esta base de datos es obtener una señal patrón que será utilizada posteriormente en el proceso de reconocimiento. El código correspondiente a esta etapa se puede observar en la Figura 4-2.

```
Fs=22050;                                %Frecuencia de muestreo
s=wavrecord(2*Fs,Fs,1);                  %Función de grabacion
wavwrite(s,Fs,16,'grabacion1.wav')       %Función para guardar la grabacion
sound (s,Fs);                            %Reproducir la grabacion
```

Figura 4-2 Código para adquirir la voz.

Fuente: Apolo K, Coba N, 2016.

En esta etapa es importante considerar que Matlab maneja estándares de frecuencia de muestreo de 8000, 11025, 2250 y 44100 muestras por segundo, en este caso se tomó una frecuencia de muestreo de 22050. En la función de grabar el tiempo de cada grabación es de 2s, tiempo necesario para la emisión de cada uno de los comandos, y se eligió un solo canal debido a que solo se va a grabar voz. Se puede especificar además el número de bits por segundo que se desean grabar en este caso 16 bits por segundo que al no especificar Matlab utiliza por defecto.

Para guardar la grabación se debe especificar los mismos parámetros que para grabar además del nombre con el que se guardara la grabación. La función **sound** nos permite escuchar el sonido grabado para de esta manera guardar o descartar la grabación.

2.3 Pre procesamiento

La etapa de pre procesamiento se enfocó principalmente en la eliminación de silencios de la señal de voz para de esta manera obtener una señal limpia a la cual se le realizó la extracción de características. Los procesos a seguir se detallan a continuación.

2.3.1 Normalización y determinación del umbral

La normalización permite aplicar una ganancia de 0dB a la señal de voz adecuando de esta manera las amplitudes de la señal para que tomen valores desde -1 a 1 con el objetivo de evitar la distorsión de la señal.

El umbral está representado por el cálculo de energía en determinada señal, dependiendo del tono de voz de cada persona esta energía aumentara o disminuirá, en el caso de los niños este umbral es relativamente bajo en referencia a un adulto. Para determinar el umbral a utilizar en la etapa de eliminación de silencios, se tomó un valor que englobe el cálculo de energía de 10 grabaciones de cada usuario, debido a que los umbrales de los dos usuarios máximos son 0.0183 y 0.0238 respectivamente y se necesita eliminar ruido que puede confundirse con sonidos sordos se eligió el valor de 0.05. Los valores de los umbrales de cada usuario se muestran en la Tabla 1-2.

Tabla 1-2 Cálculo del umbral

	USUARIO 1					**USUARIO 2**				
Derecha	0.0168	0.0050	0.0028	0.0106	0.0123	0.0169	0.0183	0.0171	0.0069	0.0092
Izquierda	0.0077	0.0111	0.0099	0.0072	0.0088	0.0085	0.0171	0.0188	0.0163	0.0105
Adelante	0.0075	0.0093	0.0092	0.0057	0.0045	0.0199	0.0137	0.0238	0.0184	0.0231
Atrás	0.0167	0.0123	0.0183	0.0151	0.0158	0.0193	0.0145	0.0117	0.0116	0.0117
Alto	0.0089	0.077	0.0130	0.0143	0.0139	0.0079	0.0127	0.0083	0.0081	0.01
Valor	**0.05**									

Fuente: Apolo K, Coba N, 2016.

```
lon=length(s);                          %Longitud de la señal
d=max(abs(s));                          %Encontrar la muestra con mayor valor
s=s/d;                                  %Normalizar la señal
prom=sum(s.*s)/lon;                     %Promedio de la señal total
umbral=0.05;                            %Valor de umbral calculado
```

Figura 5-2 Código para realizar la normalización y cálculo energía.

Fuente: Apolo K, Coba N, 2016.

2.3.2 Segmentación

Para la segmentación se eligió intervalos de 10ms por lo que se obtuvieron aproximadamente 400 ventanas de acuerdo a la frecuencia de muestro y tiempo de grabación elegidos.

$$\#Seg = (frecuecia\ de\ muestreo \times tiempo\ de\ grabación) \times t_{intervalo} \qquad (2\text{-}1)$$

$$\#Seg = (22050 \times 2) \times 10ms \cong 440$$

```
for i = 1:400:lon-400                   %Segmentación de la señal cada 10ms
seg = s(i:i+399);                       %Almacenar segmentos
end
```

Figura 6-2 Código para segmentar la voz.

Fuente: Apolo K, Coba N, 2016.

2.3.3 Calculo de energía

Se calcula la energía de cada segmento para determinar los segmentos sonoros de la señal.

```
energia = sum(seg.*seg)/400;            %Promedio de energia cada segmento
```

Figura 7-2 Código para cálculo de energía de cada segmento.

Fuente: Apolo K, Coba N, 2016.

2.3.4 Eliminación de silencios

Para poder eliminar de la señal de voz los segmentos de silencios se hace una comparación entre el valor energético de cada segmento y el valor energético completado por el umbral de la señal total, si este primer valor mencionado anteriormente es mayor entonces se almacena el segmento

de la señal en el caso de ser el valor energético del segmento menor no se almacena por lo cual dicho segmento se considera como un periodo de silencio.

```
if( energia> umbral*prom)                %Condicion para determinarla seañal y descartar el silencio
y=[y;seg(1:end)];                        %Almacena la señal sin silencios
end
```

Figura 8-2 Código para eliminar silencios.

Fuente: Apolo K, Coba N, 2016.

2.4 Filtrado preénfasis

Se utilizó filtro preénfasis (filtro pasa alto de primer orden) con el objetivo de suavizar el espectro y acentuar frecuencias altas correspondientes a las consonantes debido a que al momento de eliminar silencios se puede perder información.

En lo que al reconocimiento de voz se refiere para el filtrado preénfasis representado en la Ecuación 1-25 se utiliza una constante de 0.95 debido a que este valor permite obtener una amplificación de 20dB.

```
%FILTRO PREENFASIS
 b=[1 -0.95];
 y1=filter(b,1,y);
 wavwrite(y1,Fs,'Limpia_usuario.wav')
```

Figura 9-2 Código para filtrado preénfasis

Fuente: Apolo K, Coba N, 2016.

En la figura anterior se puede observar el valor de los coeficientes del numerador (1) y del denominador (0.95), para filtrar la señal sin silencios se ha utilizado la función filter, donde:

b: Coeficientes del numerador del filtro

a: Coeficientes del denominador del filtro

y: Señal a filtrar

2.5 Extracción de características

Eliminados los silencios y filtrada la señal de cada una de las grabaciones se creó un archivo.m denominado “entrenar” cuya función fue el cálculo de los coeficientes LPC.

De cada una de las 25 grabaciones de cada comando se calculó los coeficientes LPC y se realizó el promedio entre estos para de esta manera obtener una señal patrón de cada comando que sirvieron de referencia para el reconocimiento.

El número de coeficientes LPC a calcular fue de 23, calculados utilizando la Ecuación 1-57:

$$p = \frac{22050}{1000} \cong 23$$

Para el cálculo realizado anteriormente se debe considerar valores enteros ya que la función que realiza el cálculo de coeficientes LPC en MATLAB solo admite enteros positivos. El proceso realizado para el comando Derecha se muestra en la Figura 10-2.

```
%ENTRENAMIENTO DE LOS COMATDOS A RECONOCER
%DERECHA

voz_De1=wavread ('D1');
ent_De1=lpc (voz_De1,23);

voz_De2=wavread ('D2');
ent_De2=lpc (voz_De2,23);

voz_De3=wavread ('D3');
ent_De3=lpc (voz_De3,23);

voz_De4=wavread ('D4');
ent_De4=lpc (voz_De4,23);

voz_De5=wavread ('D5');
ent_De5=lpc (voz_De5,23);

voz_De6=wavread ('D6');
ent_De6=lpc (voz_De6,23);

voz_De7=wavread ('D7');
ent_De7=lpc (voz_De7,23);

voz_De8=wavread ('D8');
ent_De8=lpc (voz_De8,23);

voz_De9=wavread ('D9');
ent_De9=lpc (voz_De9,23);

voz_De10=wavread ('D10');
ent_De10=lpc (voz_De10,23);
          :
voz_De25=wavread ('D25');
ent_De25=lpc (voz_De25,23);

patronD=(ent_De1+ent_De2+ent_De3+ent_De4+ent_De5+ent_De6+ent_De7+ent_De8+ent_De9+ent_De10+
ent_De11+ent_De12+ent_De13+ent_De14+ent_De15+ent_De16+ent_De17+ent_De18+ent_De19+ent_De20+
ent_De21+ent_De22+ent_De23+ent_De24+ent_De25)/25;
```

Figura 10-2 Código para calcular coeficientes lpc de la base de datos.

Fuente: Apolo K, Coba N, 2016.

Para el cálculo de los coeficientes LPC se utilizó la función:

$$lpc(x, n)$$

donde:

x: Representa la señal de voz de donde se extraerán los coeficientes.
n: Número de coeficientes lpc.

La función descrita anteriormente abarca los siguientes procesos realizados internamente por el software:

- Ventaneo (Ventana Hamming cada 30ms)
- Filtro de Predicción Lineal
- Calculo de coeficientes utilizando el algoritmo de Levinson Durbin

De cada señal patrón como de la señal ingresada por el usuario se obtiene un vector con 23 coeficientes lpc.

El código correspondiente a la interfaz de reconocimiento y análisis para el cálculo de lpc de la señal de voz ingresada se muestra en la Figura 11-2.

```
entrenar;

[g_usuario,Fs,bits]=wavread('Limpia_usuario.wav');

lpcusuario=lpc(g_usuario,19);

save lpcLimpia_usuario.mat lpcusuario;
```

Figura 11-2 Código para calcular lpc de la señal ingresada por un usuario.
Fuente: Apolo K, Coba N, 2016.

Nota: La señal envolvente obtenida a partir de los coeficientes LPC será denominada como señal predicha.

2.5.1 Análisis LPC

Se realizó un Análisis de LPC con la ayuda de Matlab para graficar cada una de las señales que se utiliza en la Codificación Predictiva Lineal para la determinación de los coeficientes LPC a través del Error Predicción, se calculó los formantes de la señal de voz a través del Método de las Raíces y Método Peak-Picking (envolvente H(z)), así también se muestran graficas del modelo AR del espectro de la señal de voz de usuario con su respectiva envolvente LPC.

2.5.1.1 Error de Predicción:

Para el Error de Predicción se procedió a enventanar la señal de voz a través de una Ventana de Hamming.

```
%######ERROR DE PREDICCIÓN ##########
%Seña enventanada
load lpcLimpia_usuario.mat% cargamos la varible almacenada lpcusuario
%que contiene los coeficientes lpc de la señal
N=240;%numero de puntos para la ventana de hamming
v=hamming(240);
[s,fs,bits]=wavread('Limpia_usuario.wav');
t=1:240;% variable para el numero de muestras
sw=v.*s(t); %mutlipicacion de la señal de voz por la ventana de hamming
coeflpc=lpc(sw,23);
sw=sw/max(abs(sw));%normalizamos la señal
est_sw = filter([0 -coeflpc(2:end)],1,sw);%se aplica el filtro de predicción H(z)=A(z)
e = sw - est_sw;% se calcula el Error de predicción
```

Figura 12-2 Código para el Error de Predicción

Fuente: Apolo K, Coba N, 2016.

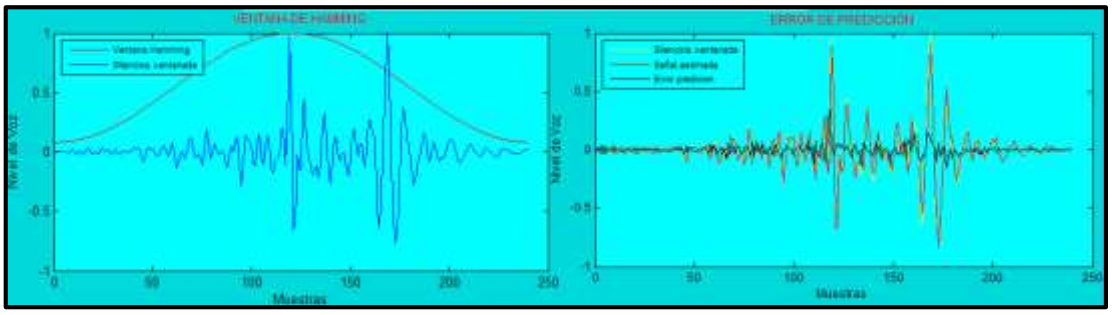


Figura 13-2 Error de Predicción

Fuente: Apolo K, Coba N, 2016.

2.5.1.2 Modelo AR o de todos polos

$$\mathrm{H(z)} = \frac{S(z)}{U(z)} = \frac{G}{1 + \sum_{k=1}^{p} \alpha_k \, z^{-k}}$$

```
%######MODELO AR ##########

[g_usuario,Fs]=wavread('Limpia_usuario.wav');%señal de voz eliminada los Silencios
num1=19;%numero de coeficientes Lpc
op1 = length(g_usuario);%longitud de la señal
Rsw1 = xcorr(g_usuario);% Vector de Autocorrelacion
R1 = Rsw1(op1:op1); % Obtención R(0)
G1 = sqrt(sum( lpcusuario.*R1')); % Obtención de la Ganancia
envolvente1 = abs(G1./fft( lpcusuario,op1));% Obtención de la envolvente H(z)
SW1 = abs(fft(g_usuario,op1)); % Transformada de Fourier de la señal original

%Graficas
axes(handles.axes3);
plot(20*log10(SW1(1:(op1/2))))%Graffica Transformada de Fourier de la señal original
hold on;
plot(20*log10(envolvente1(1:(op1/2))),'y')%Graffica  envolvente señal
title('MODELO AR ', 'FontSize', 9,'color','red')
xlabel('Frecuencia');
ylabel('Ganancia');
hleg1 = legend('señal.usuario.silencios','envolvente',2)
hold off;
```

Figura 14-2 Código para Modelo AR

Fuente: Apolo K, Coba N, 2016.

2.5.1.3 Formantes de la Señal

Para el cálculo de los Formantes de la señal de voz se lo realizó a través de los siguientes métodos.

2.5.1.3.1 Método Peak-Picking

Este método permite encontrar los formantes de la señal a través de la envolvente empleando la función de Transferencia, para ello se aplica un filtro digital para representar este sistema.

$$\mathrm{H(z)} = \frac{1}{1 + \sum_{k=1}^{p} \alpha_k \, z^{-k}}$$

Matlab proporciona el comando [h,w] = freqz (b,a,n,fs) para representar el filtro digital, donde:

b: numerador

a: denominador

n: número de puntos

fs: frecuencia de muestreo

```
%###################METODO PEAK-PICKING  ########

[x,fs]=wavread('Limpia_usuario');%señal de voz eliminada los silencios

a=lpc(x,19);%variable que depermina los coeficientes Lpc donde
%(señal silencios,#de coeficientes)

[h,f]=freqz(1,a,240,fs);%respuesta en frecuencia de un filtro digital donde:
%freqz(numerador,denominador,numero de puntos ,Frecuencia de muestreo)

axes(handles.axes1);
plot(f,20*log10(abs(h)+eps));%Grafica del Filtro Digital
legend('Envolvente H(Z)');
title('Envolvente_ H(Z)', 'FontSize', 9,'color','red')
xlabel('Frequency (Hz)');
ylabel('Gain (dB)');
```

Figura 15-2 Código Peak Picking

Fuente: Apolo K, Coba N, 2016.

2.5.1.3.2 Método de las Raíces

```
%#############FORMANTE METODO DE LAS RAICES

roots_a=roots(a); % la variable roots_a calcula el valor de las raices
formants_a=angle(roots_a)*fs/(2*pi);%Determina las frecuencia de los Formantes Fi=θ_i/(2πT_s )
a_sorted = sort(abs(formants_a));% determina el valor absoluto ,ordena el vector

%imprimir cada 2 posicion debido a que cada par de polos complejos conjugados corresponde a un formante
set(handles.text2,'String',a_sorted(2))% formante F1
set(handles.text3,'String',a_sorted(4))% formante F2
set(handles.text4,'String',a_sorted(6))% formante F3
```

Figura 16-2 Código Método de las Raíces

Fuente: Apolo K, Coba N, 2016.

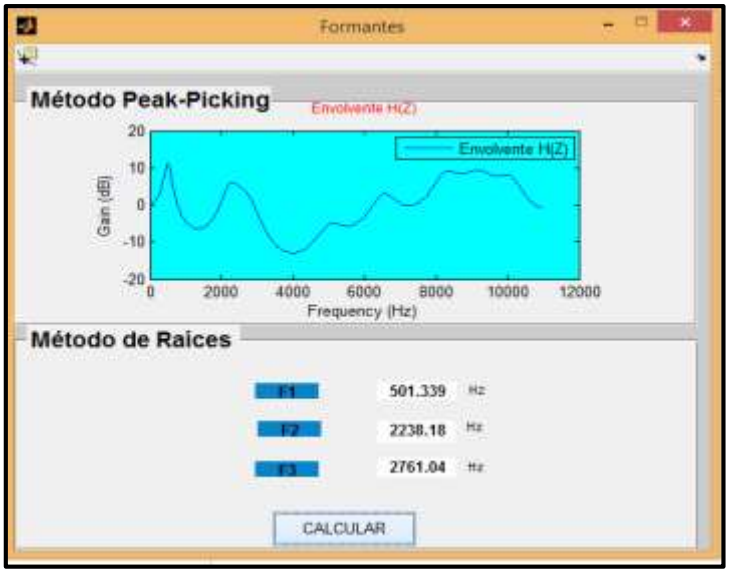


Figura 17-2 Interfaz Gráfica Formantes

Fuente: Apolo K, Coba N, 2016.

2.6 Cálculo de Distancias

Este proceso se realiza en la interfaz gráfica correspondiente al reconocimiento de voz y análisis, se calcula la distancia entre los vectores que almacenan los coeficientes lpc de cada señal de voz patrón y la señal de voz ingresada por el usuario, obteniendo un vector de cada comando a reconocer con las distancias entre cada uno de los coeficientes.

Finalmente se calcula la distancia total que va a ser factor de comparación.

```
%----CALCULO DE LAS DISTANCIAS

dis1=((PatronD-lpcusuario).^2); %Distancia entre patron Derecha y lpcusuario
dist1=sum(dis1);                %Sumatoria para calculo de distancia total

dis2=((PatronIZ-lpcusuario).^2);% Distancia entre patron Izquierda y lpcusuario
dist2=sum(dis2);                %Sumatoria para calculo de distancia total

dis3=((PatronAD-lpcusuario).^2);% Distancia entre patron Adelante y lpcusuario
dist3=sum(dis3);                %Sumatoria para calculo de distancia total

dis4=((PatronAT-lpcusuario).^2);%Distancia entre patron Atras y lpcusuario
dist4=sum(dis4);                 %Sumatoria para calculo de distancia total

dis5=((PatronP-lpcusuario).^2);% Distancia entre patron Para y lpcusuario
dist5=sum(dis5);                %Sumatoria para calculo de distancia total
```

Figura 18-2 Código para calcular distancias.

Fuente: Apolo K, Coba N, 2016.

2.7 Comparación de Distancias

Se elige como comando reconocido la menor distancia total entre la señal patrón y la señal de voz lpc ingresada, ya que en las señales que se aproximan esta distancia es mínima.

De esta manera se realiza el reconocimiento de cada uno de los comandos descritos anteriormente.

```
%..COMPARACIÓN DE DISTANCIAS TOTALES

if      (dist1<dist2)&(dist1<dist3)&(dist1<dist4)&(dist1<dist5)

        set(handles.text1,'String','DERECHA' )

        elseif    (dist2<dist1)&(dist2<dist3)&(dist2<dist4)&(dist2<dist5)

        set(handles.text1,'String','IZQUIERDA')

        elseif (dist3<dist1)&(dist3<dist2)&(dist3<dist4)&(dist3<dist5)

        set(handles.text1,'String','Adelante')

        elseif   (dist4<dist1)&(dist4<dist2)&(dist4<dist3)&(dist4<dist5)

        set(handles.text1,'String','ATRAS')

        else  (dist5<dist1)&&(dist5<vti2)&(dist5<dist3)&(dist5<dist4)

        set(handles.text1,'String','PARA')

end
```

Figura 19-2 Código para elegir la menor distancia total.

Fuente: Apolo K, Coba N, 2016.

2.8 Caso Práctico

El Caso Práctico que se utilizó para la demostración del programa desarrollado en el software Matlab, consistió en el control de un Carro Robot 4wd, que fue manipulado de acuerdo a las órdenes de voz ejecutadas en Matlab. El comando o palabra reconocida envía datos a través del puerto serial a Arduino para el control de dichos movimientos, para ello se necesitó tener una comunicación entre Matlab y Arduino. Se escogieron 5 comandos de voz a reconocer que se detallan a continuación:

- Derecha
- Izquierda
- Adelante
- Atrás
- Para

2.8.1 Selección del Módulos de Comunicación Inalámbrica

Se escogió el módulo NRF24L01 debido a que tiene las mejores características ya que al ser un módulo transceptor puede ser configurado como emisor o receptor, es económico fácil de encontrar en el mercado, posee un alcance aceptable con una velocidad de 1 Mbps y 2 Mbps.

2.8.2 Selección de la Tarjeta de Adquisición

La selección de la tarjeta de adquisición se realizó en función del número de entradas y salidas tanto analógicas como digitales a utilizarse en la implantación del módulo de reconocimiento de voz y en el movimiento del Carro Robot 4wd de acuerdo a las órdenes de voz específicas.

Se escogió la tarjeta Arduino Uno que se encuentra en el mercado con un bajo costo, además es compatible con el módulo de comunicación inalámbrica NRF24L01 y posee las entradas y salidas tanto analógicas como digitales requeridas.

Tabla 2-2 Descripción de Entradas y Salida.

Tx	**Señal Digital**	**Señal Analógica**	**Rx**	**Señal Digital**
Datos binarios que envía Matlab	5			
LCD con bus I2C		1	Salidas para L298m	4
Comunicación Inalámbrica NRF24L01	8		Comunicación Inalámbrica NRF24L01	8
TOTAL	12	1	**TOTAL**	9

Fuente: Apolo K, Coba N, 2016.

2.8.3 Enlace Matlab Arduino

La tarjeta Arduino es una placa que es compatible con Matlab la cual permite realizar la adquisición de datos. Matlab ofrece un paquete ArduinoIO que permite obtener una conexión entre Guide de Matlab y Arduino, este paquete puede ser usado normalmente en versiones de Matlab R2013b o versiones anteriores pero no antes de R2011a. (MathWorks, 2016, http://www.mathworks.com/matlabcentral/fileexchange/32374-legacy-matlab-and-simulink-support-for-arduino)

Los pasos para establecer la instalación para la conexión entre Matlab y Arduino se especifican en el ANEXO E.

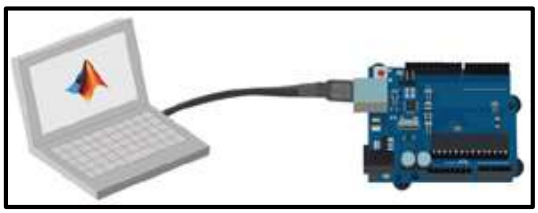


Figura 20-2 Enlace Matlab Arduino.

Fuente: (https://es.mathworks.com/cmsimages/107904_wm_Arduino-MATLAB.jpg

2.9 Diseño e Implementación del Módulo de Reconocimiento de Voz

El módulo de Reconocimiento de Voz consta de dos etapas de comunicación inalámbrica un Transmisor y un Receptor.

2.9.1 Módulo Transmisor

En la Figura 21-2 se muestra el diagrama correspondiente al Módulo Transmisor de Reconocimiento de Voz.

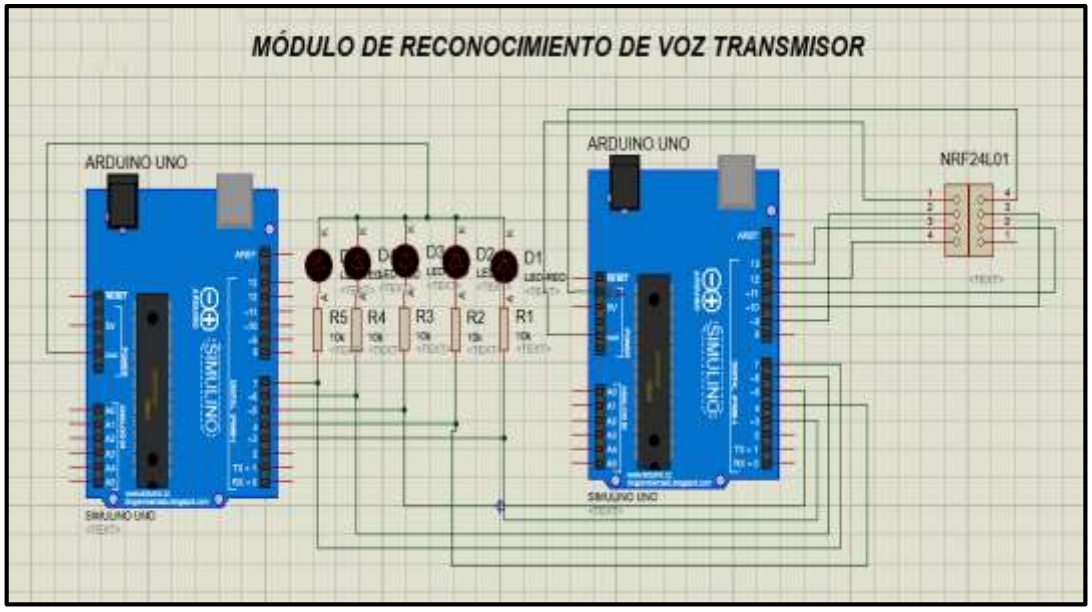


Figura 21-2 Módulo de Reconocimiento de Voz Tx.

Fuente: Apolo K, Coba N, 2016.

El Transmisor está constituido de dos placas Arduino Uno, cinco leds indicadores de comando, un LCD y un bus I2C. El primer Arduino efectúa la comunicación o adquisición de datos entre Matlab y Arduino a través del paquete Arduino IO.

Matlab envía datos digitales a los pines de Arduino, tomando en consideración el comando reconocido. Este primer Arduino recibe los datos digitales, mostrando sus salidas a través de leds indicadores que varían de acuerdo al comando.

En la Figura 22-2 y Figura 23-2 se muestra el código en Matlab para él envió de datos hacia Arduino.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%COMUNICACION MATLAB ARDUINO%%%%%%%%%%%%%%%%%%%%%%%%%%
clear a;
global a;

%%delete(instrfindall);

a=arduino('COM2');%se asigna el numero de puerto que esta conectado Arduino Uno

%%instrfind

 a.pinMode(3,'output');% se asigna el pin 3 de Arduino como salida
 a.pinMode(4,'output');%se asigna el pin  4 de Arduino como salida
 a.pinMode(5,'output');%se asigna el pin  5 de Arduino como salida
 a.pinMode(6,'output');%se asigna el pin  6 de Arduino como salida
 a.pinMode(7,'output');%se asigna el pin  7 de Arduino como salida

 handles.a=a;


handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Reconocimiento_Voz_Ninos wait for user response (see UIRESUME)
% uiwait(handles.figure1);
```

Figura 22-2 Comandos en Matlab enviar datos a Arduino

Fuente: Apolo K, Coba N, 2016.

Se debe asignar el número de puerto que ha sido asignado para el Arduino Uno y así también los pines de salida , este caso se utilizaron 5 pines para indicar a través de bits cada uno de los comandos de voz reconocidos en Matlab.

```
%..COMPARACIÓN DE DISTANCIAS TOTALES

if      (dist1<dist2)&(dist1<dist3)&(dist1<dist4)&(dist1<dist5)

         Distan=dist1;

         set(handles.text1,'String','DERECHA' )%se imprime el comando de voz DERECHA

         %Envio de datos a la tarjeta Arduino Uno
         handles.a.digitalWrite(3,1);% se envia el 1 al pin 3 de Arduino
         handles.a.digitalWrite(4,0);% se envia el 0 al pin 4 de Arduino
         handles.a.digitalWrite(5,0);% se envia el 0 al pin 5 de Arduino
         handles.a.digitalWrite(6,0);%se envia el  0 al pin 6 de Arduino
         handles.a.digitalWrite(7,0);% se envia el 0 al pin 7 de Arduino

elseif    (dist2<dist1)&(dist2<dist3)&(dist2<dist4)&(dist2<dist5)

           Distan=dist2;

    set(handles.text1,'String','IZQUIERDA')%se imprime el comando de voz IZQUIERDA

         handles.a.digitalWrite(3,0);% se envia el 0 al pin 3 de Arduino
         handles.a.digitalWrite(4,1);% se envia el 1 al pin 4 de Arduino
         handles.a.digitalWrite(5,0);% se envia el 0 al pin 5 de Arduino
         handles.a.digitalWrite(6,0);% se envia el 0 al pin 6 de Arduino
         handles.a.digitalWrite(7,0);% se envia el 0 al pin 7 de Arduino
```

```
[voz,Fs]=wavread('Grabacion Derecha.wav');      %Leer audio
voz=s;                                          %Asignar el audio a una variable
lon=length(s);                                  %Longitud de la señal
d=max(abs(s));                                  %Encontrar la muestra con mayor valor
s=s/d;                                          %Normalizar la señal
prom=sum(s.*s)/lon;                             %Promedio de la señal total
umbral=0.09;                                    %Valor de umbral calculado

        elseif (dist3<dist1)&(dist3<dist2)&(dist3<dist4)&(dist3<dist5)

                Distan=dist3;

            set(handles.text1,'String','ADELANTE')%se imprime el comando de voz ADELANTE


                handles.a.digitalWrite(3,0);% se envia el 0 al pin 3 de Arduino
                handles.a.digitalWrite(4,0);% se envia el 0 al pin 4 de Arduino
                handles.a.digitalWrite(5,1);% se envia el 1 al pin 5 de Arduino
                handles.a.digitalWrite(6,0);% se envia el 0 al pin 6 de Arduino
                handles.a.digitalWrite(7,0);% se envia el 0 al pin 7 de Arduino


        elseif   (dist4<dist1)&(dist4<dist2)&(dist4<dist3)&(dist4<dist5)

                 Distan=dist4;

             set(handles.text1,'String','ATRAS')%se imprime el comando de voz ATRAS

                handles.a.digitalWrite(3,0);% se envia el 0 al pin 3 de Arduino
                handles.a.digitalWrite(4,0);% se envia el 0 al pin 4 de Arduino
                handles.a.digitalWrite(5,0);% se envia el 0 al pin 5 de Arduino
                handles.a.digitalWrite(6,1);% se envia el 1 al pin 6 de Arduino
                handles.a.digitalWrite(7,0);% se envia el 0 al pin 7 de Arduino

else   (dist5<dist1)&&(dist5<dist2)&(dist5<dist3)&(dist5<dist4)

        Distan=dist5;

    set(handles.text1,'String','PARA')%se imprime el comando de voz PARA

          handles.a.digitalWrite(3,0);% se envia el 0 al pin 3 de Arduino
          handles.a.digitalWrite(4,0);% se envia el 0 al pin 4 de Arduino
          handles.a.digitalWrite(5,0);% se envia el 0 al pin 5 de Arduino
          handles.a.digitalWrite(6,0);% se envia el 0 al pin 6 de Arduino
          handles.a.digitalWrite(7,1);% se envia el 1 al pin 7 de Arduino

end
```

Figura 23-2 Datos enviados desde Matlab a Arduino

Fuente: Apolo K, Coba N, 2016.

En la parte de las comparaciones de las distancias de los comandos de voz a reconocer, se envía los datos (bits) a la tarjeta Arduino Uno del comando de voz que ha sido reconocido.

Tabla 3-2 Salida del Primer Arduino.

NUMERO DE COMANDOS DE VOZ	COMANDOS A RECONOCER	PINES DEL PRIMER ARDUINO				
		Pin 3	Pin4	Pin 5	Pin6	Pin 7
Comando 1	Derecha	1	0	0	0	0
Comando 2	Izquierda	0	1	0	0	0
Comando 3	Adelante	0	0	1	0	0
Comando 4	Atrás	0	0	0	1	0
Comando 5	Para	0	0	0	0	1

Fuente: Apolo K, Coba N, 2016.

El segundo Arduino se encarga de leer los datos del primer Arduino considerando los mismos pines de conexión que se utilizaron en el primero, estos datos fueron comparados para determinar cada uno de los comandos a reconocer y los mismos se visualizaron a través de un LCD 16x2 con un bus I2C. Se utilizó el bus I2C debido a que utiliza solo 2 pines analógicos del Arduino Uno (A4 y A5) los cuales son conectados al bus I2C correspondientes a los pines (SDA y SCL).

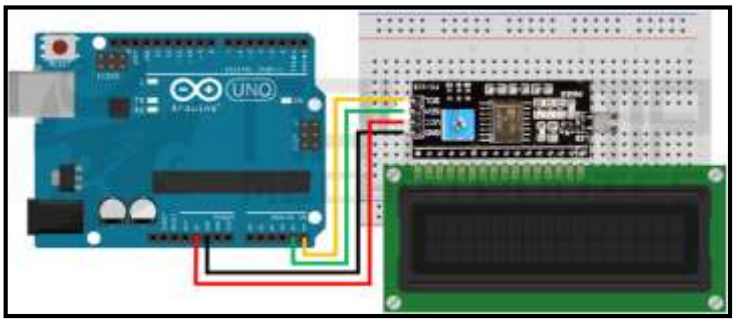


Figura 24-2 Conexión LCD y bus I2C

Fuente: http://www.naylampmechatronics.com/modules//smartblog/images/35-single-default.jpg

Para enviar los datos al Receptor se utilizó el módulo de radio frecuencia NRF24L01 el cual permite establecer comunicación inalámbrica. (ver ANEXO F)

Tabla 4-2 E/S del segundo Arduino.

NUMERO DE COMANDOS DE VOZ	LECTURA DE DATOS DEL SEGUNDO ARDUINO					LCD	ENVÍO DE DATOS AL RECEPTOR
	Pin3	Pin4	Pin5	Pin6	Pin7		
Comando 1	1	0	0	0	0	DERECHA	Dato 0
Comando 2	0	1	0	0	0	IZQUIERDA	Dato 1
Comando 3	0	0	1	0	0	ADELANTE	Dato 2
Comando 4	0	0	0	1	0	ATRÁS	Dato 3
Comando 5	0	0	0	0	1	PARA	Dato 4

Fuente: Apolo K, Coba N, 2016

El código correspondiente a la programación que se realizó en la tarjeta Arduino Uno se muestra en el ANEXO G.

2.9.2 Módulo Receptor

El Modulo Receptor permite receptar los datos que fueron enviados a través del transmisor, está constituido de una tarjeta o placa Arduino, un módulo de radio frecuencia NRF24L01, un Regulador De Voltaje Lm2596, un driver para motor DC L298 y una batería Lipo de 7.4v a 2000mAh.

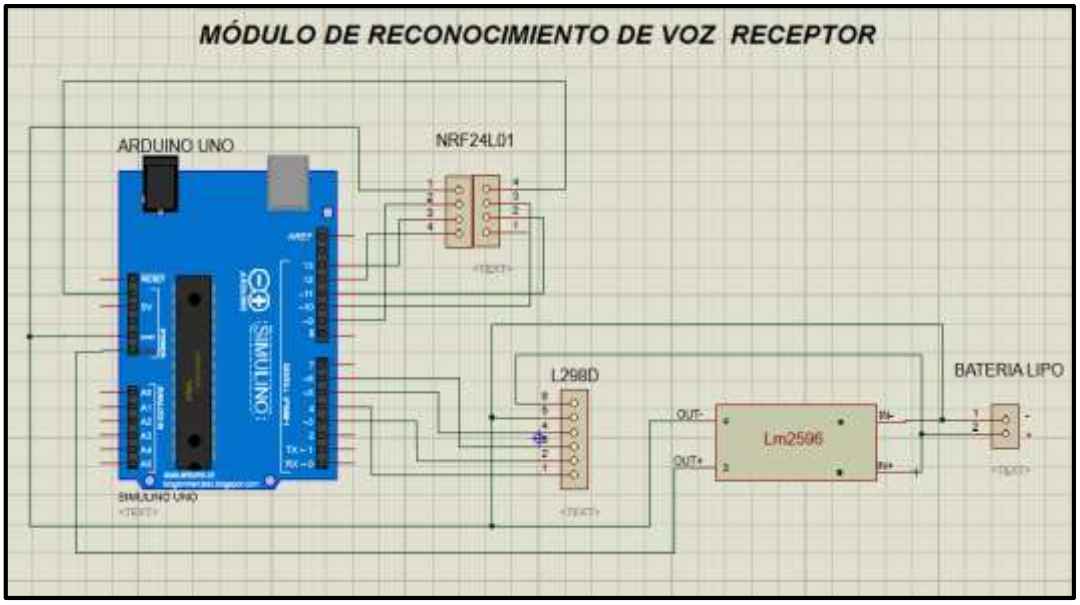


Figura 25-2 Módulo de Reconocimiento de Voz Rx.

Fuente: Apolo K, Coba N, 2016.

Los datos recibidos por este Arduino son analizados para identificar el dato correspondiente, para dar como resultado las diferentes salidas de este Arduino, las cuales sirvieron de entradas para el Driver de motores DC L298N.

Tabla 5-2 E/S Arduino Rx.

DATOS RECIBIDOS (ARDUINO RX)	SALIDAS DE ARDUINO RX			
	Pin 3	Pin 4	Pin 5	Pin 6
Dato 0	0	1	1	0
Dato 1	1	0	0	1
Dato 2	1	0	1	0
Dato 3	0	1	0	1
Dato 4	0	0	0	0

Fuente: Apolo K, Coba N, 2016.

Tabla 6-2 Entradas Driver de motores DC.

IN1 Pin5 (Arduino Rx)	**IN2** Pin6 (Arduino Rx)	**IN3** Pin3 (Arduino Rx)	**IN4** Pin4 (Arduino Rx)	**MOTOR 1**	**MOTOR 2**
5V	GND	GND	5V	Giro hacia adelante	Giro hacia Atrás
GND	5V	5V	GND	Giro hacia Atrás	Giro hacia adelante
5V	GND	5V	GND	Giro hacia adelante	Giro hacia adelante
GND	5V	GND	5V	Giro hacia Atrás	Giro hacia Atrás
GND	GND	GND	GND	Se detiene	Se detiene

Fuente: Apolo K, Coba N, 2016.

El código correspondiente a la programación que se realizó en la tarjeta Arduino Uno se muestra en el ANEXO H.

CAPÍTULO III

3 MARCO DE RESULTADOS Y DISCUSIÓN

En este capítulo se muestra el resultado final del diseño y funcionalidades de cada una de las interfaces graficas diseñadas y programadas en MATLAB para el reconocimiento de comandos voz, así también se realiza un análisis de cada una de las gráficas obtenidas en el procesamiento.

Para el programa se consideró el cálculo de la energía de la voz para eliminar silencios indeseados al momento de procesar la señal de voz, debido a que esta energía varía dependiendo de la persona que emita el sonido por lo que toma un valor diferente por lo general en hombres mujeres y niños.

Como se explicó en el capítulo anterior el programa de reconocimiento consta de dos interfaces gráficas y un archivo.m cuyas funcionalidades serán explicadas posteriormente.

3.1 Programa de Reconocimiento de voz para Niños

El programa de reconocimiento de voz para niños consta de las interfaces graficas denominadas:

1. Base de datos
2. Reconocimiento de voz para niños

3.1.1 Interfaz Base de Datos

Esta interfaz gráfica consta de 25 botones que al ser accionados cumplen con los siguientes procesos y subprocesos descritos en el capítulo II:

1. **Adquisición de la voz:** grabar los comandos a reconocer
2. **Pre procesamiento:** normalización y determinación del umbral, segmentación, cálculo de energía, eliminación de silencios y filtrado preénfasis.

Se consideraron 25 grabaciones de cada comando para poder obtener una señal patrón de cada comando que contenga las características suficientes para ser diferenciada de las demás.

Se utilizó un diseño infantil ya que el programa de reconocimiento de voz está orientado a ese público en específico.

En esta interfaz además se puede observar las gráficas de la señal grabada en función al nivel de voz y número de muestras correspondientes a la señal grabada y la señal en la cual se ha eliminado los silencios y suavizado el espectro.

La Figura 1-3 corresponde a la Base de Datos para la grabación del comando derecha cabe recalcar que el mismo modelo fue utilizado para grabar los demás comandos de voz la única modificación que se realizo es el cambio de nombre para las diferentes grabaciones.

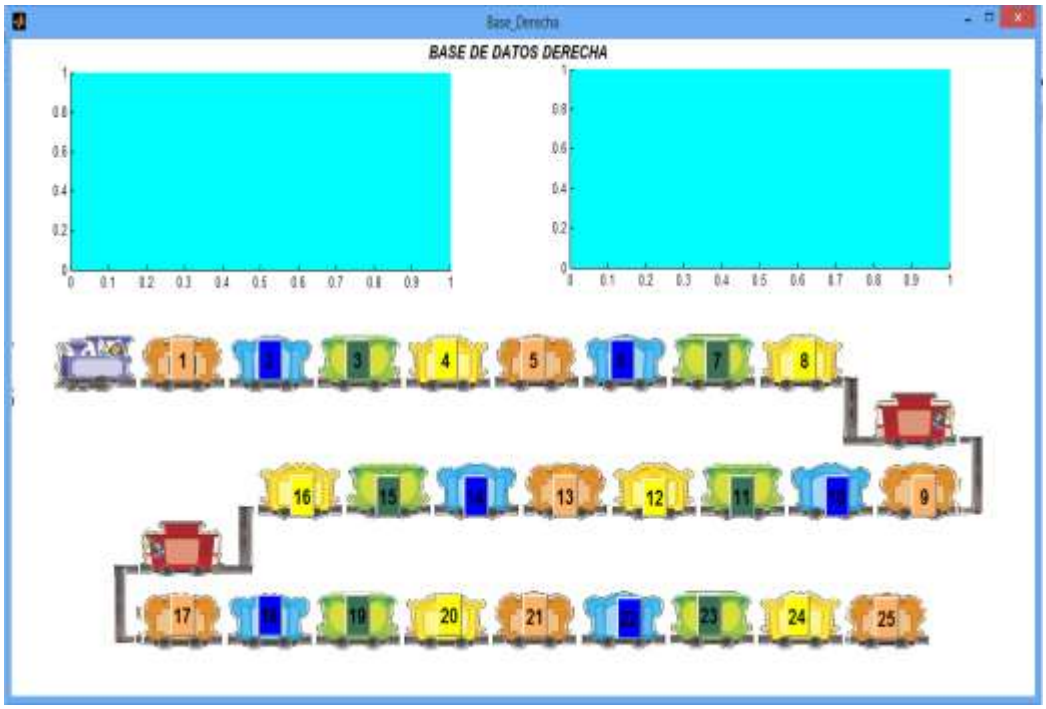


Figura 1-3 Interfaz Gráfica Base de Datos.

Fuente: Apolo K, Coba N, 2016.

3.1.2 Interfaz Reconocimiento de voz para Niños

Esta interfaz gráfica consta de dos bloque denominados Procesamiento de Señal y Reconocimiento.

3.1.2.1 Procesamiento de Señal

Este bloque es utilizado para el análisis de los procesos correspondientes a:

Adquisición: esta opción permite visualizar las gráficas en función al dominio de tiempo, frecuencia y número de muestras de la grabación ingresada por el usuario para realizar el proceso de reconocimiento.

Eliminación de silencios: al igual que la opción anterior nos permite visualizar las gráficas en función al dominio de tiempo, frecuencia y número de muestras de la grabación ingresada por el usuario a la cual se le ha eliminado los silencios.

LPC: este opción del menú permite visualizar las gráficas correspondientes a:

1. La señal de voz sin silencios a la cual se la ha normalizado y enventanado usando una ventana de hamming de 240 muestras a la cual denominaremos como señal enventanada.
2. La señal enventanada (amarilla), la señal predicha que ha sido reconstruida a partir de la señal enventanada utilizando los coeficientes LPC (roja) y el error de predicción (negro) obtenido de la resta de las dos señales mencionadas anteriormente.
3. Las señal usuario (azul) y la envolvente LPC.
4. Formantes de la señal con el filtro preénfasis.

Filtrado preénfasis: Se muestra mediante esta opción la gráfica en función del tiempo, frecuencia y número de muestras de la señal filtrada. El filtrado se realiza de la señal en la que se ha eliminado los silencios con el objetivo de acentuar frecuencias altas que pudieron ser afectadas al momento del realizar dicho proceso. Se utilizó un filtro FIR pasa altos de primer orden debido a que este es el más usado en el PDS por su funcionalidad.

Distancias: Por medio de esta opción se muestran las distancias entre cada señal patrón y la señal predicha, para corroborar que la menor distancia es la que se utiliza para el reconocimiento.

3.1.2.2 Reconocimiento

Mediante este bloque se realizan los mismos procesos que en la interfaz base de datos la diferencia es que este procedimiento se realiza para la señal de voz ingresada por el usuario que va a ser reconocida por lo cual además se aumentan el cálculo y comparación de distancias, cada uno de estos procesos se activan al ser accionado el botón respectivo, estos son:

Botón grabar: Adquisición de la voz, pre procesamiento y filtrado preénfasis.

Botón Identificar: Extracción de características, cálculo de distancias y comparación de distancias. Cuando se acciona este botón aparece un mensaje donde se muestra la palabra que ha sido reconocida.

Botón salir: Permite salir del programa.

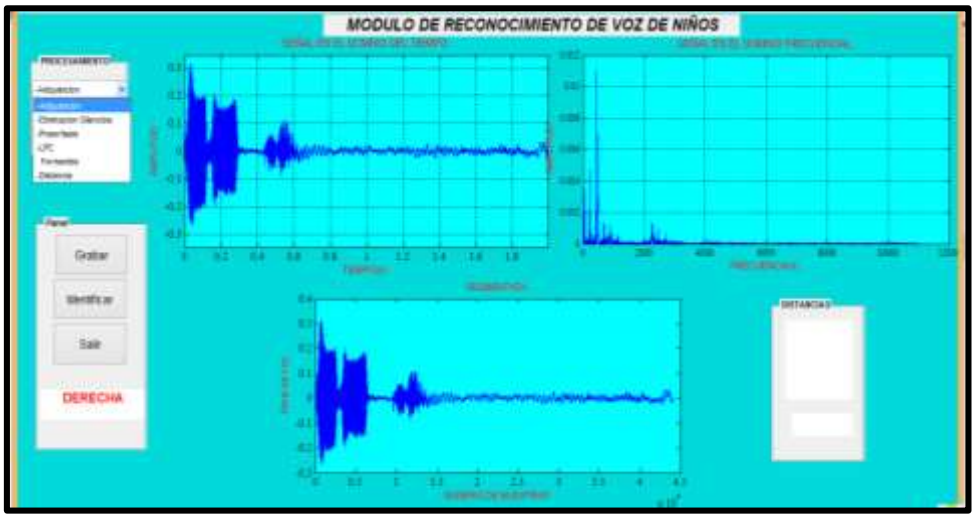


Figura 2-3 Interfaz Gráfica Reconocimiento

Fuente: Apolo K, Coba N, 2016.

3.2 Resultado de la Implementación del Módulo de Reconocimiento de Voz

Figura 3-3 Módulo de voz y caso práctico

Fuente: Apolo K, Coba N, 2016.

3.3 Pruebas y Resultados

3.3.1 Prueba-1

Para la fase de pruebas se consideró tres usuarios y 10 pronunciaciones de cada comando a reconocer dando un total de 50 pronunciaciones por usuario, esto se realizó considerando un nivel de ruido aproximado entre 50-60 dB, el comando de voz del usuario se lo adquirió en tiempo real.

- Usuario-1

Nombre: Roberto Quinchuela
Sexo: Masculino
Edad: 8 años

Figura 4-3 Foto de Usuario-1
Fuente: Apolo K, Coba N, 2016.

Tabla 1-3 Prueba-1 Usuario-1

	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA
1	✓	✓	✓	✓	✓
2	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	✓
4	✓	✓	✓	✓	✓
5	✓	✓	X	X	✓
6	✓	✓	✓	✓	✓
7	✓	X	✓	✓	✓
8	X	✓	✓	X	✓
9	✓	✓	✓	X	✓
10	✓	✓	✓	✓	✓
ACIERTOS	9	9	9	7	10
RESULTADO TOTAL	44 (aciertos)		88%		

Fuente: Apolo K, Coba N, 2016.

- Usuario-2

Nombre: Jeimi Parra.

Sexo: Femenino

Edad: 11 años

Figura 5-3 Foto de Usuario-2

Fuente: Apolo K, Coba N, 2016.

Tabla 2-3 Prueba-1 Usuario-2

	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA
1	✓	✓	✓	✓	✓
2	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	✓
4	✓	✓	X	✓	✓
5	✓	✓	X	✓	✓
6	✓	✓	✓	✓	X
7	✓	✓	X	✓	✓
8	✓	✓	X	X	✓
9	✓	✓	✓	X	X
10	✓	✓	✓	✓	✓
ACIERTOS	10	10	6	8	8
RESULTADO TOTAL	42 (aciertos)		84%		

Fuente: Apolo K, Coba N, 2016.

- Usuario-3

Nombre: José Quinchuela

Sexo: Masculino

Edad: 10 años

Figura 6-3 Foto de Usuario-3

Fuente: Apolo K, Coba N, 2016.

Tabla 3-3 Prueba1 Usuario-3

	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA
1	✓	✓	✓	✓	✓
2	✓	✓	✓	✓	✓
3	✓	✓	✓	X	✓
4	✓	X	✓	✓	✓
5	✓	✓	✓	✓	✓
6	X	✓	✓	✓	✓
7	✓	X	✓	✓	✓
8	✓	✓	✓	X	✓
9	✓	✓	✓	X	✓
10	X	✓	✓	✓	✓
ACIERTOS	9	8	10	7	10
RESULTADO TOTAL	43 (aciertos)		86%		

Fuente: Apolo K, Coba N, 2016.

3.3.2 Resultado Prueba-1

De acuerdo a la Prueba-1 realizada con el Modulo de Reconocimiento de voz se determinó que el porcentaje de fidelidad para el Usuario-1 es de 88%, para el Usuario-2 es de 84% y para el Usuario-3 es de 86%, obteniendo una fidelidad promedio de 86%.

3.3.2.1 Coeficientes LPC

En la Tabla 4-3, Tabla 5-3 y Tabla 6-3 se muestran los 23 coeficientes LPC de cada comando de voz a reconocer que han sido ingresados por cada usuario.

- Usuario-1

Tabla 4-3 Coeficientes LPC Usuario-1

	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	ALTO
1	-2,768	-2,374	-2,656	-2,621	-3,341
2	5,168	4,042	4,338	4,699	6,45
3	-7,311	-5,421	-5,218	-6,724	-9,063
4	8,227	5,832	4,522	7,602	9,752
5	-7,154	-4,691	-2,515	-6,668	-7,713
6	4,283	2,651	0,104	4,362	3,759
7	-0,742	-0,375	1,635	-1,302	0,466
8	-1,954	-1,416	-1,827	-1,425	-3,216
9	2,992	1,966	0,886	2,835	3,682
10	-2,197	-1,483	0,32	-2,609	-2,149
11	0,469	0,385	-0,961	1,205	-0,008
12	1,23	0,685	0,828	0,599	1,515
13	-1,942	-1,205	-0,281	-1,886	-1,762
14	1,585	1,074	-0,248	2,183	1,011
15	-0,562	-0,446	0,512	-1,49	0,054
16	-0,424	-0,168	-0,412	0,402	-0,742
17	1,008	0,676	0,109	0,64	1,008
18	-1,063	-0,723	0,12	-1,116	-0,874
19	0,74	0,64	-0,199	1,14	0,547
20	-0,356	-0,33	0,213	-0,806	-0,233
21	0,112	0,159	-0,137	0,409	0,044
22	0,02	-0,034	0,107	-0,174	0,044
23	-0,016	-0,01	-0,028	0,041	-0,038

Fuente: Apolo K, Coba N, 2016.

- Usuario-2

Tabla 5-3 Coeficientes LPC Usuario-2

	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA
1	-0,106	-0,327	-0,553	0,401	-1,157
2	0,835	0,235	-0,277	0,162	0,25
3	-0,359	-0,444	0,366	-0,593	0,143
4	0,318	0,345	0,172	-0,002	-0,055
5	-0,435	-0,132	0,011	0,166	0,004
6	0,265	0,411	0,121	0,664	0,382
7	-0,341	-0,368	-0,354	0,132	-0,597
8	0,132	0,122	0,395	0,079	0,624
9	-0,435	-0,387	-0,1	-0,295	-0,261
10	0,1	0,169	-0,014	0,184	-0,003
11	-0,4	-0,151	0,081	0,021	0,01
12	0,302	0,328	0,03	0,388	0,187
13	-0,214	-0,132	-0,108	-0,095	-0,467
14	0,299	0,21	0,024	0,156	0,454
15	-0,086	-0,188	0,033	-0,037	-0,03
16	0,111	0,02	-0,044	0,052	-0,215
17	-0,069	0,055	-0,117	-0,136	0,065
18	0,379	0,207	0,184	0,233	0,282
19	0,148	0,237	0,118	0,321	-0,123
20	0,335	0,042	0,003	0,3	0,135
21	-0,017	-0,126	-0,163	-0,016	-0,271
22	0,106	0,05	0,053	-0,052	-0,021
23	-0,127	0,127	0,108	-0,042	0,176

Fuente: Apolo K, Coba N, 2016.

- Usuario-3

Tabla 6-3 Coeficientes LPC Usuario-3

	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA
1	-2,326	-2,626	-1,032	-2,864	-3,341
2	4,58	5,486	0,696	5,733	6,45
3	-6,449	-8,382	-0,274	-8,746	-9,063
4	7,427	10,607	0,161	10,725	9,752
5	-6,568	-10,735	-0,727	-10,602	-7,713
6	4,048	8,609	1,077	8,169	3,759
7	-1,007	-4,683	-1,107	-3,94	0,466
8	-1,902	-0,105	0,739	-0,354	-3,216
9	3,012	3,711	-0,426	3,184	3,682
10	-2,58	-5,357	0,435	-3,729	-2,149
11	0,804	4,423	-0,586	2,262	-0,008
12	1,091	-1,705	0,751	0,192	1,515
13	-2,3	-1,683	-0,529	-2,071	-1,762
14	2,545	4,418	0,234	2,64	1,011
15	-1,597	-5,566	-0,055	-1,667	0,054
16	0,694	5,444	0,319	-0,112	-0,742
17	0,346	-4,019	-0,269	1,834	1,008
18	-0,622	2,493	0,257	-2,683	-0,874
19	0,652	-1,044	-0,128	2,752	0,547
20	-0,36	0,29	-0,036	-2,051	-0,233
21	0,122	0,1	-0,017	1,206	0,044
22	-0,026	-0,084	0,022	-0,509	0,044
23	-0,042	0,047	0,012	0,122	-0,038

Fuente: Apolo K, Coba N, 2016.

3.3.2.2 Coeficientes LPC Señales Patrón

En la Tabla 7-3, Tabla 8-3, Tabla 9-3 se muestran los 23 coeficientes LPC de cada comando de voz correspondiente a las Señales Patrón de cada usuario que sirvieron como base principal para el reconocimiento.

- Usuario-1

Tabla 7-3 Coeficientes LPC

	PATRON DERECHA	PATRON IZQUIERDA	PATRON ADELANTE	PATRON ATRÁS	PATRON PARA
1	-2,612	-2,476	-2,745	-2,617	-3,315
2	4,702	4,317	4,64	4,538	6,117
3	-6,443	-5,99	-5,775	-6,331	-7,949
4	6,964	6,672	5,305	6,977	7,73
5	-5,842	-5,769	-3,339	-6,027	-5,334
6	3,299	3,699	0,715	3,916	1,988
7	-0,376	-1,223	1,403	-1,158	0,931
8	-1,806	-0,849	-2,061	-1,018	-2,476
9	2,521	1,762	1,377	2,047	2,456
10	-1,784	-1,476	-0,06	-1,817	-1,181
11	0,359	0,482	-0,922	0,645	-0,408
12	0,999	0,6	1,096	0,579	1,439
13	-1,582	-1,158	-0,623	-1,281	-1,445
14	1,293	1,123	-0,082	1,353	0,664
15	-0,434	-0,597	0,559	-0,703	0,233
16	-0,401	0,025	-0,586	-0,021	-0,63
17	0,925	0,443	0,277	0,581	0,623
18	-1,01	-0,522	0,066	-0,756	-0,418
19	0,769	0,463	-0,258	0,667	0,219
20	-0,456	-0,25	0,316	-0,395	-0,101
21	0,211	0,105	-0,206	0,184	0,066
22	-0,054	-0,021	0,125	-0,047	-0,029
23	0,025	-0,018	-0,035	0,006	0,003

Fuente: Apolo K, Coba N, 2016.

- Usuario-2

Tabla 8-3 Coeficientes LPC Señales Patrón

	PATRON DERECHA	PATRON IZQUIERDA	PATRON ADELANTE	PATRON ATRÁS	PATRON PARA
1	0,169	-0,356	-0,802	1,04	-0,921
2	0,78	0,134	0,075	0,123	0,055
3	-0,054	-0,184	0,23	-0,197	0,192
4	0,492	0,09	0,277	-0,54	0,119
5	-0,297	-0,04	-0,219	0,222	-0,182
6	0,212	0,305	0,329	0,209	0,204
7	-0,418	-0,303	-0,453	0,436	-0,214
8	0,036	0,162	0,386	-0,118	0,304
9	-0,529	-0,254	-0,243	-0,048	-0,228
10	-0,14	0,096	0,065	-0,111	0,133
11	-0,527	-0,017	0,045	0,111	-0,153
12	0,166	0,273	0,218	-0,044	0,308
13	-0,122	-0,055	-0,23	0,301	-0,191
14	0,414	0,076	0,105	0,107	0,114
15	0,047	-0,115	-0,018	0,064	-0,059
16	0,351	0,005	0,14	-0,136	0,046
17	0,184	0,06	-0,103	-0,026	-0,124
18	0,352	0,238	0,203	-0,058	0,25
19	0,206	0,024	0,064	0,302	0,027
20	0,26	0,089	-0,029	0,217	0,03
21	0,057	-0,059	-0,085	0,119	-0,118
22	0,041	0,073	0,146	-0,072	0,029
23	0,017	0,074	0,078	-0,017	0,084

Fuente: Apolo K, Coba N, 2016.

- Usuario-3

Tabla 9-3 Coeficientes LPC Señales Patrón

	PATRON DERECHA	PATRON IZQUIERDA	PATRON ADELANTE	PATRON ATRÁS	PATRON PARA
1	-2,275	-2,585	-0,937	-2,744	-3,056
2	4,469	5,248	0,539	5,315	5,955
3	-6,355	-7,827	-0,158	-7,93	-8,447
4	7,508	9,689	0,078	9,544	9,202
5	-7,093	-9,583	-0,628	-9,392	-7,382
6	5,233	7,517	0,911	7,368	3,714
7	-2,698	-4,071	-0,918	-3,972	0,276
8	-0,025	0,146	0,538	0,324	-2,702
9	1,512	2,558	-0,316	2,422	2,764
10	-1,831	-3,567	0,389	-3,358	-1,112
11	0,953	2,611	-0,495	2,731	-0,901
12	0,424	-0,599	0,657	-0,976	2,131
13	-1,564	-1,578	-0,442	-0,868	-2,046
14	2,166	2,985	0,254	2,039	1,155
15	-1,817	-3,139	-0,108	-2,174	-0,146
16	1,348	2,541	0,322	1,608	-0,346
17	-0,512	-1,319	-0,251	-0,562	0,371
18	0,078	0,382	0,216	-0,271	-0,049
19	0,188	0,316	-0,121	0,771	-0,154
20	-0,197	-0,425	-0,014	-0,804	0,251
21	0,136	0,389	-0,056	0,586	-0,197
22	-0,06	-0,174	0,08	-0,293	0,103
23	0,005	0,07	0,0004	0,099	-0,017

Fuente: Apolo K, Coba N, 2016.

3.3.2.3 Distancias Totales

En la Tabla 10-3, Tabla 11-3 y Tabla 12-3 se muestran las Distancias Totales calculadas entre los vectores que contiene las los coeficientes LPC patrón y el vector LPC del cada voz ingresada por el usuario.

- Usuario-1

Tabla 10-3 Distancias Totales Usuario-1

USUARIO INGRESA	SEÑAL PATRON					MENOR DISTANCIA
	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA	
DERECHA	6,167	12,848	61,38	7,86	18,312	6,167 **(D)**
IZQUIERDA	5,636	4,673	19,327	6,987	21,628	4,673 **(IZ)**
ADELANTE	50,231	49,344	2,87	58,626	42,365	2,87 **(AD)**
ATRÁS	8,01	9,985	68,815	5,379	31,511	5,379 **(AT)**
PARA	27,154	43,535	80,741	35,483	18,366	18,366 **(P)**

Fuente: Apolo K, Coba N, 2016.

D=Derecha **IZ**=Izquierda **AD**=Adelante **AT**=Atrás **P**=Para

- Usuario-2

Tabla 11-3 Distancias Totales Usuario-2

USUARIO INGRESA	SEÑAL PATRON					MENOR DISTANCIA
	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA	
DERECHA	0,541	1,221	2,044	2,2	2,063	0,541 **(D)**
IZQUIERDA	1,416	0,299	1,019	0,842	1,08	0,299 **(IZ)**
ADELANTE	3,299	0,806	0,462	1,918	0,569	0,462 **(AD)**
ATRÁS	2,514	1,359	3,215	0,574	3,143	0,574 **(AT)**
PARA	3,981	1,566	0,883	3,877	0,867	0,867 **(P)**

Fuente: Apolo K, Coba N, 2016.

D=Derecha **IZ**=Izquierda **AD**=Adelante **AT**=Atrás **P**=Para

- Usuario-3

Tabla 12-3 Distancias Totales Usuario-3

USUARIO INGRESA	SEÑAL PATRON					MENOR DISTANCIA
	DERECHA	IZQUIERDA	ADELANTE	ATRÁS	PARA	
DERECHA	14,007	60,041	31,44	54,012	25,491	14,007 **(D)**
IZQUIERDA	133,133	43,795	308,507	71,384	231,239	43,795 **(IZ)**
ADELANTE	188,22	361,74	0,217	342,194	264,093	0,217 **(AD)**
ATRÁS	75,905	43,393	163,09	29,612	106,564	29,612 **(AT)**
PARA	59,713	99,809	318,1	83,765	6,688	6,688 **(P)**

Fuente: Apolo K, Coba N, 2016.

D=Derecha **IZ**=Izquierda **AD**=Adelante **AT**=Atrás **P**=Para

3.3.3 Prueba-2

Para esta segunda prueba se consideró un total de 14 grabaciones por usuario considerando el mismo margen de ruido de la prueba anterior.

- Usuario-1

Tabla 13-3 Prueba2 Usuario-1

#	COMANDO	RECONOCIMIENTO	
1	Derecha	✓	
2	Izquierda	✓	
3	Derecha	✓	
4	Izquierda	✓	
5	Derecha	✓	
6	Izquierda	✓	
7	Adelante	✓	
8	Atrás	✓	
9	Adelante	X	
10	Derecha	✓	
11	Izquierda	✓	
12	Adelante	✓	
13	Atrás	✓	
14	Para	✓	
TOTAL		13	92,85 %

Fuente: Apolo K, Coba N, 2016.

- Usuario-2

Tabla 14-3 Prueba2 Usuario-2

#	COMANDO	RECONOCIMIENTO	
X	Derecha	✓	
2	Izquierda	✓	
3	Derecha	✓	
4	Izquierda	✓	
5	Derecha	✓	
6	Izquierda	✓	
7	Adelante	X	
8	Atrás	✓	
9	Adelante	✓	
10	Derecha	✓	
11	Izquierda	✓	
12	Adelante	X	
13	Atrás	✓	
14	Para	✓	
TOTAL		12	85,71%

Fuente: Apolo K, Coba N, 2016.

- Usuario-3

Tabla 15-3 Prueba2 Usuario-3

#	COMANDO	RECONOCIMIENTO	
1	Derecha	✓	
2	Izquierda	✓	
3	Derecha	✓	
4	Izquierda	✓	
5	Derecha	✓	
6	Izquierda	✓	
7	Adelante	✓	
8	Atrás	X	
9	Adelante	✓	
10	Derecha	✓	
11	Izquierda	✓	
12	Adelante	✓	
13	Atrás	X	
14	Para	✓	
TOTAL		14	85,71 %

Fuente: Apolo K, Coba N, 2016.

3.3.4 Resultado Prueba-2

Con la realización de la Prueba-2 se determinó que el porcentaje de fidelidad para el Usuario-1 es de 92,85, para el Usuario-2 es de 85,71% y para el Usuario-3 es de 85,71% obteniendo una fidelidad promedio de 88,09 %.

3.3.5 Prueba 3 (Caso Práctico)

En esta tercera prueba se consideró un total de 5 grabaciones por usuario considerando un margen de ruido de 70db debido a la emisión de ruido del Carro Robot 4wd

Tabla 16-3 Prueba-3

#	COMANDO	Usuario-1		Usuario-2		Usuario-3	
1	Derecha	✓		✓		✓	
2	Izquierda	✓		✓		X	
3	Adelante	✓		X		✓	
4	Atrás	✓		✓		X	
5	Para	✓		✓		✓	
TOTAL		5	100 %	4	80%	3	60 %

Fuente: Apolo K, Coba N, 2016.

Esta tercera Prueba dio como resultado un porcentaje de fidelidad para el Usuario-1 es de 100%, para el Usuario-2 es de 80% y para el Usuario-3 es de 60% obteniendo una fidelidad promedio de 80 %.

3.3.6 Resultado Final

De acuerdo a las pruebas realizadas anteriormente se determinó que el Módulo de Reconocimiento de voz para Niños tiene una efectividad total de 84,70%

Tabla 17-3 Pruebas Totales

	PRUEBA 1	PRUEBA 2	PRUEBA 3
USUARIO 1	88	92,85	100
USUARIO 2	84	85,71	80
USUARIO 3	86	85,71	60
SUBTOTAL	86	88,09	80
TOTAL	84,70%		

Fuente: Apolo K, Coba N, 2016.

Gráfica 1-3 Resultados
Fuente: Apolo K, Coba N, 2016.

3.3.7 Comparación con otro Módulo existente en el mercado

Se realizó una comparación entre el Módulo de Reconocimiento de voz para Niños y el Módulo utilizado en la tesis "DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE MOVIMIENTO DIRECCIONAL DE UNA SILLA DE RUEDAS PARA SER CONTROLADA POR RECONOCIMIENTO DE UN PATRÓN DE VOZ MEDIANTE ELECTRÓNICA DE POTENCIA Y MOTORES DC COMO ACTUADORES "(Méndez et al., 2015, pp.73-75) donde se utilizó la tarjeta de reconocimiento de voz SR-07, el cual arrojo una efectividad de 77% , mientras que el módulo implementado en el Trabajo de Titulación posee una efectividad de 84,70%. Cabe recalcar que ambos casos se utilizó los mismos comandos aunque el número de comandos de voz para obtener dicho resultado es mayor en este Trabajo de Titulación.

3.4 Análisis de Resultados

3.4.1 Derecha

Las gráficas mostradas en la interfaz de reconocimiento de voz representa la señal ingresada por el usuario en función del tiempo, frecuencia y número de muestras, además se ha desglosado algunas gráficas para poder realizar su análisis.

- **Adquisición**

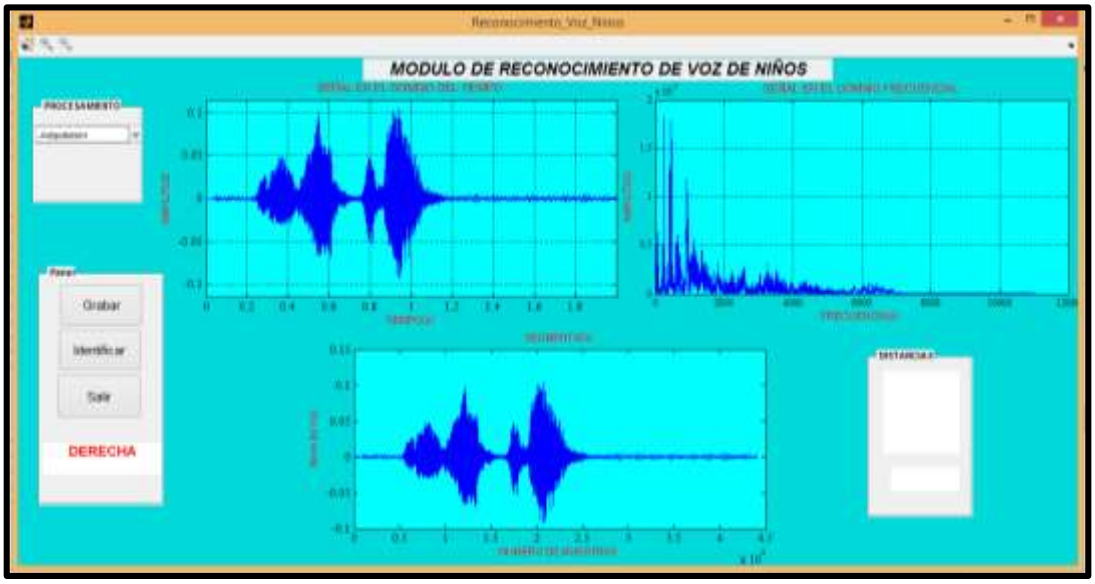


Figura 7-3 Interfaz Gráfica Adquisición

Fuente: Apolo K, Coba N, 2016.

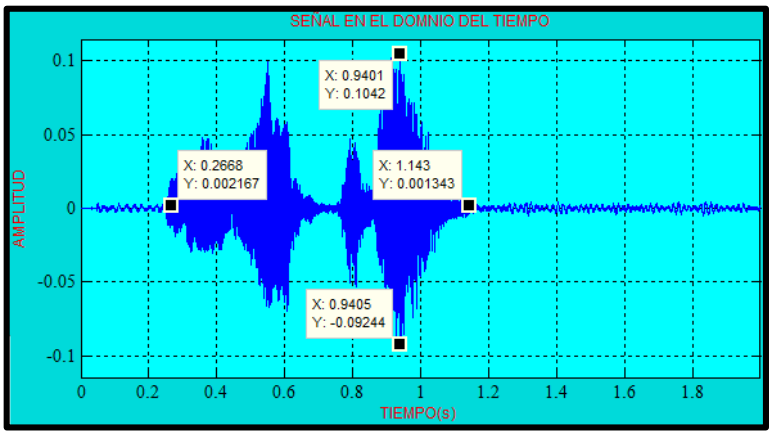


Figura 8-3 Señal original en función del tiempo

Fuente: Apolo K, Coba N, 2016.

En la Figura 8-3 se puede observar la representación gráfica de la señal de voz en el dominio del tiempo, donde el tiempo de duración de la grabación es de 2 segundos y además se puede ver que el tiempo de duración correspondiente al comando derecha se encuentra aproximadamente entre los valores (1,143-0,2668) segundos dando como un total de 0,87 s por lo cual el resto de la señal representa el ruido del ambiente o periodos en silencios. Los valores de Amplitud se encuentran aproximadamente entre [-0,09 0.10]

- **Eliminación de silencios**

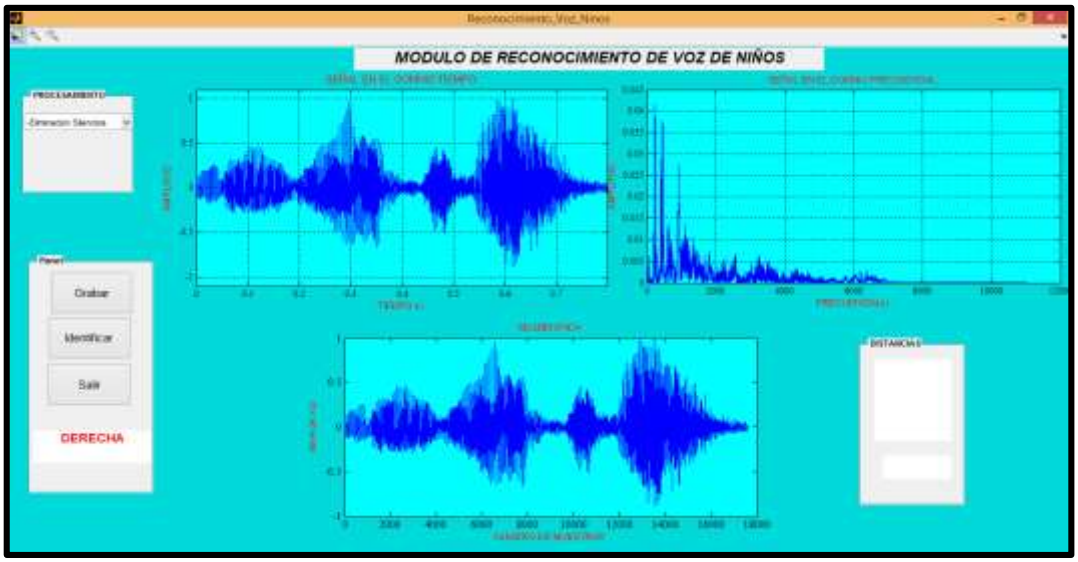


Figura 9-3 Señal de voz eliminada silencios del comando derecha

Fuente: Apolo K, Coba N, 2016.

En la Figura 9-3 se puede observar la representación gráfica de la señal de voz normaliza en amplitud en la cual se ha eliminado los silencios de acuerdo a la condición del umbral ya que los valores de energía que se encuentran por debajo del umbral son eliminados.

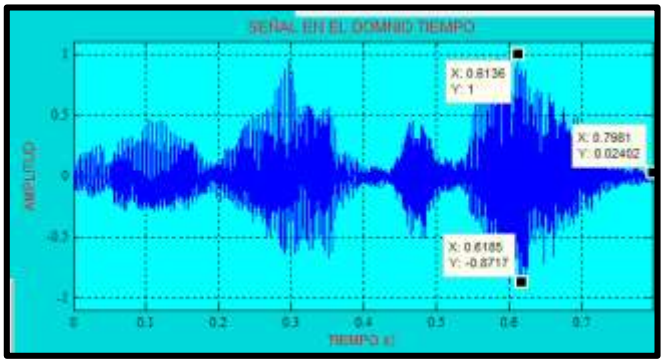


Figura 10-3 Señal eliminada los Silencios en función del tiempo

Fuente: Apolo K, Coba N, 2016.

En la Figura 10-3 la señal de voz se encuentra hasta 0,7981 segundos con lo cual si realiza una comparación con la Figura 8-3 se corrobora que el ruido del ambiente y los periodos de silencios fueron eliminados. Los valores de Amplitud se encuentran aproximadamente entre [-0,8 1].

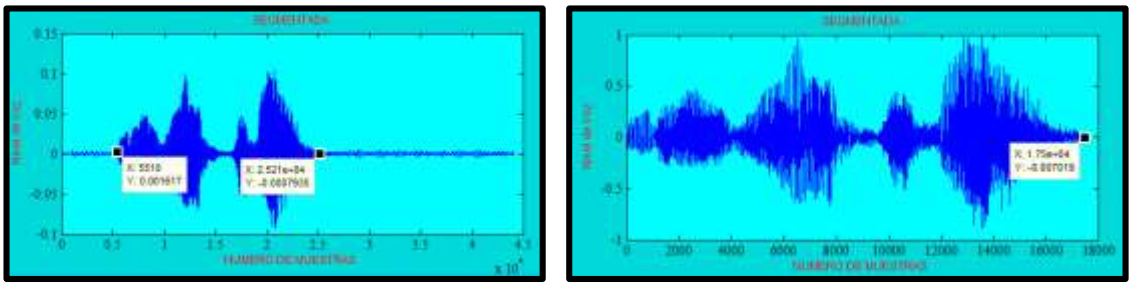


Figura 11-3 Señal original y Señal eliminada los Silencios en función del número de muestras

Fuente: Apolo K, Coba N, 2016.

Tabla 18-3 Comparación en función al número de muestras

	NÚMERO TOTAL DE MUESTRAS
SEÑAL ORIGINAL	25210-5510 =19700 (aproximado)
SEÑAL SIN SILENCIOS	17500
TOTAL DE MUESTRAS ELIMINADAS	2200

Fuente: Apolo K, Coba N, 2016.

- **Preénfasis**

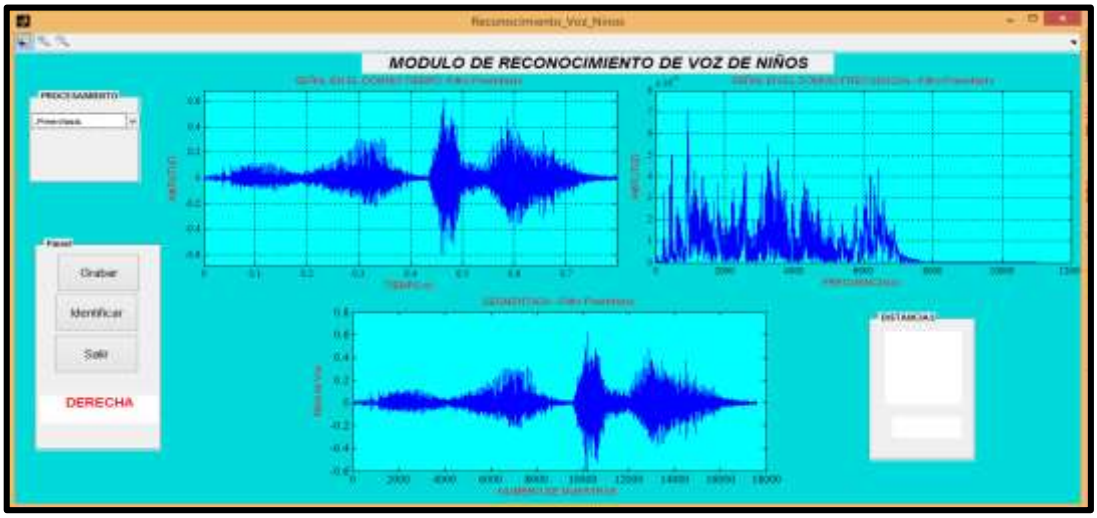


Figura 12-3 Señales de voz con Filtro Preénfasis

Fuente: Apolo K, Coba N, 2016.

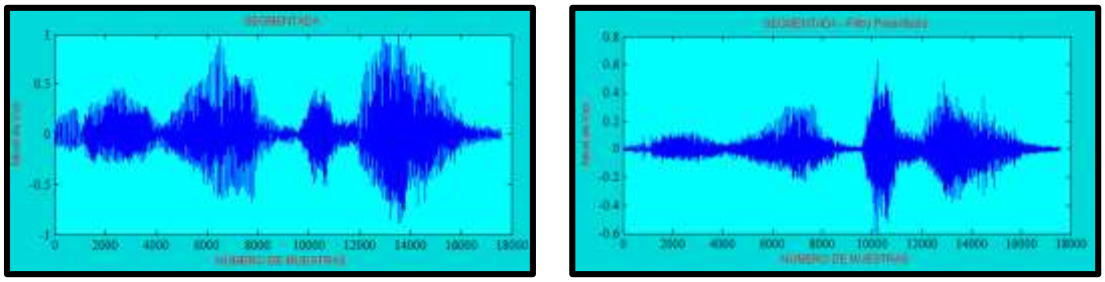


Figura 13-3 Señal Silencios sin filtro y Señal Silencios con filtro

Fuente: Apolo K, Coba N, 2016.

En la Figura 13-3 se muestran las señales en función al número de muestras correspondientes a la señal que ha sido eliminada los silencios, esta señal pasa por el filtro preénfasis y se puede denotar que este filtro realiza la acentuación de las frecuencias altas de la señal correspondientes a las consonantes (en este caso la ch) ya que corresponde a un sonido sordo es por ello que la acentuación de estas señales son parecidos a ruido, pero son generadas como un ruido gaussiano ya que posee información de la señal en estas frecuencias altas , así también mantiene una relación uniforme de la señal de voz.

- **Análisis LPC**

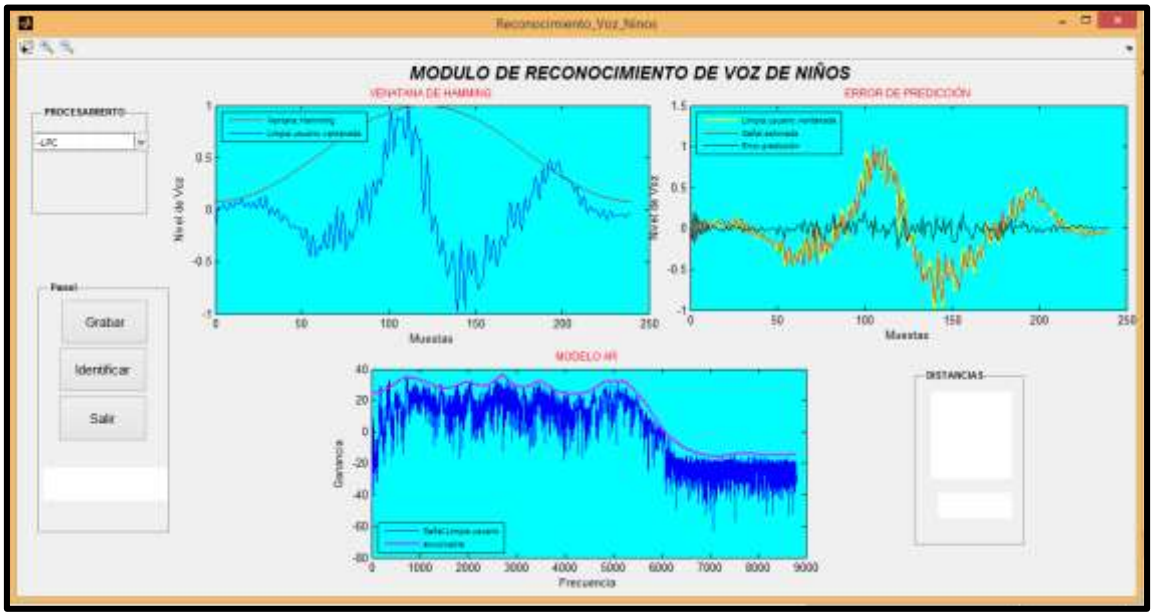


Figura 14-3 Análisis de LPC comando Derecha

Fuente: Apolo K, Coba N, 2016.

Para el análisis de LPC cabe recalcar que con la función lpc de Matlab se obtuvieron los coeficientes LPC los cuales son de gran importancia para encontrar la envolvente de la señal de

voz. A su vez dichos coeficientes se pueden remplazar en las ecuaciones de Predicción de Error para obtener cada una de las gráficas correspondientes a ese proceso.

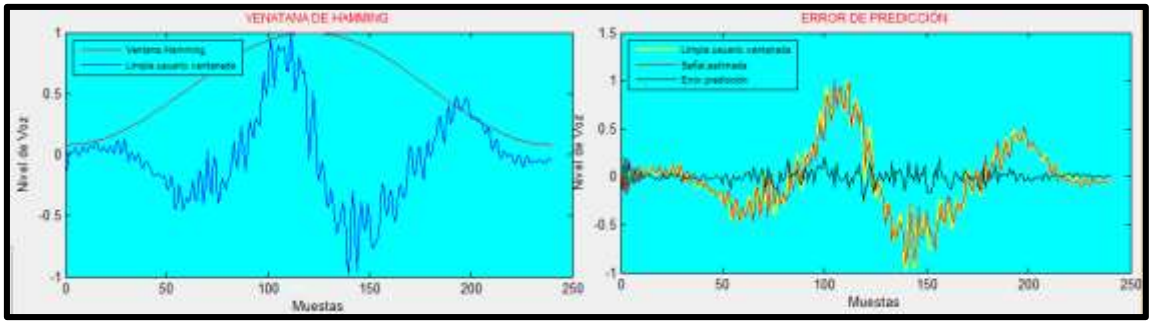


Figura 15-3 Señal Limpia_Usuario enventanada y Error de Predicción

Fuente: Apolo K, Coba N, 2016.

En la Figura 15-3 se muestra la señal Limpia_Usuario la cual ha sido enventanada a 240 muestras con una ventana de hamming, en el Error de Predicción se tiene la Señal Limpia_Usuario enventanada (amarilla), la Señal estimada (roja) ,se pude denotar que esta señal estimada o predicha si tiene una buena aproximación a la señal original dado que toma muestras pasadas y presentes para obtener dicha señal predicha , además se puede observar que el error entre estas dos señales es mínimo, esta señal de error de predicción también representa la señal de excitación la cual está conformada por sonidos sonoros y sordos , considerando que se va a tener mayor margen de error en los sonidos sordos.

- **Distancias**

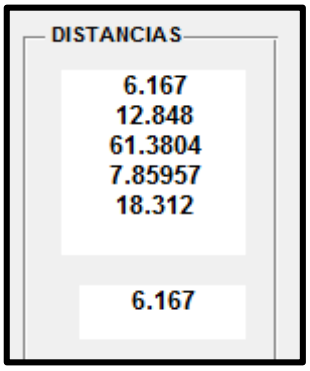


Figura 16-3 Distancias Totales

Fuente: Apolo K, Coba N, 2016.

En la figura anterior se muestra las Distancias Totales entre la señal de voz ingresada por el usuario (Derecha) y las señales patrón. Dando como resultado el reconocimiento del comando de

voz Derecha ya que este posee una menor distancia en comparación de las demás distancias calculadas.

3.4.2 Izquierda

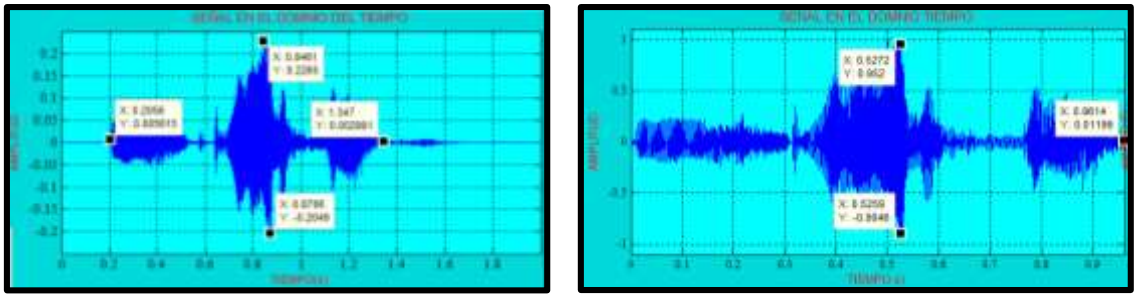


Figura 17-3 Señal original (izquierda) y señal Silencios en función del tiempo

Fuente: Apolo K, Coba N, 2016.

Tabla 19-3 Comparación entre la Señal Original y Silencios

Señal Original		**Señal Eliminada Silencios**	
Tiempo de grabación Total	2 s		
Tiempo del comando de Voz (aproximado)	(1,347-0,2056)= 1,14 s	Tiempo del comando de Voz	0,9614 s
Amplitud Máxima	[-0,2 0.8]	Amplitud Máxima	[-0,8 1]

Fuente: Apolo K, Coba N, 2016.

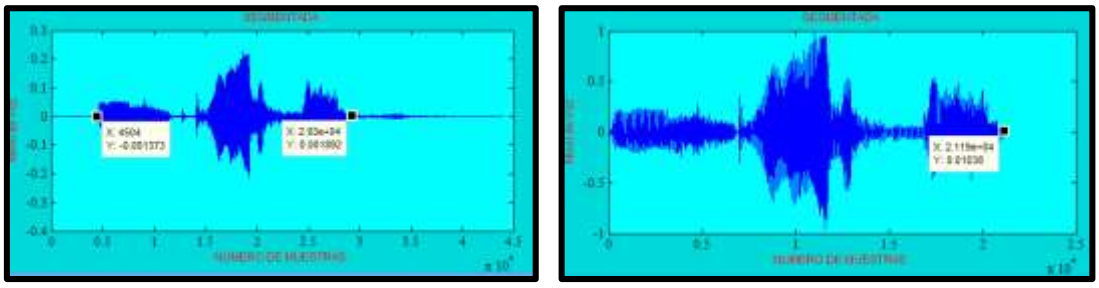


Figura 18-3 Señal original (izquierda) y señal Silencios en función de número de muestras

Fuente: Apolo K, Coba N, 2016.

Tabla 20-3 Comparación en función al número de muestras

	NÚMERO TOTAL DE MUESTRAS
SEÑAL ORIGINAL	29300-4504 =24796
SEÑAL SIN SILENCIOS	21190
TOTAL DE MUESTRAS ELIMINADAS	**3606**

Fuente: Apolo K, Coba N, 2016.

- **Preénfasis**

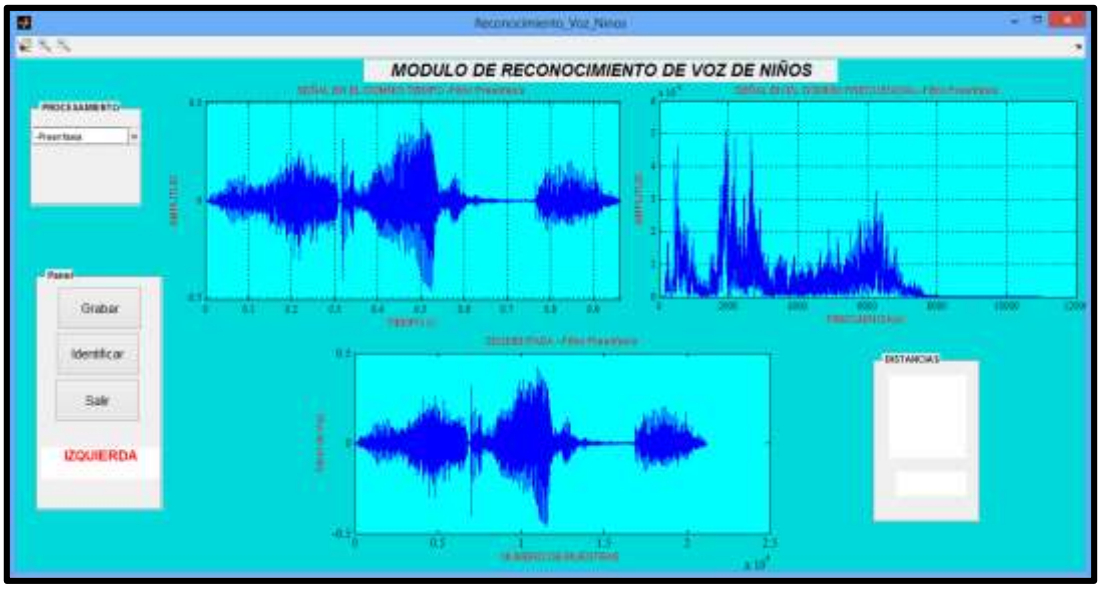


Figura 19-3 Señal de voz filtrada

Fuente: Apolo K, Coba N, 2016.

En la Figura 19-3 se muestran las señales en función al tiempo, número de muestras y frecuencia correspondientes a la señal con el filtro preénfasis y se puede denotar que este filtro realiza la acentuación de las frecuencias altas de la señal correspondientes a la consonante (z) ya que corresponde a un sonido sordo, así también se mantiene una relación uniforme de la señal de voz.

- **Análisis LPC**

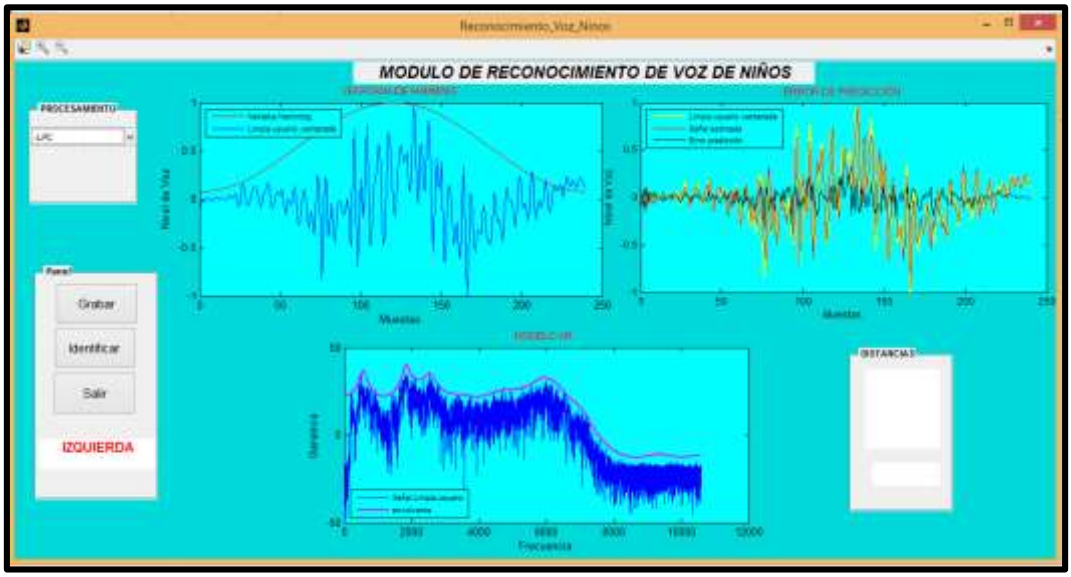


Figura 20-3 Análisis LPC Izquierda

Fuente: Apolo K, Coba N, 2016.

- **Distancia**

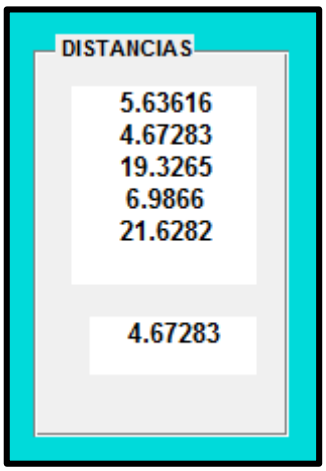


Figura 21-3 Distancias Totales

Fuente: Apolo K, Coba N, 2016.

3.4.3 Adelante

- **Adquisición**

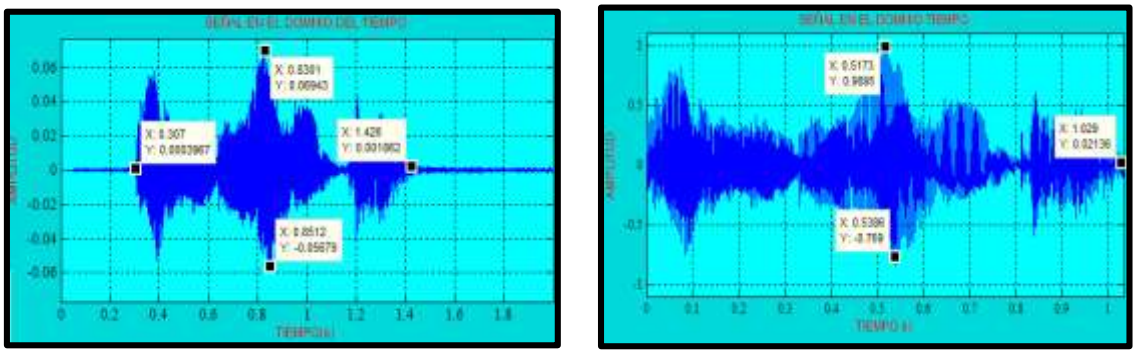


Figura 22-3 Señal original (adelante) y señal Silencios en función del tiempo

Fuente: Apolo K, Coba N, 2016.

Tabla 21-3 Comparación entre la Señal original y la Señal Silencios

Señal Original		**Señal Eliminada Silencios**	
Tiempo de grabación Total	2 s		
Tiempo del comando de Voz (aproximado)	(1,426-0,307)= 1,12 s	Tiempo del comando de Voz	1,029 s
Amplitud Máxima	[-0,05 0.069]	Amplitud Máxima	[-0,769 0,9895]

Fuente: Apolo K, Coba N, 2016.

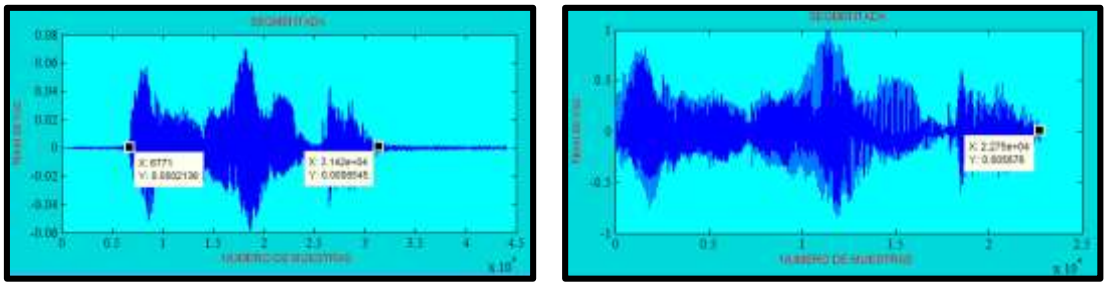


Figura 23-3 Señal original (adelante) y señal Silencios en función de número de muestras

Fuente: Apolo K, Coba N, 2016.

Tabla 22-3 Comparación entre la Señal Original y Silencios

	NÚMERO TOTAL DE MUESTRAS
SEÑAL ORIGINAL	31420-6771 =24649
SEÑAL SIN SILENCIOS	22750
TOTAL DE MUESTRAS ELIMINADAS	**1899**

Fuente: Apolo K, Coba N, 2016.

- **Preénfasis**

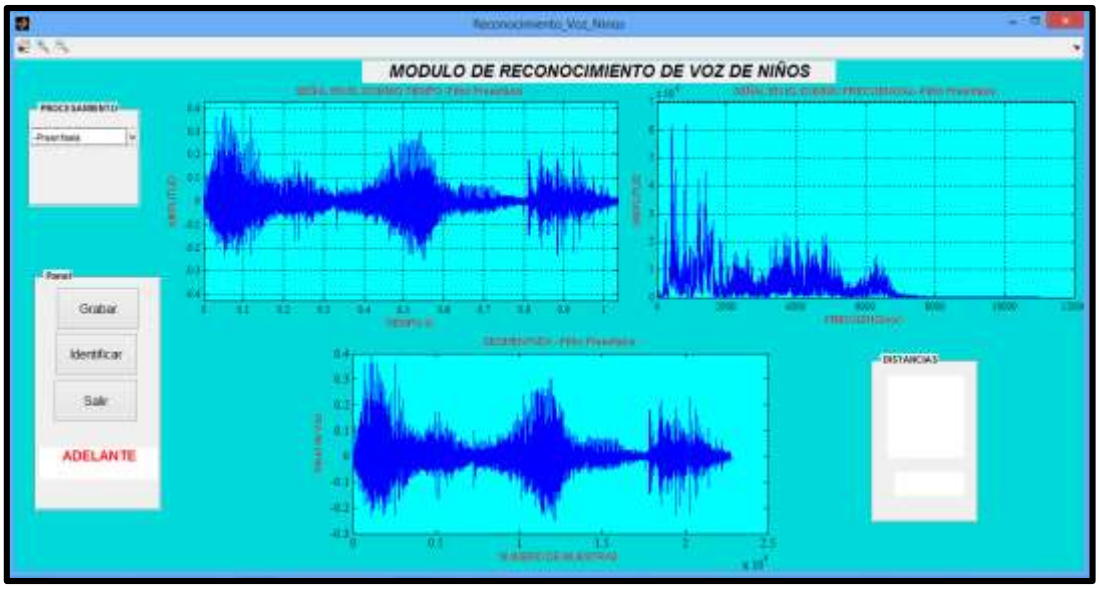


Figura 24-3 Interfaz Gráfica Preénfasis

Fuente: Apolo K, Coba N, 2016.

En la Figura 24-3 se muestran las señales en función al tiempo, número de muestras y frecuencia correspondientes a la señal eliminada Silencios con el filtro preénfasis y se puede denotar que no existe una acentuación de frecuencias debido a que este comando de voz no tiene sonidos sordos sin embargo se mantiene una relación uniforme de la señal de voz.

- **Análisis LPC**

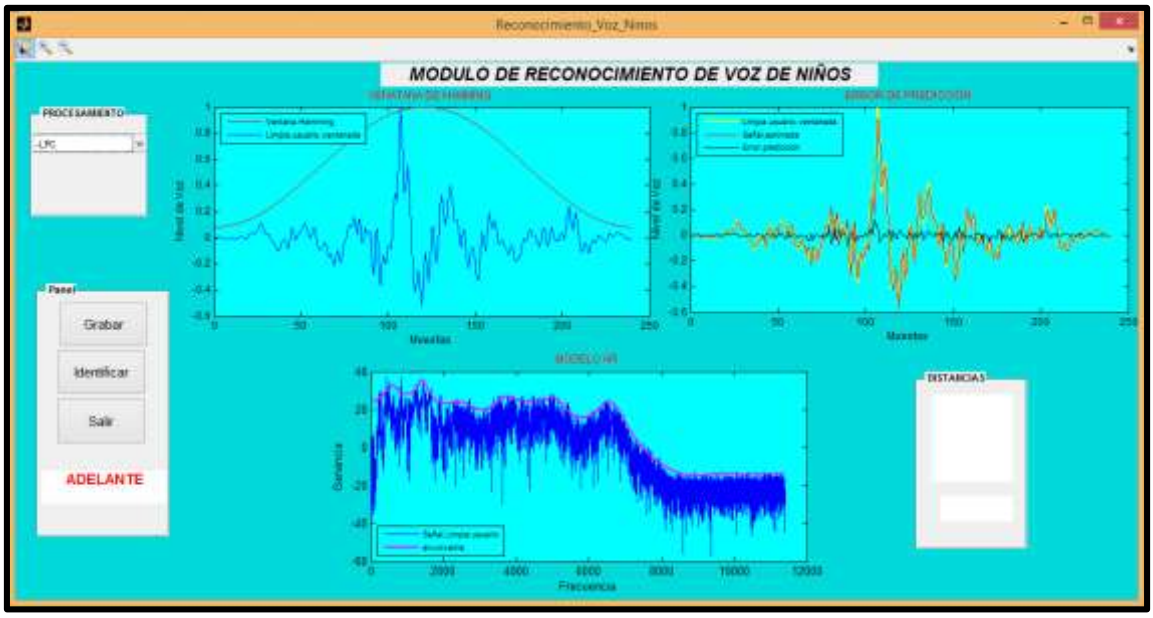


Figura 25-3 Interfaz Gráfica LPC (adelante)

Fuente: Apolo K, Coba N, 2016.

- **Distancia**

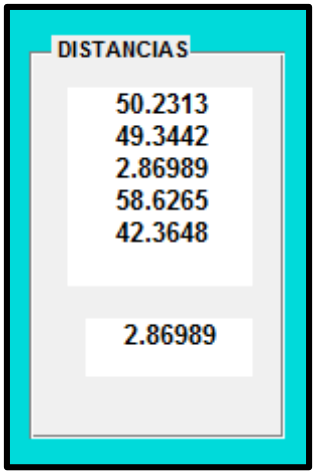


Figura 26-3 Distancias Totales

Fuente: Apolo K, Coba N, 2016.

3.4.4 Atrás

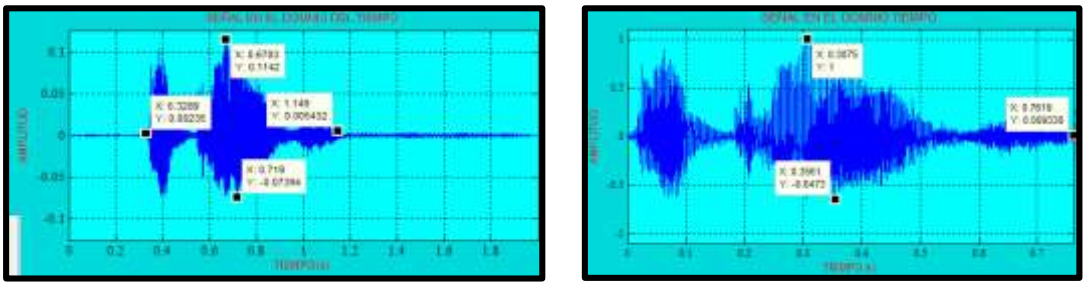


Figura 27-3 Señal original (atrás) y señal Silencios en función del tiempo

Fuente: Apolo K, Coba N, 2016.

Tabla 23-3 Comparación entre la Señal original y Señal Silencios

Señal Original		**Señal Eliminada Silencios**	
Tiempo de grabación Total	2 s		
Tiempo del comando de Voz (aproximado)	(1,149-0,3289)=0,82s	Tiempo del comando de Voz	0,79s
Amplitud Máxima	[-0,07394 0,1142]	Amplitud Máxima	[-0,6473 1]

Fuente: Apolo K, Coba N, 2016.

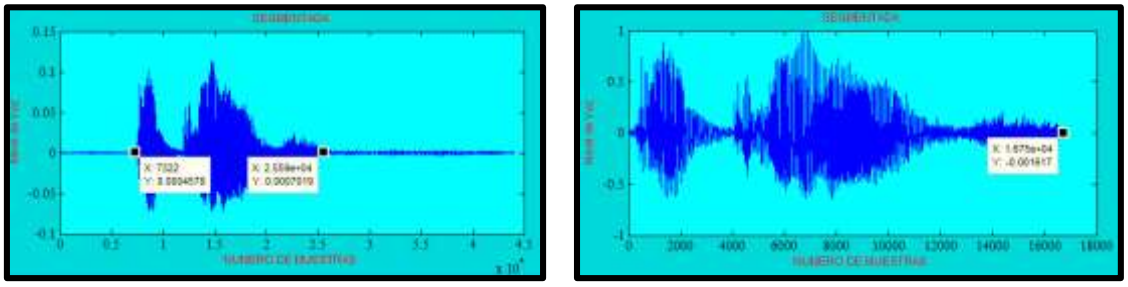


Figura 28-3 Señal original (atrás) y señal Silencios en función de número de muestras

Fuente: Apolo K, Coba N, 2016.

Tabla 24-3 Comparación entre la Señal Original y Señal Silencios

	NÚMERO TOTAL DE MUESTRAS
SEÑAL ORIGINAL	25590-7322 =18268
SEÑAL SIN SILENCIOS	16750
TOTAL DE MUESTRAS ELIMINADAS	**1518**

Fuente: Apolo K, Coba N, 2016.

- **Preénfasis**

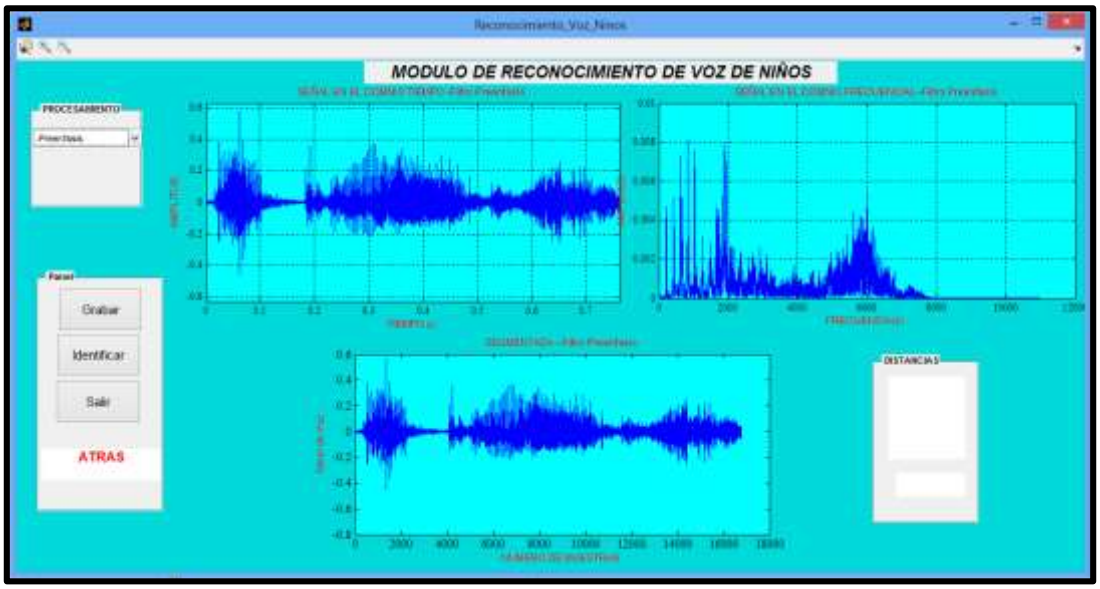


Figura 29-3 Interfaz Gráfica Preénfasis

Fuente: Apolo K, Coba N, 2016.

En estas Figura 29-3 se muestran las señales en función al tiempo, número de muestras y frecuencia correspondientes a la señal silencios con el filtro preénfasis y se puede denotar que este filtro realiza la acentuación de las frecuencias altas de la señal correspondiente a la consonante (s) ya que corresponde a un sonido sordo además se mantiene una relación uniforme de la señal de voz.

- **Análisis LPC**

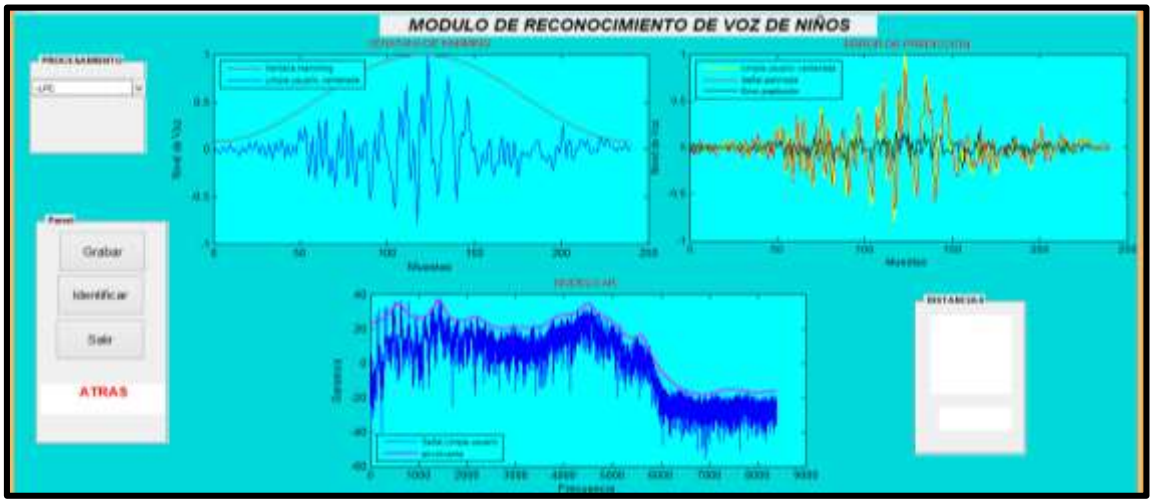


Figura 30-3 Interfaz Gráfica LPC (atrás)

Fuente: Apolo K, Coba N, 2016.

- **Distancia**

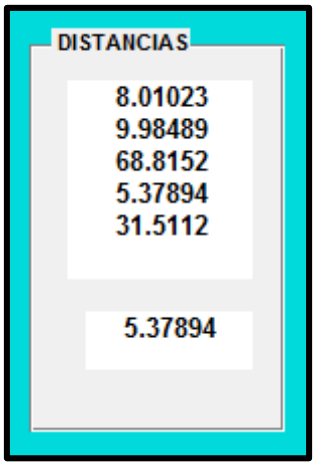
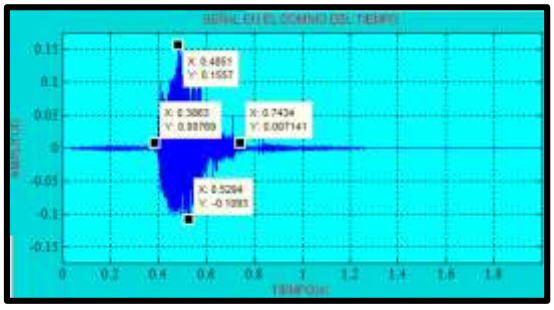


Figura 31-3 Distancias Totales

Fuente: Apolo K, Coba N, 2016

3.4.5 Para

- **Adquisición**



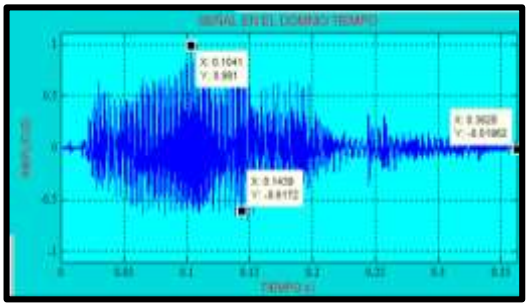


Figura 32-3 Señal original (para) y señal Silencios en función del tiempo

Fuente: Apolo K, Coba N, 2016.

Tabla 25-3 Comparación entre la Señal original y Señal Silencios

Señal Original		**Señal Eliminada Silencios**	
Tiempo de grabación Total	2 s		
Tiempo del comando de Voz (aproximado)	(0,7434-0,3863)=0,3571 s	Tiempo del comando de Voz	0,3628 s
Amplitud Máxima	[-0,1093 0,1557]	Amplitud Máxima	[-0,6172 0,981]

Fuente: Apolo K, Coba N, 2016.

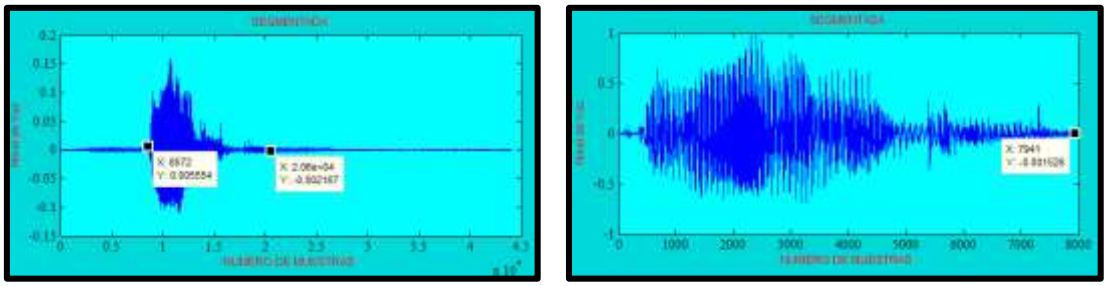


Figura 33-3 Señal original (para) y señal Silencios en función de numero de muestras

Fuente: Apolo K, Coba N, 2016.

Tabla 26-3 Comparación entre la Señal Original y Silencios

	NÚMERO TOTAL DE MUESTRAS
SEÑAL ORIGINAL	20600-8872 =12028
SEÑAL SIN SILENCIOS	7941
TOTAL DE MUESTRAS ELIMINADAS	**4087**

Fuente: Apolo K, Coba N, 2016.

- **Preénfasis**

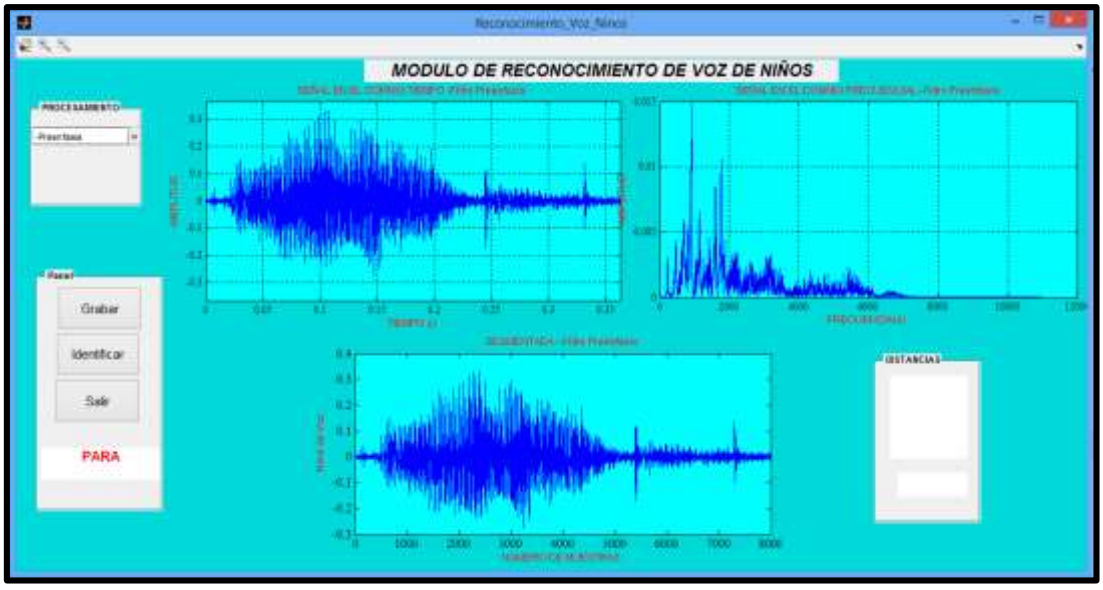


Figura 34-3 Interfaz Gráfica Preénfasis

Fuente: Apolo K, Coba N, 2016.

En estas Figura 34-3 se muestran las señales en función al tiempo, número de muestras y frecuencia correspondientes a la señal eliminada Silencios con el filtro preénfasis y se puede denotar que no existe una acentuación de frecuencias debido a que este comando de voz no tiene sonidos sordo pero se puede denotar que mantiene una relación uniforme de la señal de voz.

- **Análisis de LPC**

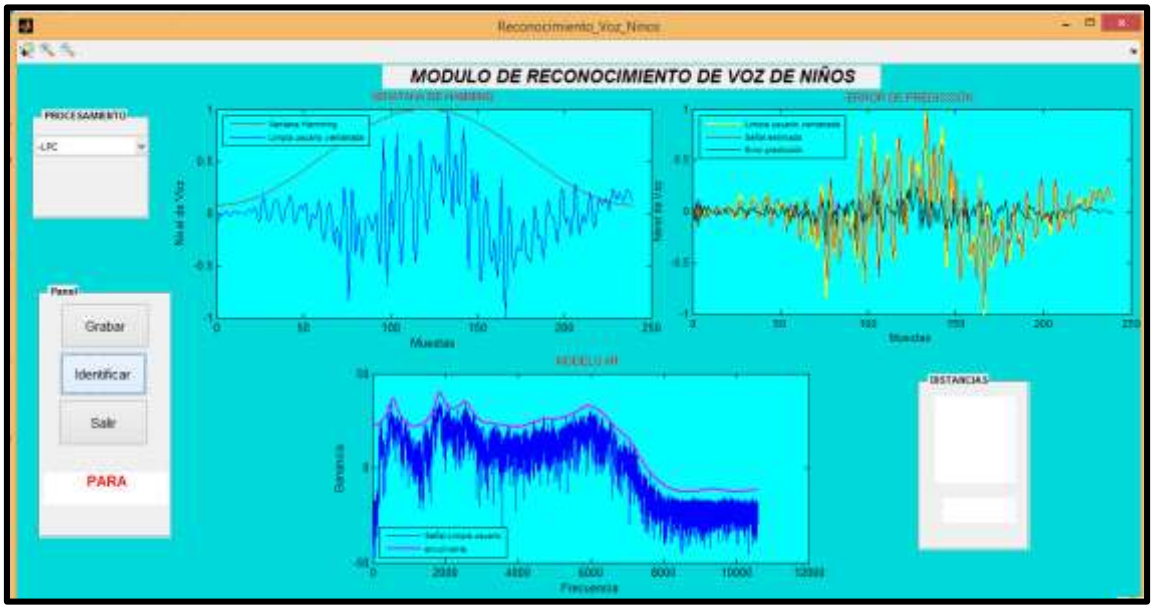


Figura 35-3 Interfaz Gráfica LPC (para)

Fuente: Apolo K, Coba N, 2016.

- **Distancia**

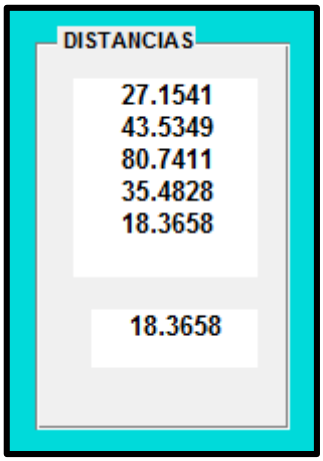


Figura 36-3 Distancias Totales

Fuente: Apolo K, Coba N, 2016.

- **Formantes de los comandos de voz**

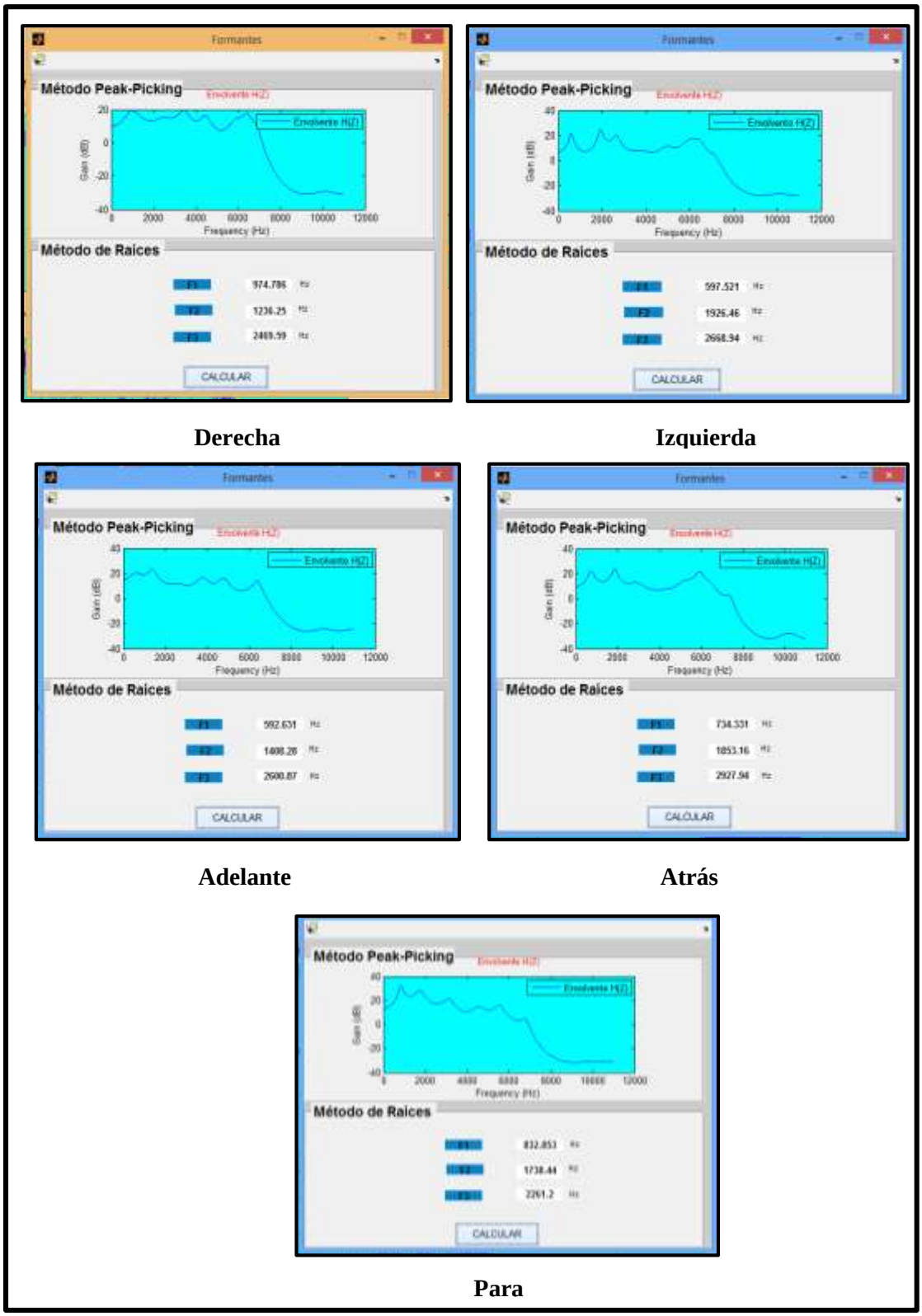


Figura 37-3 Envolvente de las señales (Formantes)

Fuente: Apolo K, Coba N, 2016.

3.5 Sistematización

¿Cuáles son las aplicaciones del Procesamiento de la Señal Digital?

Dentro del Procesamiento de la Señal Digital se pueden distinguir varias aplicaciones entes ellas se tiene:

PROCESAMIENTO SEÑAL DIGITAL	APLICACIÓN
Voz	• Síntesis y reconocimiento • Reconocimiento de interlocutor • Compresión.
Imagen	• Compresión/Transmisión • Reconocimiento • Realidad virtual
Telecomunicaciones	• Módems • Cancelación de ecos • Multiplexación de canales • Ecualización de canales • Criptografía.
Consumo	• Juguetes • TV y audio digitales • Cámaras.
Aplicaciones militares	• RADAR
Electromedicina	• Diagnóstico automático • Sistemas de obtención y tratamiento de imágenes médicas • Prótesis.

¿Qué software se puede utilizar para el Procesamiento de la Señal de Voz?

Existen diversos software para Procesamiento de la Señal de Voz entre los que se destacan Matlab, LabVIEW, SciPy.

¿Qué problemas presentan los módulos DSP existentes en el mercado?

Son muy susceptibles al ruido es decir que en un ambiente con un alto margen de ruido su efectividad reduce considerablemente, además de ser costosos.

¿Qué problemas implica procesar una señal y como se puede solucionar?

Los problemas que se pueden presentar en el caso de las señales analógicas y digitales para el procesamiento de la señal pueden distinguir las siguientes:

El procesamiento de la señal tanto analógica como digital implica una amplia gama de proceso matemático para ello existen software de alto nivel que permiten resolver las ecuaciones matemáticas de una manera más fácil e incluso obtener las gráficas correspondientes.

En el muestreo de la señal se puede presentar el fenómeno de aliasing que causa que señales continuas distintas se tornen indistinguibles cuando se realiza el muestreo de la señal, sucede cuando la señal original no puede ser reconstruida de forma unívoca a partir de la señal. La solución para evitar este fenómeno se aplica el Teorema de Nyquist el cual establece que se debe muestrear una señal al menos al doble de la frecuencia máxima de la señal original.

En una señal analógica o digital puede existir ruido que es una señal no deseada que se mezcla con la señal original. Se puede emplear filtros para eliminar estas señales indeseables.

En la adquisición de la señal de voz en de gran importancia que la señal sea adquirida sin ruidos ambientales para ello es importante adquirir un micrófono que permita obtener la señal con el menor ruido ambiental posible.

A pesar de poseer un buen micrófono que capte la señal de voz cabe la posibilidad que exista un margen de ruido ambiental una de la soluciones para eliminar este ruido es obteniendo un valor de umbral de la voz, para ello las señales que estén por debajo de este umbral van a ser considerados como ruido y serán eliminados.

Existen diversos métodos que permiten realizar el reconocimiento de voz basándose en la envolvente de la señal, sin embargo unos proporciona mejores resultados que otros para lo cual es importante considerar estudios previos realizados con los diferentes métodos

CONCLUSIONES

- Previa investigación acerca del procesamiento de la señal digital se logró diseñar e implementar un Módulo de Reconocimiento de Voz para Niños basado en comandos utilizando el software Matlab, el mismo que fue aplicado en un caso práctico y comparado con otro modulo con la finalidad de corroborar su funcionamiento.

- El software Matlab facilita el procesamiento de señales digitales proporcionando una gama extensa de herramientas permitiendo la utilización de la interfaz gráfica, la cual mejora el entorno de trabajo del programador y así también proporciona al usuario final una interfaz amigable que le permite interactuar con el programa de la manera más fácil posible.

- En este tipo de sistema de Reconocimiento de voz se presentan algunos inconvenientes como el ruido, distancia del micrófono y la similitud que existe entre algunas palabras lo cual dificultan el correcto funcionamiento del reconocimiento.

- En el proceso de eliminación de silencios la determinación del umbral es pieza clave debido a que se si se elige un umbral muy alto se eliminarían ciertos fonemas y al momento de comparar resultara una confusión entre palabras que se pronuncian de igual manera.

- El procesamiento de la señal de voz tanto en adultos o como niños se procesa de la misma manera pero se debe considerar que para el reconocimiento de comandos de voz uno de los aspectos importante es el umbral ya que en los niños este valor es relativo bajo en comparación a la de un adulto.

- La acentuación de frecuencias altas juega un papel importante en el reconocimiento debido a que comprime la señal de voz de manera que no se sature en amplitud y además acentúa frecuencias altas correspondientes a sonidos sordos que pueden ser eliminadas en el proceso de extracción de características.

- El uso de la Codificación Predictiva Lineal (LPC) permitió obtener las características del tracto vocal que corresponde a los coeficientes LPC los cuales son de gran importancia para el reconocimiento ya que estos coeficientes contienen información específica de cada comando y a su vez permitieron obtener la envolvente ,los formantes de la señal y determinar

el cálculo del margen de Error de Predicción el cual fue mínimo logrado así una mejor eficiencia del Módulo de Reconocimiento de voz basado en comandos.

- Es importante considerar que este programa de Reconocimiento de Voz basado en comandos de voz es independiente del estado físico y emocional del usuario debido a que se utiliza una base de datos para la obtención de una señal patrón, por lo tanto si el usuario graba con un determinado tono e intenta acceder al sistema este no podrá reconocer.

RECOMENDACIONES

- En la adquisición de la señal de voz realizarlo con un micrófono diadema que sea directivo ya que este permite captar la señal de voz proveniente del usuario en una sola dirección y al ser diadema permite una mayor comodidad al usuario.

- En la técnica Codificación Predictiva Lineal (LPC) utilizada para la extracción de características se debe elegir el número de coeficientes de acuerdo a la fórmula tomando en consideración que se pueden o no adicionar 3 o 4 coeficientes, para de esta manera minimizar el error entre la señal a reconocer y las señales patrón.

- Considerar un número de grabaciones para la Base de Datos similar al propuesto para de esta manera obtener una señal base que contenga todas las posibles variaciones del habla.

- Elegir un método de extracción de características y comparación más eficaz para que el error del reconocedor sea el más bajo posible.

- Investigar los métodos actuales utilizados en el reconocimiento de voz en los cuales las limitaciones sean mínimas.

- Utilizar el Módulo de Reconocimiento implementado en otro caso práctico con la finalidad de darle una mejor utilidad.

BIBLIOGRAFÍA

1. **ALONSO,** A. *La Voz Humana*. 4ª ed. Madrid-España : Visión Libros, 2011, pp. 15-30.

2. **ARDUINO**. [Consulta: 10 de agosto del 2016]. Disponible en: http://arduinodhtics.weebly.com/iquestqueacute-es.html

3. **ARDUINO**. [Consulta: 10 de agosto del 2016]. Disponible en: https://www.arduino.cc/en/Guide/Introduction

4. **ARDUINO**. [Consulta: 10 de agosto del 2016]. Disponible en: Arduino,2016,http://arduino.cl/intel-galileo/

5. **ARGANDOÑA MARTINEZ,** Felipe Daniel**.** *Implementación de un codificador de voz CELP mejorado para canales de banda angosta* [en línea] (tesis) (Maestría) .Universidad Nacional de Ingeniería, Facultad de Ingeniería Eléctrica y Electrónica. Lima-Perú. 2008. pp. 27. Disponible en: http://cybertesis.uni.edu.pe/handle/uni/156

6. **COBETA,** Marco; et al. *Patología de la voz [*en línea]. Barcelona-España : Marge Médica Books, 2013. pp. 107-109. [Consulta: 28 julio del 2016]. Disponible en: https://books.google.com.ec/books?id=OdFUAQAAQBAJ&printsec=frontcover&dq=patologia+de+la+voz+pdf&hl=es&sa=X&ved=0ahUKEwil24i7u6LQAhVh_IMKHa_uBAgQ6AEIGTAA#v=onepage&q&f=false

7. **DIYMakers.** [Consulta: 11 de agosto del 2016]. Disponible en: http://diymakers.es/intel-galileo-guia-inicial/

8. **DIYMakers**. *Arduino+Bluetooth*. [Consulta: 12 de agosto del 2016]. Disponible en: http://diymakers.es/arduino-bluetooth/

9. **DUQUE SANCHEZ, Christian, & MORALES PEREZ, Mauricio**. *Caracterización de voz empleando análisis tiempo-frecuencia aplicada al reconocimiento de emociones [*en línea] (tesis). (Ingeniería) Universidad Tecnológica de Pereira, Facultad de Ingenierías Eléctrica, Electrónica, Física y Ciencias de la Computación. Pereira-Colombia. 2007. pp. 06-07. [Consulta: 15 julio del 2016]. Disponible en: http://repositorio.utp.edu.co/dspace/handle/11059/89

10. **ELECTRONICLAB**. [Consulta: 21 de agosto del 2016]. Disponible en: http://electronilab.co/tienda/modulo-lm2596-convertidor-de-voltaje-dc-dc-buck-1-25v-35v/

11. **Erle Robotics.** [Consulta: 20 de agosto del 2016]. Disponible en: https://erlerobotics.gitbooks.io/erle-robotics-erle-copter/content/es/safety/lipo.html

12. **JACKSON,** M.C. *La Voz Normal* [en línea]. Buenos Aires -Argentina : Ed. Médica Panamericana, 1992. pp. 54. [Consulta: 15 mayo 2016]. Disponible en: https://books.google.com.ec/books?id=dKfW0DW8tIEC&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

13. **MATHWORKS**. *MATLAB* [en línea]. [Consulta: 8 de junio del 2016]. Disponible en https://www.mathworks.com/products/matlab/features.html

14. **MATHWORKS.** [Consulta: 3 de octubre del 2016]. Disponible en: http://www.mathworks.com/matlabcentral/fileexchange/32374-legacy-matlab-and-simulink-support-for-arduino

15. **MÉNDEZ AGAMA, Vinicio Daniel, & RAMIREZ DE LA CRUZ, Daniel Oswaldo**. *Diseño e implementación del sistema de movimiento direccional de una silla de ruedas para ser controlada por reconocimiento de patrón de voz mediante electrónica de potencia y motores DC como actuadores [*en línea] (tesis). (Ingeniería) Universidad Politécnica Salesiana Sede Quito, Carrera de Ingeniería Electrónica. Quito-Ecuador. 2015. pp. 73-75. [Consulta: 15 noviembre 2016]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/9148

16. **MORAL,** Daniel; et al. *Procesado digital de voz para el reconocimiento del hablante aplicado a dispositivos móviles [*en línea] (tesis). (Ingeniería) Escuela Técnica Superior de Ingenieros Industriales y de Telecomunicación, Ingeniería Técnica de Telecomunicación, Especialidad en Sonido e Imagen. Pamplona-España. 2011. pp. 26-28. [Consulta: 15 julio del 2016]. Disponible en: http://academicae.unavarra.es/xmlui/bitstream/handle/2454/9025/629090.pdf?sequence=1&isAllowed=y

17. **Naylamp mechatronics.** [Consulta: 12 de agosto del 2016]. Disponible en: http://www.naylampmechatronics.com/

18. **NATIONAL INSTRUMENTS**. *LabVIEW* [en línea]. [Consulta: 8 de junio del 2016]. Disponible en http://www.ni.com/labview/esa/

19. **Niple.** *Comunicación inalámbrica con el transceptor nrf24l01*. [Consulta: 13 de agosto del 2016]. Disponible en: http://www.niplesoft.net/blog/2016/01/13/comunicacion-inalambrica-con-el-transceptor-nrf24l01/

20. **OpenHardware.** *Arduino, Sensores y comunicaciones, tutoriales.* [Consulta: 13 de agosto del 2016]. Disponible en: http://openhardware.pe/transceptores-nrf24l01-2-4ghz-radio-wireless-how-to/

21. **OROPEZA, José L.; SUAREZ Sergio. "**Algoritmos y Métodos para el Reconocimiento de Voz en Español Mediante Sílabas". *Computación y Sistemas* [en línea], 2006, (México) 9(3), pp. 270-286. [Consulta: 2 agosto 2016]. ISSN 1405-5546. Disponible en: http://www.ejournal.unam.mx/cys/vol09-03/CYS09307.pdf

22. **POHLMANN,** K, C**.** *Principios de audio digital.* 4ª ed. Aravaca, Madrid-España : McGraw-Hill, 2002, pp. 1-2.

23. **RUEDA ROJO, LETICIA.** *Mejoras en reconocimiento del habla basadas en mejoras en la parametrización de la voz* [En línea] (tesis). (Ingeniería) Universidad Autónoma de Madrid, Escuela Politécnica Superior, Ingeniería de Telecomunicación. Madrid-España. 2011. pp. 46-48. [Consulta: 4 julio 2016]. Disponible en: https://repositorio.uam.es/handle/10486/6734

24. **SciPy.org**. *SciPy* [en línea]. [Consulta: 9 de junio del 2016]. Disponible en https://www.scipy.org/about.html

25. **SORIA,** Emilio; et al**.** *Tratamiento digital de señales*. Madrid-España : Pearson Education, S.A., 2003, pp. 2-4.

26. **TecmiKro.** [Consulta: 11 de agosto del 2016]. Disponible en: http://tecmikro.com/modulos-para-arduino-pic-avr/329-l298n-modulo-driver paramotores.html

27. **TORRENTE,** Ó., *ARDUINO Curso práctico de formación [en línea]*. Mexico-Mexico: Alfaomega, S.A. [Consulta: 10 de agosto del 2016]. Disponible en: http://ecoinformatica.cl/wp-content/uploads/2016/03/Arduino-Curso-Pr%C3%A1ctico.pdf

28. **Tektronix.** "Raspberry Pi". "eTech". [en línea], 2012, (EE.UU) (11), pp. 7-8. [Consulta: 10 de agosto del 2016]. ISSN 1002091. Disponible en: http://etech.designspark.info/ELE_0050_eTech%2010/ELE_0050_eTech_ES/pubData/source/ELE_0050_eTech_ES.pdf

29. **VELÁSQUEZ RAMÍREZ, Genoveva.** *Sistema de reconocimiento de voz en Matlab* [En línea] (tesis). (Ingeniería) Universidad de San Carlos de Guatemala, Facultad de Ingeniería, Escuela de Ingeniería Mecánica Eléctrica. San Carlos-Guatemala. 2008. pp. 34-46. [Consulta: 28 junio 2016]. Disponible en: http://biblioteca.usac.edu.gt/tesis/08/08_0223_EO.pdf

30. **VILLALOBOS PONCE, José Alexis, POCEROS MARTINEZ, Fernando, & PEREZ BADILLO, Eyra Oxana.** *Sistema de seguridad por reconocimiento de voz* [En línea] (tesis). (Ingeniería) Instituto Politécnico Nacional, Escuela Superior de Ingeniería Mecánica y Eléctrica, Ingeniería en Comunicaciones y Electrónica. D.F-México. 2013. pp. 8-10. [Consulta: 17 mayo 2016]. Disponible en: http://tesis.ipn.mx:8080/xmlui/handle/123456789/12309

31. **VIVANCO JIMENEZ, Wilmer Enrique, & OCHOA CORONEL, Wilmer Fernando.** *Diseño y estudio de factibilidad para la implementación de un laboratorio de procesamiento de señales y simulación para la Universidad Politécnica Salesiana sede Cuenca* [En línea] (tesis). (Ingeniería) Universidad Politécnica Salesiana, Facultad de Ingenierías, Carrera de Ingeniería Electrónica. Cuenca-Ecuador. 2010. pp. 30--32. [Consulta: 4 julio 2016]. Disponible en: http://dspace.ups.edu.ec/handle/123456789/2206

ANEXOS

ANEXO A: CATEGORÍAS DE LOS FONEMAS SEGÚN EL MODO DE ARTICULACIÓN.

Rasgo		Órganos		Ejemplos
Vocálicas		Las cuerdas vocales vibran al paso del aire sin oclusión completa del tracto vocal en ningún punto		/a/, /e/, /i/, /o/, /u/
Consonánticas	Oclusivas	El aire se retiene y se expulsa de golpe. Se producen por el cierre momentáneo total o parcial del tracto vocal seguido de una liberación más o menos abrupta del aire retenido. Por ejemplo las totales /p/, /t/, /k/ o las parciales /b/, /d/, /g/. Estas últimas son sonoras		/p/, /b/, /t/, /d/, /k/, /g/,
	Fricativas	El aire sale lentamente a través de una pequeña abertura de la boca. Se caracterizan por ser ruidos aleatorios generados por la turbulencia que produce el flujo de aire al pasar por un estrechamiento del tracto. Pueden ser sonoros como /y/ si hay componente glótica, o sordos como /f/, /s/ o /j/ (también /z/ en otras versiones del español)		/f/, /z/, /j/, /s/
	Africadas	El aire se retiene y después se expulsa a través de una pequeña abertura. Si los fonemas comienzan como oclusivos y la liberación del aire es fricativa, se denominan africados. La oclusión y la constricción se producen en el mismo punto de articulación		/ch/
	Líquidas	Vibrantes	La lengua obstaculiza parcialmente el canal. El aire sale por los lados de la boca. Son producidos al pasar el aire por la punta de la lengua y producir su vibración. Tienen componente glótica	/r/, /rr/
		Laterales	La lengua estrecha el canal al rozar con el paladar y produce una o más vibraciones. Se producen cuando se hace pasar la señal sonora glótica por los costados de la lengua	/l/, /ll/

Sonidos Sonoros	Sonios Sordos (no sonoros)
A,e,i,o.u.b.d.g.l.ll.m.n.ñ,r,rr,v,w,y	Ch,f,h,j,k,p,s,t,z

ANEXO B: COMPARACIÓN MATLAB, LAVIEW, Y SciPy.

La comparación se realiza utilizando ponderación numérica de acuerdo al punto de vista de las autoras de este trabajo. La ponderación se da de acuerdo a los siguientes indicativos:

1. Bajo
2. Medio
3. Alto
4. Muy alto

ASPECTO	MATLAB	LABVIEW	SciPy
Conocimiento del lenguaje	*3*	0	0
Funcionalidades básicas	4	4	3
Funcionalidades avanzadas	3	4	2
Gráficos e imágenes	4	4	3
Potencia del lenguaje de programación	4	3	3
Fiabilidad	4	4	3
Rapidez	3	4	3
Información	4	4	3
Facilidad de manejo	4	4	4
Licencia y facilidad de obtención	3	2	4
Instalación	4	4	4
Compatibilidad con otros lenguajes	4	4	2
	44	**41**	**34**

ANEXO C: CARACTERÍSTICAS DE TARJETAS DE ADQUISICIÓN

TIPO DE TARJETA	ARDUINO UNO	ARDUINO MEGA 2560	ARDUINO LEONARDO
Microcontrolador	ATmega328	ATmega2560	ATmega32u4
Entradas / Salidas digitales	14 pines (6 con PWM)	54 (14 con PWM)	20 (Canales PWM 7)
Entradas analógicas.	6 pines	16	12
Voltaje de funcionamiento	5V	5V	5V
Tensión de alimentación (recomendado)	7-12V	7-12V	7-12V
Tensión de alimentación máxima	20V (no recomendado)	6-20V	
Memoria Flash	32 KB (ATmega328) de los cuales 0,5 KB utilizados por bootloader	256 KB	32 KB (4 KB usados para el bootloader)
SRAM	2 KB (ATmega328)	8 KB	2.5 KB
EEPROM	1 KB (ATmega328)	4 KB	1 KB
Velocidad de reloj	16 MHz	16 MHz	16 MHz
Alimentación	USB o batería externa.	USB o batería externa.	USB o batería externa.

Fuente: http://www.electronicaembajadores.com/Admin/Content/eyontzqw.pdf

CARACTERÍSTICAS DE TARJETA GALILEO	
Voltaje de Entrada (recomendado)	5V
Voltaje de Entrada (límites)	5V
Pines I/O Entrada/Salida Digitales	14 (de los cuales 6 poseen salida PWM)
Pines de Entrada Analógicos	6
Corriente Total DC salida en todas las líneas I/O (Entrada/Salida)	80 mA
Corriente DC para 3.3V Pin	800 Ma
Corriente DC para 5V Pin	800 mA

Fuente: http://arduino.cl/intel-galileo/

CARACTERÍSTICAS DE RASPBERRY PI	
Fabricante	Fundación Raspberry Pi
Tipo	Placa computadora (SBC)
Sistema operativo	Linux ARM (Debian,Fedora, Arch Linux),RISC OS
Alimentación	2,5 W (modelo A) 3,5 W (modelo B)
CPU	ARM1176JZF-S (armv6k) a 700 MHz
GPU	Broadcom VideoCore IV
Memoria	256 MiB (Modelo A y primeros modelos B) 512 MiB (modelo B)
Capacidad de almacenamiento	Tarjeta SD o SDHC

Fuente: http://www.raspiman.com/que-es-una-raspberry-pi/

ANEXO D: ESPECIFICACIONES TECNICAS STEREO HEADSET MODELO 662862.

Fuente:http://www.omegatechnology.com.sv/omegasv00/advanced_search_result.php?keywords=+662862&x=0&y=0

ANEXO E: ENLACE MATLAB-ARDUINO

- Crear una cuenta en MathWorks de Matlab.
- Una vez creada la cuenta seleccionar la opción Community, File Exchange.

Figura Página MathWorks

Fuente: (http://www.mathworks.com/matlabcentral/?s_tid=gn_mlc)

- Buscar el paquete ArduinoIO que se encuentra con el nombre Legacy MATLAB and Simulink Support for Arduino

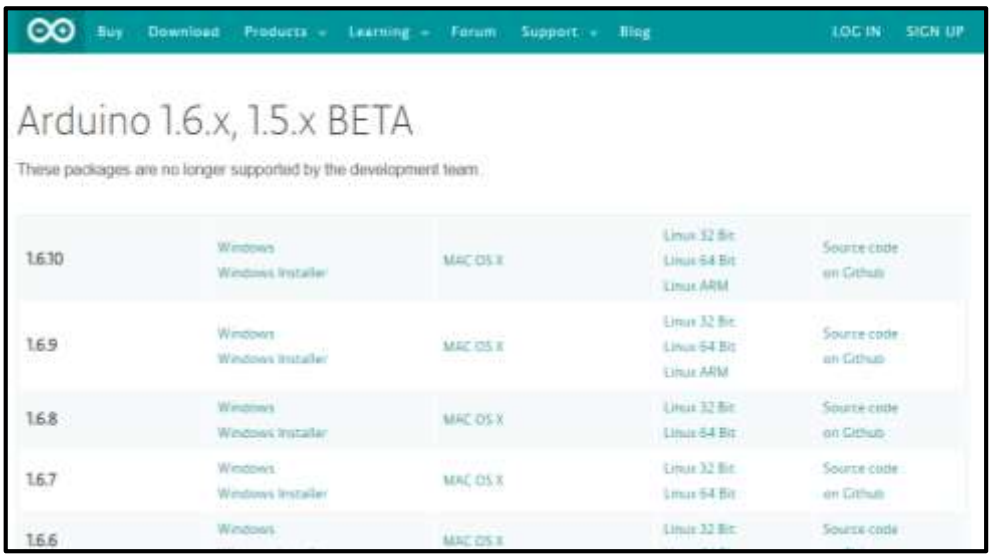
Fuente :(http://www.mathworks.com/matlabcentral/fileexchange/?utf8=%E2%9C%93&term=Legacy+MATLAB+nd+Simulink+Support+for+Arduino)

- Descargar el paquete ArduinoIO (Legacy MATLAB and Simulink Support for Arduino)
- El paquete ArduinoIO proporciona 5 Sketch que son programas hechos con el software Arduino.

Tabla: Características de Sketch ArduinoIO

SKETCH DE ARDUINO	CARACTERÍSTICAS
adio.pde	Entradas/Salidas Analógicas y Digitales Comandos básicos de comunicación serial.
adioe.pde	adio.pde + soporte de encoders
adioes.pde	adioe.pde + soporte de servo motores
motor_v1.pde	adioes.pde + afmotor v1 shield
motor_v2.pde	adioes.pde + afmotor v2 shield

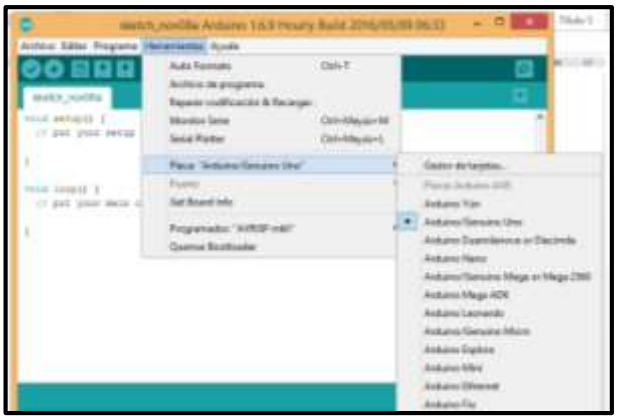
- Descargar Arduino Software (IDE) versión 1.9.6 para Windows que se encuentra disponible en la página oficial de Arduino.



Fuente: (https://www.arduino.cc/en/Main/OldSoftwareReleases#previous)

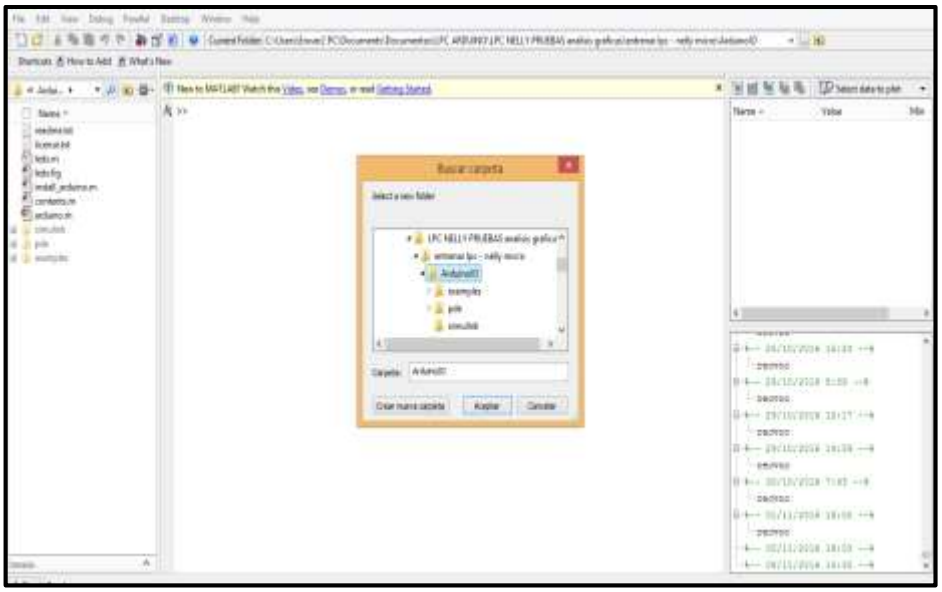
- Abrir la carpeta arduino-1.9.6 y luego arduino
- Conectar la placa Arduino a través del cable USB e instalar los Driver para que tener un puerto para la placa Arduino.
- Ir al Panel de Control, clic en Administrador de Dispositivos
- Ver un icono de aviso sobre la placa Arduino.
- Click con el botón derecho sobre el nombre del dispositivo Arduino y seleccionar la opción "Actualizar software de controlador"
- Se tiene dos opciones. Por un lado Windows nos ofrece buscar el controlador de manera automática y la segunda opción indicar manualmente la ruta del mismo. Seleccionar la segunda opción.
- Los drivers vienen incluidos en la carpeta arduino-1.9.6 que contiene el software, colocar la ruta donde se encuentra **drivers.**

- Aparecerá un cuadro de diálogo. En él se indica que se a instalar un software del que Windows no puede verificar el editor.
- Click en “Instalar este software de controlador de todas formas”.
- Después de un momento aparecerá una ventana que indica que ha sido instalado correctamente.

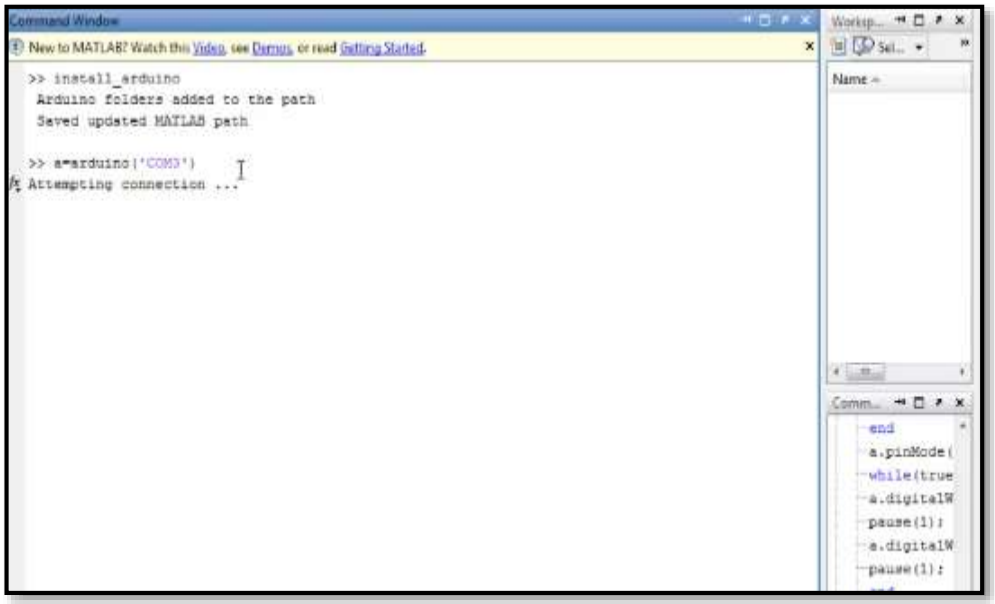


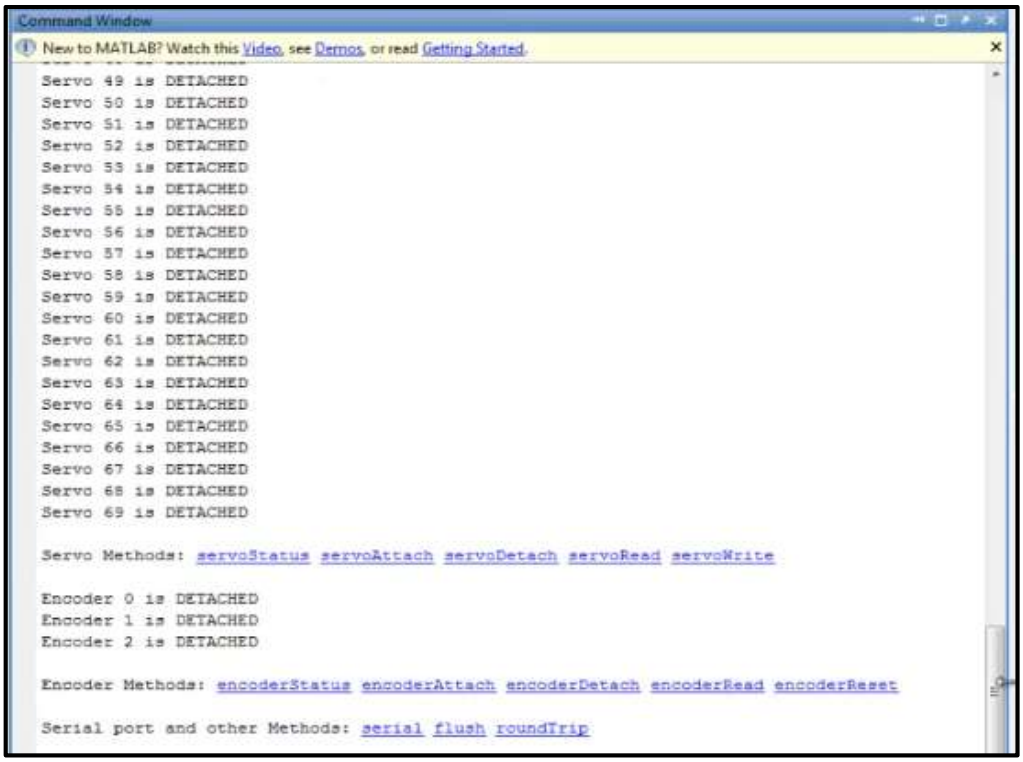
- Seleccionar el puerto COM y placa correspondiente.
- Compilar y subir el Sketch adioes.pde a la placa Arduino para que pueda entender las ordenes enviadas desde el MATLAB. Cerrar el software “ARDUINO” para no tener el puerto ocupado y así poder establecer una conexión a través del MATLAB.

- Ejecutar Matlab como Administrador y colocar la carpeta ArduinoIO en el directorio “Current Folder” de Matlab.



- Insertar install_arduino
- Para posterior ejecutar el comando a=arduino ('COMX') donde X es el número de puerto de conexión Matlab y Arduino y una vez ejecutado el comando aparecerá una serie de instrucciones la cual indica que se pode establecer la conexión Matlab – Arduino.

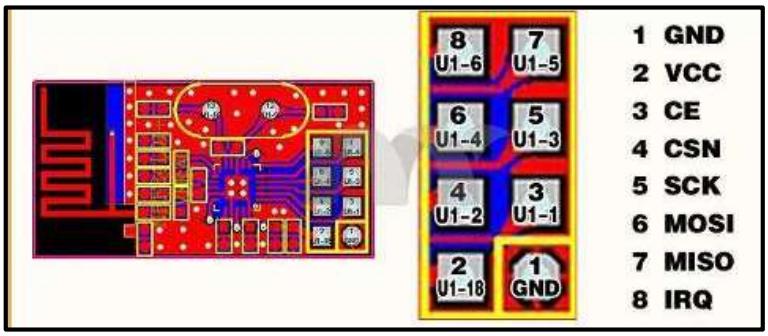



Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Servo 49 is DETACHED
Servo 50 is DETACHED
Servo 51 is DETACHED
Servo 52 is DETACHED
Servo 53 is DETACHED
Servo 54 is DETACHED
Servo 55 is DETACHED
Servo 56 is DETACHED
Servo 57 is DETACHED
Servo 58 is DETACHED
Servo 59 is DETACHED
Servo 60 is DETACHED
Servo 61 is DETACHED
Servo 62 is DETACHED
Servo 63 is DETACHED
Servo 64 is DETACHED
Servo 65 is DETACHED
Servo 66 is DETACHED
Servo 67 is DETACHED
Servo 68 is DETACHED
Servo 69 is DETACHED
Servo Methods: servoStatus servoAttach servoDetach servoRead servoWrite
Encoder 0 is DETACHED
Encoder 1 is DETACHED
Encoder 2 is DETACHED
Encoder Methods: encoderStatus encoderAttach encoderDetach encoderRead encoderReset
Serial port and other Methods: serial flush roundTrip

ANEXO F: CONEXIÓN DE MÓDULO NRF24L01

Interface SPI: VCC, GND, CE, CSN, SCK, MISO, MOSI y IRQ opcional.

Fuente:http://www.prometec.net/nrf2401/



Fuente:http://mco-s1-p.mlstatic.com/modulo-transceptor-rf-nrf24l01-arduino-pic-avr-24-ghz-1371-MCO3746363892_012013-O.jpg

PIN	NRFL2401	ARDUINO UNO
GND	1	GND
VCC	2	3.3 voltios
CE	3	9
CSN	4	10
SCK	5	13
MOSI	6	11
MISO	7	12
IRQ	8	-

Descargar las librerías:

	LIBRERÍAS
Módulo NRF24L01	RF24.h
LCD	Wire.h LCD.h
Bus I2C	LiquidCrystal_I2C.h

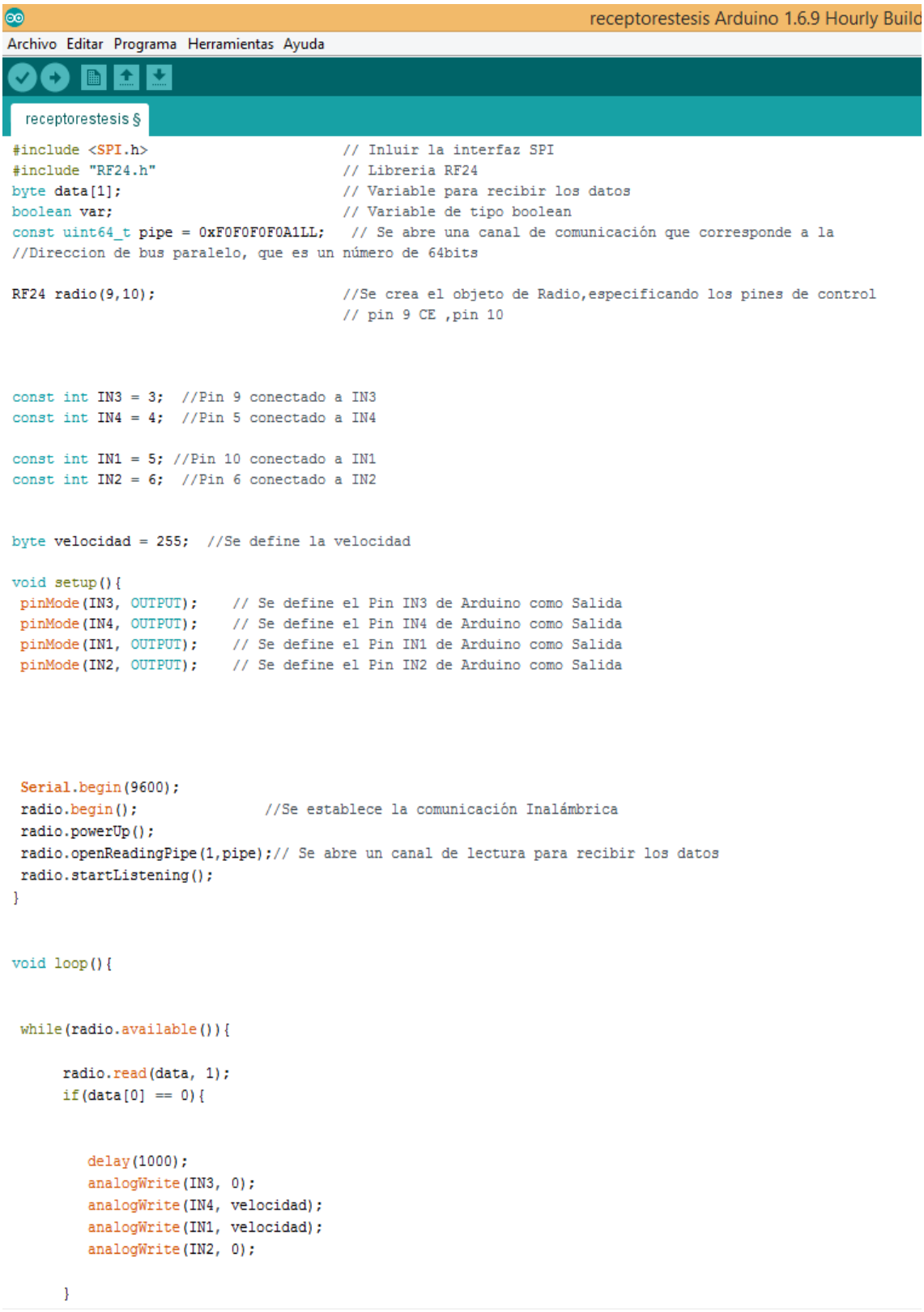
ANEXO G: SKETCH TRANSMISOR


```
#include  <SPI.h>                          // Libreria SPI
#include  "RF24.h"                         // Libreria RF24
#include <Wire.h>                          // Incluir la librería I2c que en Arduino se llama Wire
#include <LCD.h>                           // Libreria para el LCD



#include <LiquidCrystal_I2C.h>
byte data[1];                               // Variable para enviar los datos

const uint64_t pipe = 0xF0F0F0F0A1LL;       // Se abre una canal de comunicación que corresponde a la
//Direccion de bus paralelo, que es un número de 64bits



//Se crea el objeto de Radio,especificando los pines de control
RF24 radio(9, 10);                    // pin 9 CE ,pin 10

LiquidCrystal_I2C             lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);



//se define la dirección del LCD qu en este caso es 0x3F
//se crea la instancia del objeto LiquidCrystal_I2C con los pines que se utilizan
```

```
lcd.print("Modulo_Rec_Voz");   // Imprime en la LCD el mensaje
lcd.setCursor ( 0, 1 );        // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:1)

lcd.print("   IZQUIERDA  "); //Imprime el mensaje
lcd.setCursor ( 0, 2 );       // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:2)


}

// se leen la entradas de arduino y se establece una condición
if (digitalRead(3) == LOW && digitalRead(4) == LOW && digitalRead(5) == HIGH &&
    digitalRead(6) == LOW && digitalRead(7) == LOW) {



pinMode(7, INPUT_PULLUP);           // Se define el Pin 7 de Arduino como Entrada


radio.begin();                      // Se establece la comunicación Inalámbrica


 radio.powerUp();
 radio.openWritingPipe(pipe);       // Se abre un canal de escritura para enviar los datos
 radio.stopListening();
 Serial.begin(9600);                //Se inicializa el puerto serie




}

void loop() {
  //se leen la entradas de arduino y se establece una condición
  if (digitalRead(3) == HIGH && digitalRead(4) == LOW && digitalRead(5) == LOW &&
      digitalRead(6) == LOW && digitalRead(7) == LOW) {


//si cumple la condicion se define una variable y se almacena un valor
data[0] = 0;
lcd.clear();                  //Borra la pantalla LCD
                              //y posiciona el cursor en la esquina superior izquierda(posición (0,0))


lcd.print("Modulo_Rec_Voz");  //Imprime en la LCD el mensaje
lcd.setCursor ( 0, 1 );       //Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:1)
lcd.print("    DERECHA   "); //Imprime el mensaje
lcd.setCursor ( 0, 2 );       // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:2)


}

// se leen la entradas de arduino y se establece una condición
if (digitalRead(3) == LOW  && digitalRead(4) == HIGH  && digitalRead(5) == LOW  &&
    digitalRead(6) == LOW && digitalRead(7) == LOW) {



data[0] = 1;                  //Si cumple la condicion se asigna un valor a una variable
lcd.clear();                  //Borra la pantalla LCD
                              //y posiciona el cursor en la esquina superior izquierda (posición (0,0))
```

```
data[0] = 2;                    //Si cumple la condicion se asigna un valor a una variable
lcd.clear();                    //Borra la pantalla LCD
                                //y posiciona el cursor en la esquina superior izquierda (posición (0,0))
lcd.print("Modulo_Rec_Voz");     // Imprime en la LCD el mensaje


lcd.setCursor ( 0, 1 );         // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:1)
lcd.print("    ADELANTE  ");      //Imprime el mensaje
lcd.setCursor ( 0, 2 );         // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:2)


}

// se leen la entradas de arduino y se establece una condición
if (digitalRead(3) == LOW &&  digitalRead(4) == LOW &&
    digitalRead(5) == LOW  && digitalRead(6) == HIGH && digitalRead(7) == LOW)  {



 data[0] = 3;                    //Si cumple la condicion se asigna un valor a una variable
 lcd.clear();                    //Borra la pantalla LCD
                                 // y posiciona el cursor en la esquina superior izquierda (posición (0,0))
 lcd.print("Modulo_Rec_Voz");    // Imprime en la LCD el mensaje


lcd.setCursor ( 0, 1 );         // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:1)
lcd.print("     ATRAS  ");      //Imprime el mensaje
lcd.setCursor ( 0, 2 );         // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:2)

 }

 // se leen la entradas de arduino y se establece una condición


if (digitalRead(3) == LOW && digitalRead(4) == LOW && digitalRead(5) == LOW &&
    digitalRead(6) == LOW &&  digitalRead(7) == HIGH)  {



 data[0] = 4;                    //Si cumple la condicion se asigna un valor a una varible
 lcd.clear();                    //Borra la pantalla LCD
                                 //y posiciona el cursor en la esquina superior izquierda (posición (0,0))
 lcd.print("Modulo_Rec_Voz");   // Imprime en la LCD el mensaje


lcd.setCursor ( 0, 1 );        // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:1)
lcd.print("     ATRAS ");      //Imprime el mensaje
lcd.setCursor ( 0, 2 );       // Se ubica el cursor en la primera posición(columna:0) de la segunda línea(fila:2)



}

radio.write(data, 1);                //Se envia los datos
```

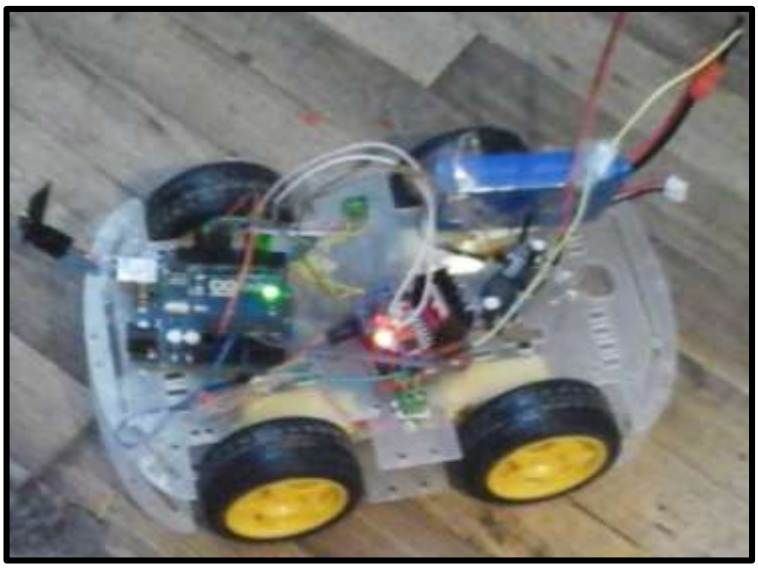
ANEXO H: SKETCH RECEPTOR

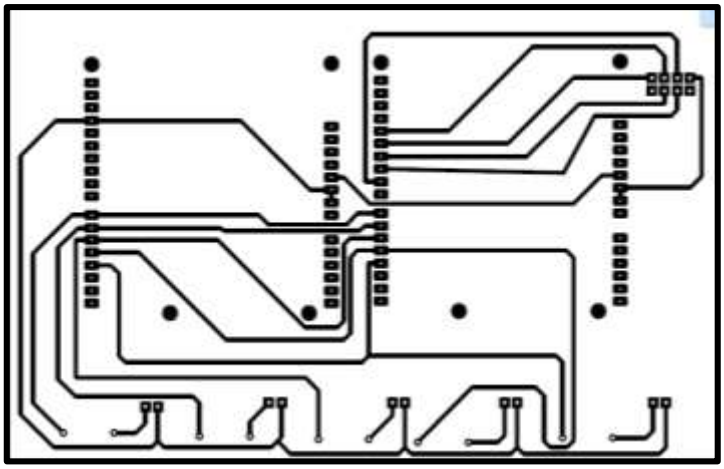


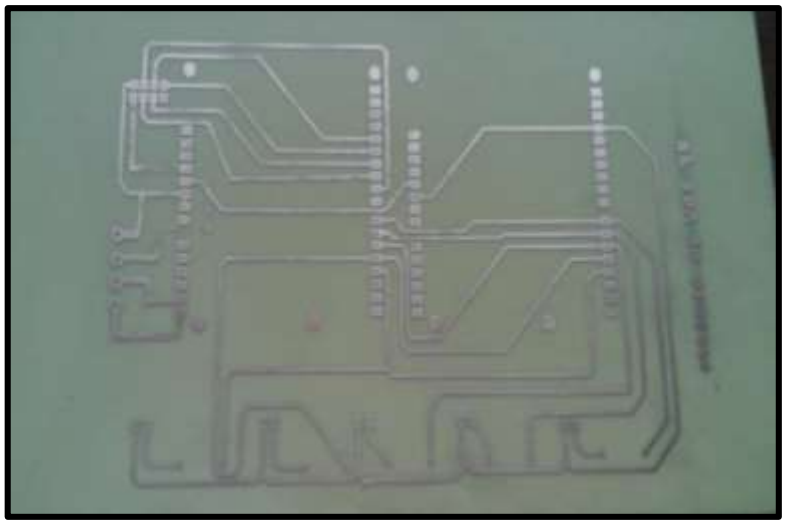

```
#include <SPI.h>                        // Inluir la interfaz SPI
#include "RF24.h"                       // Libreria RF24
byte data[1];                           // Variable para recibir los datos
boolean var;                            // Variable de tipo boolean
const uint64_t pipe = 0xF0F0F0F0A1LL;   // Se abre una canal de comunicación que corresponde a la
//Direccion de bus paralelo, que es un número de 64bits

RF24 radio(9,10);                       //Se crea el objeto de Radio,especificando los pines de control
                                        // pin 9 CE ,pin 10



const int IN3 = 3;  //Pin 9 conectado a IN3
const int IN4 = 4;  //Pin 5 conectado a IN4

const int IN1 = 5; //Pin 10 conectado a IN1
const int IN2 = 6;  //Pin 6 conectado a IN2


byte velocidad = 255;  //Se define la velocidad

void setup(){
 pinMode(IN3, OUTPUT);    // Se define el Pin IN3 de Arduino como Salida
 pinMode(IN4, OUTPUT);    // Se define el Pin IN4 de Arduino como Salida
 pinMode(IN1, OUTPUT);    // Se define el Pin IN1 de Arduino como Salida
 pinMode(IN2, OUTPUT);    // Se define el Pin IN2 de Arduino como Salida




 Serial.begin(9600);
 radio.begin();              //Se establece la comunicación Inalámbrica
 radio.powerUp();
 radio.openReadingPipe(1,pipe);// Se abre un canal de lectura para recibir los datos
 radio.startListening();
}


void loop(){


 while(radio.available()){

      radio.read(data, 1);
      if(data[0] == 0){


         delay(1000);
         analogWrite(IN3, 0);
         analogWrite(IN4, velocidad);
         analogWrite(IN1, velocidad);
         analogWrite(IN2, 0);

      }
```

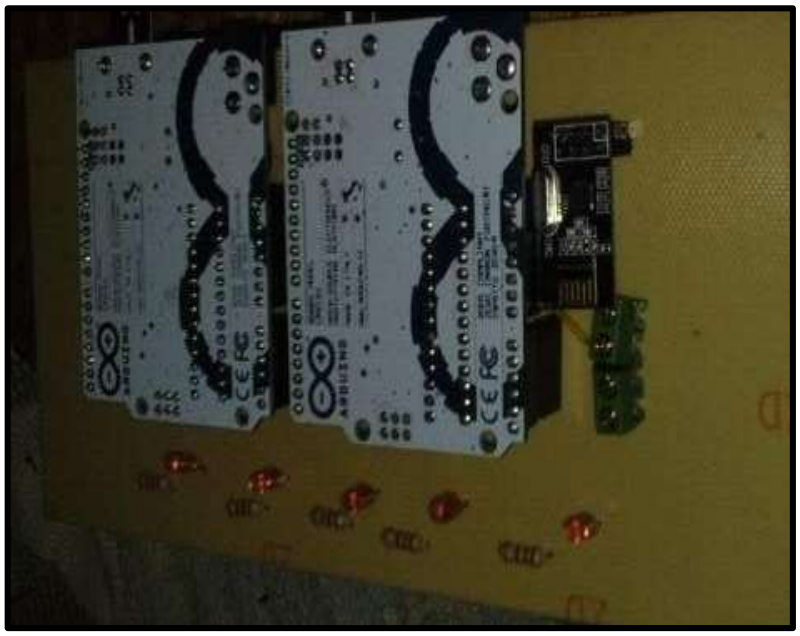
```
  if(data[0] == 1){

        delay(1000);
        analogWrite(IN3, velocidad);
        analogWrite(IN4, 0);
        analogWrite(IN1, 0);
        analogWrite(IN2, velocidad);

  }


  if(data[0] == 2){

        delay(1000);
        analogWrite(IN3, velocidad);
        analogWrite(IN4, 0);
        analogWrite(IN1, velocidad);
        analogWrite(IN2, 0);

  }



     if(data[0] == 3){

              delay(1000);
              analogWrite(IN3, 0);
              analogWrite(IN4, velocidad);
              analogWrite(IN1, 0);
              analogWrite(IN2, velocidad);
   }


if(data[0] == 4){

              delay(1000);
              analogWrite(IN3, 0);
              analogWrite(IN4, 0);
              analogWrite(IN1, 0);
              analogWrite(IN2, 0);

   }


 }
}
```

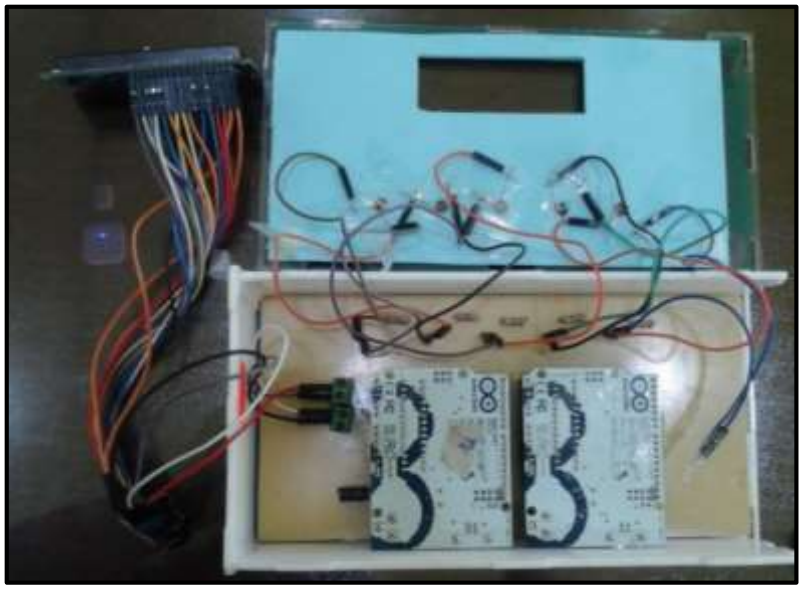
ANEXO I: IMPLEMENTACIÓN CASO PRÁCTICO

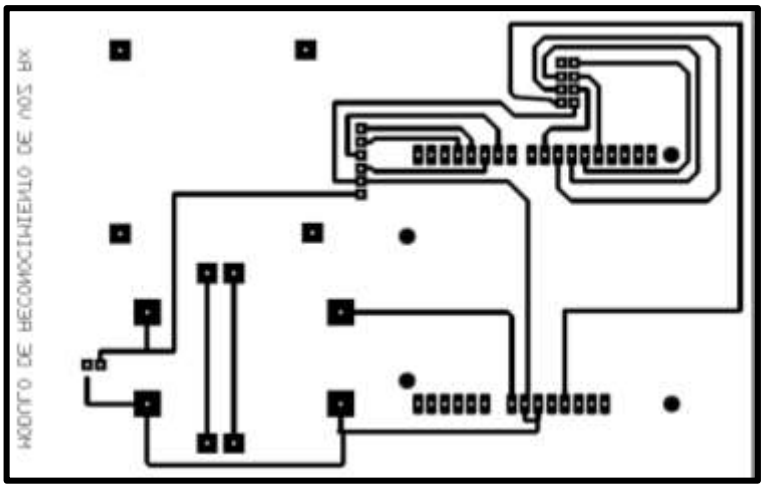


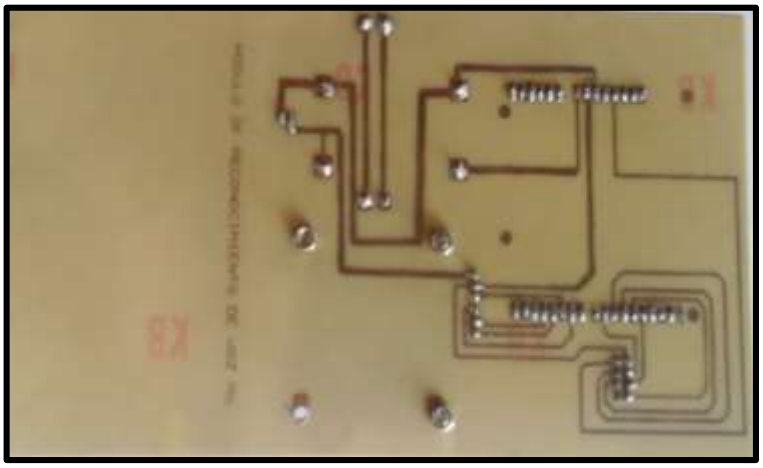




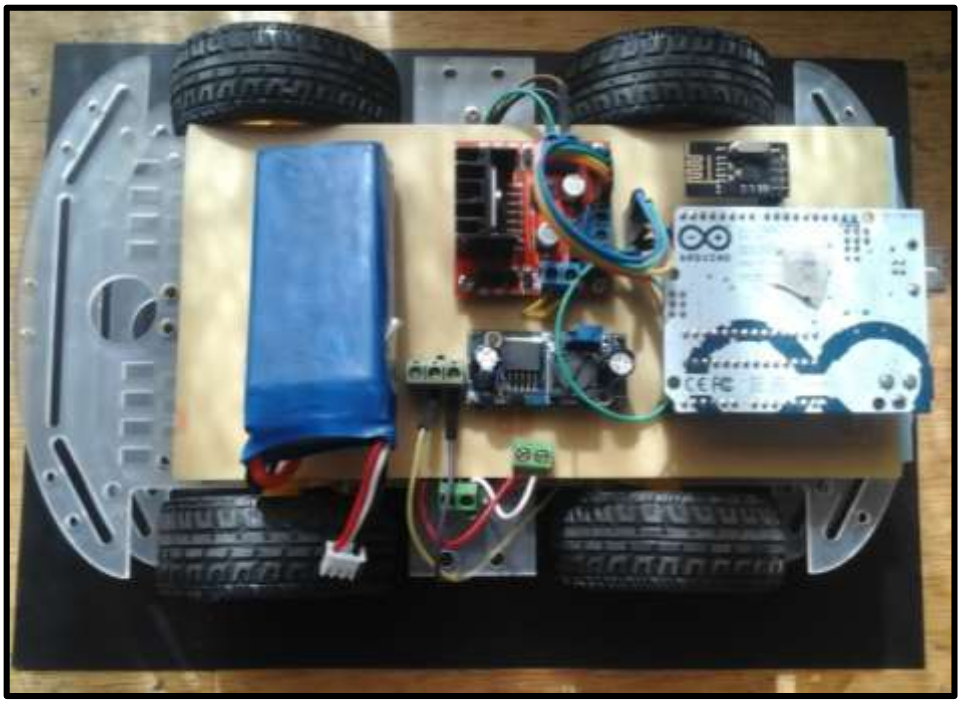





MODULO DE RECONOCIMIENTO DE VOZ RX




ESPOCH
MÓDULO DE
RECONOCIMIENTO DE VOZ
TRANSMISOR
KA_JC



ANEXO J: PRESUPUESTO

Elementos Electrónicos			
Cantidad	**Descripción**	**P.Unitario**	**P.Total**
3	Arduino Uno	20,99	62,97
2	NRF24L01	8,5	17
5	Leds	0,15	0,75
5	Resistencias 330Ω	0,09	0,45
1	Batería LiPo	27,99	27,99
1	Driver L298D	10	10
1	Modulo Conmutador	8	8
1	Chasis carro Robot 3wd	32,99	32,99
1	Bus I2C	6	6
1	LCD 16X2	9	9
2	Baquelitas	6	12
6	Acido(Percloruro Férrico)	1,25	7,5
1	Pasta para soldar	1,25	1,25
1	Chupa Suelda	3	3
1	Cautín	2,5	2,5
1	Marcado Sharpie	3	3
4	Estaño	0,8	3,2
3	Brocas de 1mm	1,25	3,75
1	Metro de cable flexible	0,85	0,85
2	Papel transfer	1,25	2,5
3	Borneras	0,6	1,8
6	Espadines Macho	0,95	5,7
5	Espadines Hembra	1	5
1	Caja de acrílico	15	15
TOTAL			242,2

RECURSOS DE HARDWARE	**TOTAL**
Elementos Electrónicos	242,2
Laptop corei5 HP	850
RECURSOS DE SOFTWARE	0,00
OTROS	60
TOTAL	1155,20

ANEXO K: CÓDIGO MATLAB

```
%##############################BASE DE DATOS
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

Fs=22050; %frecuencia de muestreo
%tiempograb=1.5; %Tiempo de grabacion
s=wavrecord(2*Fs,Fs,1); %función de grabacion
wavwrite(s,Fs,16,'grabacion1.wav')
[s,Fs]=wavread('grabacion1.wav');
sound (s,Fs);
axes(handles.axes9);
plot (s)
xlabel('NUMERO DE MUESTRAS', 'FontSize',9,'color','red' )
ylabel('Nivel de Voz', 'FontSize', 9,'color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 10)

lon=length(s);                                  %Longitud de la señal
d=max(abs(s));                                  %Encontrar la muestra con mayor valor
s=s/d;                                          %Normalizar la señal
prom=sum(s.*s)/lon;                             %Promedio de la señal total
umbral=0.05;                                    %Valor de umbral calculado

y= [ ];
for i = 1:400:lon-400                       %Segmentación de la señal cada 10ms
    seg = s(i:i+399);                           %Almacenar segmentos
    energia = sum(seg.*seg)/400;                %Promedio de energia cada segmento
    if( energia> umbral*prom)                   %Eliminacion de silencios de acuerdo a la condición
    y=[y;seg(1:end)];                           %Almacena la señal en la que se ha eliminado los silencios
    end
end

wavwrite(y,Fs,'D01.wav')
axes(handles.axes12);
plot (y)
xlabel('NUMERO DE MUESTRAS', 'FontSize',9,'color','red' )
ylabel('Nivel de Voz', 'FontSize', 9,'color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 10)
sound (y,Fs);

%FILTRO PREENFASIS
b=[1 -0.95];
y1=filter(b,1,y);
wavwrite(y1,Fs,'D1.wav')
```

```
% --- Executes just before Reconocimiento_Voz_Ninos is made visible.
function Reconocimiento_Voz_Ninos_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Reconocimiento_Voz_Ninos (see VARARGIN)
% Choose default command line output for Reconocimiento_Voz_Ninos

clear a;
global a;
%%delete(instrfindall);
a=arduino('COM2');
%%instrfind
 a.pinMode(3,'output');
 a.pinMode(4,'output');
 a.pinMode(5,'output');
 a.pinMode(6,'output');
 a.pinMode(7,'output');
 handles.a=a;
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Reconocimiento_Voz_Ninos wait for user response (see UIRESUME)
% uiwait(handles.figure1);
```

```
% --- Executes on button press in grabar.
function grabar_Callback(hObject, eventdata, ~)
% hObject    handle to grabar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
Fs=22050;                                   % Frecuencia de muestreo
voz=wavrecord(2*Fs,Fs,1);                   %función de grabacion
wavwrite(voz,Fs,16,'vozusuario.wav')
sound (voz,Fs)
[voz,Fs]=wavread('vozusuario.wav');         %genera señal discreta apartir de archivo .wav
lon=length(s);                              %Longitud de la señal
d=max(abs(s));                              %Encontrar la muestra con mayor valor
s=s/d;                                      %Normalizar la señal
prom=sum(s.*s)/lon;                         %Promedio de la señal total
umbral=0.05;                                %Valor de umbral calculado
y= [ ];
for i = 1:400:lon-400                       %Segmentación de la señal cada 10ms
    seg = s(i:i+399);                       %Almacenar segmentos
    energia = sum(seg.*seg)/400;            %Promedio de energia cada segmento
    if( energia> umbral*prom)               %Condicion para determinarla seañal y descartar el silencio
    y=[y;seg(1:end)];                       %Almacena la señal sin silencios
    end
end
wavwrite(y,Fs,'Silencios.wav')
%FILTRO PREENFASIS
 b=[1 -0.95];
 y1=filter(b,1,y);
 wavwrite(y1,Fs,'Limpia_usuario.wav')
 sound (y1,Fs)
```

```
% --- Executes on button press in Identificar.
function Identificar_Callback(hObject, eventdata, handles)
% hObject    handle to Identificar (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global a;
global Distancias;
global Distan;
global lpcusuario;

load patron;
[g_usuario,Fs,bits]=wavread('Limpia_usuario.wav');
lpcusuario=lpc(g_usuario,23);
transpuestalpc=lpcusuario';
save lpcLimpia_usuario.mat lpcusuario;
```

```
%----CALCULO DE LAS DISTANCIAS

dis1=((PatronD-lpcusuario).^2); %Distancia entre patron Derecha y lpcusuario
dist1=sum(dis1);                %Sumatoria para calculo de distancia total

dis2=((PatronIZ-lpcusuario).^2);% Distancia entre patron Izquierda y lpcusuario
dist2=sum(dis2);                %Sumatoria para calculo de distancia total

dis3=((PatronAD-lpcusuario).^2);% Distancia entre patron Adelante y lpcusuario
dist3=sum(dis3);                %Sumatoria para calculo de distancia total

dis4=((PatronAT-lpcusuario).^2);%Distancia entre patron Atras y lpcusuario
dist4=sum(dis4);                 %Sumatoria para calculo de distancia total

dis5=((PatronP-lpcusuario).^2);% Distancia entre patron Para y lpcusuario
dist5=sum(dis5);                %Sumatoria para calculo de distancia total

Distancias=[dist1,dist2,dist3,dist4,dist5];
```

```
 %..COMPARACIÓN DE DISTANCIAS TOTALES

 if      (dist1<dist2)&(dist1<dist3)&(dist1<dist4)&(dist1<dist5)
         Distan=dist1;
         set(handles.text1,'String','DERECHA' )

          handles.a.digitalWrite(3,1);%
          handles.a.digitalWrite(4,0);%
          handles.a.digitalWrite(5,0);%
          handles.a.digitalWrite(6,0);%
          handles.a.digitalWrite(7,0);%

 elseif    (dist2<dist1)&(dist2<dist3)&(dist2<dist4)&(dist2<dist5)
           Distan=dist2;
          set(handles.text1,'String','IZQUIERDA')

          handles.a.digitalWrite(3,0);%
          handles.a.digitalWrite(4,1);%
          handles.a.digitalWrite(5,0);%
          handles.a.digitalWrite(6,0);%
          handles.a.digitalWrite(7,0);%

 elseif (dist3<dist1)&(dist3<dist2)&(dist3<dist4)&(dist3<dist5)
         Distan=dist3;
         set(handles.text1,'String','ADELANTE')

         handles.a.digitalWrite(3,0);%
         handles.a.digitalWrite(4,0);%
         handles.a.digitalWrite(5,1);%
         handles.a.digitalWrite(6,0);%
         handles.a.digitalWrite(7,0);%

elseif   (dist4<dist1)&(dist4<dist2)&(dist4<dist3)&(dist4<dist5)
          Distan=dist4;
        set(handles.text1,'String','ATRAS')

        handles.a.digitalWrite(3,0);%
        handles.a.digitalWrite(4,0);%
        handles.a.digitalWrite(5,0);%
        handles.a.digitalWrite(6,1);%
        handles.a.digitalWrite(7,0);%

else  (dist5<dist1)&&(dist5<dist2)&(dist5<dist3)&(dist5<dist4)
        Distan=dist5;
        set(handles.text1,'String','PARA')

         handles.a.digitalWrite(3,0);%
         handles.a.digitalWrite(4,0);%
         handles.a.digitalWrite(5,0);%
         handles.a.digitalWrite(6,0);%
         handles.a.digitalWrite(7,1);%

end
```

```
% --- Executes on button press in Salir.
function Salir_Callback(hObject, eventdata, handles)
% hObject    handle to Salir (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
ans=questdlg('Desea salir del programa?','Salir','Si','No','No');
if strcmp(ans,'No')
    guidata(hObject,handles);
    return;
else
    clear all
    close

end
```

```
% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)

global Distancias;
global Distan;

   contents =cellstr(get(hObject,'String'));
   popChoice=contents{get(hObject,'Value')};

   if(strcmp(popChoice,'-Adquisicion'))

                    %GRAFICA EN FUNCION DEL TIEMPO

                    Fs=22050; %frecuencia de muestreo
                    [voz,Fs]=wavread('vozusuario.wav');
                    voz1 = voz(:, 1);                    % tomar el primer canal
                    L = length(voz);
                    t = (0:L-1)/Fs;                     % tiempo del vector
                    axes(handles.axes1);
                    plot(t, voz1, 'b')
                    xlim([0 max(t)])
                    ylim([-1.1*max(abs(voz1)) 1.1*max(abs(voz1))])
                    grid on
                    whitebg('c')
                    xlabel('TIEMPO(s)', 'FontSize', 9,'color','red')
                    ylabel('AMPLITUD', 'FontSize', 9','color','red')
                    title('SEÑAL EN EL DOMNIO DEL TIEMPO','color','red')
                    set(gca, 'FontName', 'Times New Roman', 'FontSize', 10)


                    %GRAFICA EN FUNCION DE LA FRECUENCIA

                    Fs=22050; %frecuencia de muestreo
                    [voz,Fs]=wavread('vozusuario.wav');
                    L = length(voz);                     % longitud de la señal
                    NFFT = 2^nextpow2(L);
                    Y = fft(voz, NFFT)/L;
                    f = Fs/2*linspace(0,1,NFFT/2+1);
                    axes(handles.axes2);
                    plot(f, 2*abs(Y(1:NFFT/2+1)));
                    grid on
                    whitebg('c')
                    xlabel('FRECUENCIA(s)', 'FontSize', 9,'color','red')
                    ylabel('AMPLITUD', 'FontSize', 9,'color','red')
                    title('SEÑAL EN EL DOMNIO FRECUENCIAL','color','red')
                    set(gca, 'FontName', 'Times New Roman', 'FontSize', 9)


                    %GRAFICA NUMERO DE MUESTRAS

                    axes(handles.axes3);
                    plot(voz);

                    title('SEGMENTADA', 'FontSize', 9,'color','red')
                    xlabel('NUMERO DE MUESTRAS', 'FontSize',9,'color','red' )
                    ylabel('Nivel de Voz', 'FontSize', 9,'color','red')
                    set(gca, 'FontName', 'Times New Roman', 'FontSize', 10)

        elseif  (strcmp(popChoice,'-Eliminacion Silencios'))
```

```
%GRAFICA EN FUNCION DEL TIEMPO

Fs=22050;                               %frecuencia de muestreo
[silencio,Fs]=wavread('Silencios.wav');
silencio1 = silencio(:, 1);             % tomar el primer canal
Ls = length(silencio);
ts = (0:Ls-1)/Fs;                       % tiempo del vector
axes(handles.axes1);
plot(ts, silencio1, 'b')
xlim([0 max(ts)])
ylim([-1.1*max(abs(silencio1)) 1.1*max(abs(silencio1))])
grid on
whitebg('c')
xlabel('TIEMPO s)', 'FontSize', 9,'color','red')
ylabel('AMPLITUD', 'FontSize', 9,'color','red')
title('SEÑAL EN EL DOMNIO TIEMPO','color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 9)

%GRAFICA EN FUNCION DE LA FRECUENCIA
NFFTs = 2^nextpow2(Ls);
Ys = fft(silencio, NFFTs)/Ls;
fs = Fs/2*linspace(0,1,NFFTs/2+1);
axes(handles.axes2);
plot(fs, 2*abs(Ys(1:NFFTs/2+1)));
grid on
whitebg('c')
xlabel('FRECUENCIA(s)', 'FontSize', 9,'color','red')
ylabel('AMPLITUD', 'FontSize', 9,'color','red')
title('SEÑAL EN EL DOMNIO FRECUENCIAL','color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 9)

%GRAFICA NUMERO DE MUESTRAS
axes(handles.axes3);
plot(silencio);
title('SEGMENTADA', 'FontSize', 9,'color','red')
xlabel('NUMERO DE MUESTRAS', 'FontSize',9,'color','red' )
ylabel('Nivel de Voz', 'FontSize', 9,'color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 10)

elseif  (strcmp(popChoice,'-Preenfasis'))

 %GRAFICA EN FUNCION DEL TIEMPO

Fs=22050;                               %frecuencia de muestreo
[silencio,Fs]=wavread('Limpia_usuario.wav');
silencio1 = silencio(:, 1);             % tomar el primer canal
Ls = length(silencio);
ts = (0:Ls-1)/Fs;                       % tiempo del vector
axes(handles.axes1);
plot(ts, silencio1, 'b')
xlim([0 max(ts)])
ylim([-1.1*max(abs(silencio1)) 1.1*max(abs(silencio1))])
grid on
whitebg('c')
xlabel('TIEMPO s)', 'FontSize', 9,'color','red')
ylabel('AMPLITUD', 'FontSize', 9,'color','red')
title('SEÑAL EN EL DOMNIO TIEMPO -Filtro Preenfasis','color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 9)

%GRAFICA EN FUNCION DE LA FRECUENCIA
NFFTs = 2^nextpow2(Ls);
Ys = fft(silencio, NFFTs)/Ls;
fs = Fs/2*linspace(0,1,NFFTs/2+1);
axes(handles.axes2);
plot(fs, 2*abs(Ys(1:NFFTs/2+1)));
grid on
whitebg('c')
xlabel('FRECUENCIA(s)', 'FontSize', 9,'color','red')
ylabel('AMPLITUD', 'FontSize', 9,'color','red')
title('SEÑAL EN EL DOMNIO FRECUENCIAL--Filtro Preenfasis','color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 9)

%GRAFICA NUMERO DE MUESTRAS
axes(handles.axes3);
plot(silencio);
title('SEGMENTADA --Filtro Preenfasis', 'FontSize', 9,'color','red')
xlabel('NUMERO DE MUESTRAS', 'FontSize',9,'color','red' )
ylabel('Nivel de Voz', 'FontSize', 9,'color','red')
set(gca, 'FontName', 'Times New Roman', 'FontSize', 10)
```

```
                %GRAFICA NUMERO DE LPC
            elseif  (strcmp(popChoice,'-LPC'))

                %######ERROR DE PREDICCIÓN ##########
                load  lpcLimpia_usuario.mat
                [s,fs,bits]=wavread('Limpia_usuario.wav');
                N=240;%numero de puntos para la ventana de hamming
                v=hamming(240);
                t=1:240;% variable para el numero de muestras
                sw=v.*s(t); %mutlipicacion de la señal de voz por la ventana de hamming
                sw=sw/max(abs(sw));%normalizamos la señal
                coeflpc=lpc(sw,23);
                est_sw = filter([0 -coeflpc(2:end)],1,sw);%se aplica el filtro de predicción H(z)=A(z)
                e = sw - est_sw;% se calcula el Error de predicción
                axes(handles.axes1);
                plot(v,'r');
                title('VENATANA DE HAMMING', 'FontSize', 9,'color','red')
                xlabel('Muestas');
                ylabel('Nivel de Voz');
                hold on;
                plot(sw);
                hleg1 = legend('Ventana.Hamming','Limpia.usuario.ventanada',2)
                set(hleg1,'FontSize',7);
                hold off;
                axes(handles.axes2);
                plot(sw,'y','linewidth',2);
                title('ERROR DE PREDICCIÓN', 'FontSize', 9,'color','red')
                xlabel('Muestas');
                ylabel('Nivel de Voz');
                hold on;
                plot(est_sw,'r');
                plot(e,'k');
                hleg1 = legend('Limpia.usuario.ventanada','Señal.estimada','Error.predicción',2)
                set(hleg1,'FontSize',7);
                hold off;
                %#######MODELO AR ##########

                [g_usuario,Fs]=wavread('Limpia_usuario.wav');%señal de voz eliminada los Silencios
                num1=19;%numero de coeficientes Lpc
                op1 = length(g_usuario);%longitud de la señal
                Rsw1 = xcorr(g_usuario);% Vector de Autocorrelacion
                R1 = Rsw1(op1:op1); % Obtención R(0)
                G1 = sqrt(sum( lpcusuario.*R1')); % Obtención de la Ganancia
                envolvente1 = abs(G1./fft( lpcusuario,op1));% Obtención de la envolvente H(z)
                SW1 = abs(fft(g_usuario,op1)); % Transformada de Fourier de la señal original

                %Graficas
                axes(handles.axes3);
                plot(20*log10(SW1(1:(op1/2))))%Graffica Transformada de Fourier de la señal original
                hold on;
                plot(20*log10(envolvente1(1:(op1/2))),'m','linewidth',2)%Graffica  envolvente señal
                title('MODELO AR ', 'FontSize', 9,'color','red')
                xlabel('Frecuencia');
                ylabel('Ganancia');
                hleg1 = legend('Señal.Limpia.usuario','envolvente',3)
                set(hleg1,'FontSize',7);
                hold off;

        elseif  (strcmp(popChoice,'  Formantes'))
                    Formantes;
                elseif  (strcmp(popChoice,'-Distancia'))

                set(handles.text4,'String',Distancias)
                set(handles.text5,'String',Distan)
    end

guidata(hObject, handles);



% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2 contents as cell array
%        contents(get(hObject,'Value')) returns selected item from popupmenu2
```

```
% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)

%###################METODO PEAK-PICKING  ########
[x,fs]=wavread('Limpia_usuario');%señal de voz eliminada los silencios
a=lpc(x,19);%variable que depermina los coeficientes Lpc donde
%(señal silencios,#de coeficientes)
[h,f]=freqz(1,a,240,fs);%respuesta en frecuencia de un filtro digital donde:
%freqz(numerador,denominador,numero de puntos ,Frecuencia de muestreo)
axes(handles.axes1);
plot(f,20*log10(abs(h)+eps));%Grafica del Filtro Digital
legend('Envolvente H(Z)');
title('Envolvente_ H(Z)', 'FontSize', 9,'color','red')
xlabel('Frequency (Hz)');
ylabel('Gain (dB)');

%##############FORMANTE METODO DE LAS RAICES

roots_a=roots(a); % la variable roots_a calcula el valor de las raices
formants_a=angle(roots_a)*fs/(2*pi);%Determina las frecuencia de los Formantes Fi=?_i/(2?T_s )
a_sorted = sort(abs(formants_a));% determina el valor absoluto ,ordena el vector

%imprimir cada 2 posicion debido a que cada par de polos complejos conjugados corresponde a un formante
set(handles.text2,'String',a_sorted(2))% formante F1
set(handles.text3,'String',a_sorted(4))% formante F2
set(handles.text4,'String',a_sorted(6))% formante F3


% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```