



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL
Facultad de Ingeniería en Electricidad y Computación

“DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO PARA EL
RECONOCIMIENTO DE SONIDO DE DISPAROS Y
SOLICITUDES DE AYUDA QUE ALERTARÁN A UN CENTRO
DE OPERACIONES REMOTO”

INFORME DE MATERIA INTEGRADORA

Previa a la obtención del Título de:
INGENIERO EN TELEMÁTICA

Presentado por:

Carlos Fernando Jumbo Sánchez
Gustavo Alfredo Totoy Guananga

GUAYAQUIL – ECUADOR

AÑO: 2016

AGRADECIMIENTOS

Mis más sinceros agradecimientos a Dios por guiarme a lo largo de la carrera, a mis padres por su contante apoyo incondicional, y al club Rivera del Lago por darnos las facilidades en las pruebas con armas de fuego.

Carlos Jumbo Sánchez

Agradezco primero a Dios por todas las bendiciones que me ha dado, tanto a nivel profesional como personal, así mismo doy las gracias a mis familiares y amigos por el apoyo que me brindaron y ayudaron a perseverar día a día.

Gustavo Totoy Guananga

DEDICATORIA

El presente proyecto lo dedico a mis padres, ellos siempre creyeron en mí y estuvieron empujándome en los momentos más difíciles. Sus ejemplos de superación y entrega, me motiva a lograr las metas que me proponga en esta vida.

Carlos Jumbo Sánchez

Este trabajo está dedicado a Dios, a mi familia y amigos quienes me brindaron su constante apoyo para salir adelante y lograr la meta propuesta.

Gustavo Totoy Guananga

TRIBUNAL DE EVALUACIÓN

Ing. Marcos Millán

PROFESOR EVALUADOR

Ing. Néstor Arreaga

PROFESOR EVALUADOR

DECLARACIÓN EXPRESA

"La responsabilidad y la autoría del contenido de este Trabajo de Titulación, nos corresponde exclusivamente; y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

Carlos Fernando Jumbo Sánchez

Gustavo Alfredo Totoy Guananga

RESUMEN

El proyecto realizado es solo un primer prototipo para poder detectar sonidos de disparos en ambientes reales y solicitudes de ayuda por parte de la ciudadanía; cabe recalcar que, para realizar el muestreo de la señal deseada, se realizaron prácticas reales con armas de fuego en un polígono de disparo para su posterior análisis.

Dado que en la actualidad es ilegal portar y usar armas de fuego se realizaron comparaciones entre la señal producida por el arma de fuego y un aplauso, para poder observar cual es el grado de similitud entre ambas señales.

Sin embargo, para poder realizar esta comparación, se implementó un pequeño prototipo que comprenden sistemas embebidos como el Raspberry Pi 3 Modelo B, Arduino Genuino Uno y su respectivo Módulo de Reconocimiento de Voz VR3. En conjunto estos dispositivos conformaran un nodo que podrá estar trabajando como sistema auxiliar para un sistema de seguridad.

El nodo receptorá las solicitudes de ayuda y los sonidos producidos por un arma de fuego para su debido procesamiento; una vez ingresado las señales en el sistema, este determinara qué tipo de información se procesó y sus resultados serán almacenados en una pequeña base de datos que estará en constante comunicación con un centro de operaciones.

De acuerdo a la información que llegase al centro de operaciones, será un operador remoto quien determinara que acciones tomar por los valores dados del sistema. Dado que el sistema no es 100% automático, será un operador el que tendrá la obligación de informar inmediatamente a las autoridades competentes lo que recepta el nodo.

ÍNDICE GENERAL

DEDICATORIA	iii
TRIBUNAL DE EVALUACIÓN	iv
DECLARACIÓN EXPRESA	v
RESUMEN	vi
ÍNDICE GENERAL.....	vii
ÍNDICE DE FIGURAS.....	ix
ÍNDICE DE TABLAS	x
CAPÍTULO 1	1
1. DESCRIPCIÓN GENERAL	1
1.1 Antecedentes.....	1
1.2 Justificación	4
1.3 Objetivos del proyecto	5
1.3.1 Objetivo General.	5
1.3.2 Objetivos Específicos	5
1.4 Alcances y Limitaciones.....	6
CAPÍTULO 2.....	7
2 ANÁLISIS Y DISEÑO	7
2.1 Planificación.....	7
2.1.1 Módulo de reconocimiento de voz.....	8
2.1.2 Módulo de reconocimiento de disparo.....	8
2.2 Marco Teórico.....	8
2.2.1 Señal de audio digital.....	8
2.2.2 Espectro de frecuencia.....	9
2.2.3 El filtro de Wiener	12
2.3 Software.....	16
2.3.1 Matlab	16
2.3.2 Simulink.....	17
2.4 Hardware	17
2.4.1 Sistemas Embebidos.....	17

2.4.2	Arduino.....	18
2.4.3	Módulo VR3	21
2.4.4	Raspberry Pi 3 Modelo B	23
2.4.5	Tarjeta de Sonido Estéreo USB Externa.....	24
CAPÍTULO 3.....		25
3	IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS	25
3.1	Arquitectura del sonido	25
3.2	Desarrollo del nodo detector de sonido	25
3.2.1	Primera Fase.....	26
3.2.2	Segunda Fase.....	26
3.2.3	Tercera Fase.....	31
3.3	Funcionamiento del nodo.....	32
3.3.1	Energización del sistema	32
3.3.2	Recepción de señales	32
3.3.3	Transmisión de datos	33
3.3.4	Recepción de datos.....	33
3.3.5	Centro de monitoreo.....	33
3.4	Instalación y Configuración.....	34
3.5	Pruebas y Soluciones	34
3.6	Resultados Obtenidos.....	37
3.6.1	Simulaciones en Matlab	37
3.6.2	Simulación en Simulink	40
3.7	Costos de implementación.....	42
CONCLUSIONES Y RECOMENDACIONES		43
BIBLIOGRAFÍA.....		45
ANEXOS.....		47

ÍNDICE DE FIGURAS

Figura 2.1: Esquema General del prototipo.....	7
Figura 2.2: Espectro de potencia de la señal discreta $x(t)$	10
Figura 2.3: Espectrograma y Espectro de frecuencia de la señal discreta $x(t)$	11
Figura 2.4: Espectro de frecuencia normalizado de la señal $x(t)$	12
Figura 2.5: Filtro de Wiener	13
Figura 2.6: Arduino Genuino Uno	20
Figura 2.7: Modulo de reconocimiento de voz VR3.....	22
Figura 2.8: Raspberry Pi 3 Modelo B	23
Figura 2.9: Tarjeta de sonido Estéreo USB Externa.....	24
Figura 3.1 Sistema de identificación	25
Figura 3.2: Configuración del módulo de reconocimiento de disparo	27
Figura 3.3: Modelo en Simulink.....	30
Figura 3.4: El modelo se encuentra en ejecución.....	31
Figura 3.5: Led Rojo indica que se guarda en base de datos	31
Figura 3.6: Consumo de CPU en la primera prueba de campo	35
Figura 3.7: Consumo del CPU con los cambios al modelo Simulink	36
Figura 3.8: Grabación de disparos con una arma calibre 9mm, a 40 metros del micrófono.....	37
Figura 3.9: Fragmento de 0.2 segundos	37
Figura 3.10: Espectro de frecuencia de la señal digital del audio.....	38
Figura 3.11: Espectro de frecuencia de la señal deseada.....	38
Figura 3.12: Gráfico de Error Cuadrático Medio.....	39
Figura 3.13: Espectros de frecuencia.....	39
Figura 3.14: Gráfico de Error Cuadrático Medio.....	40
Figura 3.15: Modelo para pruebas en pc	40
Figura 3.16: Detección del primer disparo	41
Figura 3.17: Detección del segundo disparo	41

ÍNDICE DE TABLAS

Tabla 1: Especificación técnica de Arduino Uno	21
Tabla 2: Especificación técnica del módulo VR3.....	22
Tabla 3: Costo de Implementar el nodo	42

CAPÍTULO 1

1. DESCRIPCIÓN GENERAL

1.1 Antecedentes

En las últimas décadas se ha visto un gran avance tecnológico en el estudio de señales y reconocimiento en patrones de sonido, dando lugar a que se pueda ir implementando dispositivos y sistemas que permitan identificar diferentes tipos de señales o un patrón en específico para su debido estudio y procesamiento del mismo.

Por consiguiente, existen diversos tipos de sistemas y mecanismos, que han sido diseñado e implementado para analizar diferentes tipos de señales, como por ejemplo las señales que son emitidos por la propia naturaleza, tal es el caso de la onda calórica que emana un ser vivo y otras señales que son producido por el accionamiento de componentes mecánicos o eléctricos, como los que puede producir un dispositivo de comunicación o el disparo de un arma de fuego.

Por lo que es de relevancia acotar que, a lo largo de la historia, el estudio y análisis de las señales han sido tomados en cuenta para acciones decisivas en conflictos bélicos, debido a los diferentes tipos de armas y que inclusive desde el inicio de la Revolución Industrial la producción de armas de fuego tuvo su auge. Claro está que ahora existen leyes y controles exhaustivos para las personas y compañías encargadas de usar estas armas, que son de tenencia responsable para estos mecanismos; aun así, el uso de las armas de fuego no discrimina quien sea capaz de usarlas.

Por lo que se debe recordar que en la Primera y en la Segunda Guerra Mundial se han ido diseñando y desarrollando los primeros prototipos para la localizar la artillería en el campo enemigo; dado que, a su facilidad de movimiento, estos tendían a ser ocultos fácilmente. Para su localización los científicos estaban probando de nuevo aparatos, donde usaban Radar y Sonidos de corto alcance basados en el Efecto Boomerang. [1]

Por lo que a inicio de la década de los 60 aproximadamente y a final del siglo XX, los sistemas de localización de armas en el campo de batalla poco a poco se ha ido modificando acorde a la necesidad de la milicia, que de tal manera los sistemas para identificar disparos poco a poco se han hecho obsoletos se fueron modificando por sistemas mejores como el de posicionamiento global y sistemas sensoriales que de tal manera triangulan brindando un resultado más eficiente que los sistemas usados en Radar.

Actualmente en los conflictos bélicos, los vehículos de artillería suelen contar el sistema Boomerang, este sistema fue diseñado en los primeros años del siglo XX, dado que notifica a los soldados dentro de un móvil, si ha realizado un disparo, la dirección y elevación proviene el disparo. Claro está que el sistema fue diseñado solo para cierto tipo de armas comúnmente usadas en el campo de batalla. [2]

Por ende, cuando un arma de fuego es utilizada en vía pública, los investigadores se complicaban en el proceso de investigación para el reconocimiento del autor material del hecho. Adicionalmente las evidencias para estimar el lugar de donde provino el impacto sonoro producido por el arma a veces no eran muy contundentes, debido a que no había testigos oculares o sistemas de vigilancia que puedan corroborar el hecho.

En efecto las grandes entidades, como las compañías de seguridad y gobiernos implementan diversos sistemas en diferentes puntos de la ciudad, precisamente en zonas con mayor índice de robo a mano armada, asesinatos entre otros actos delictivos. Por lo que se pretendía desde un principio reducir los actos delictivos; pero pese a los sistemas de seguridad actuales, estos no han logrado reducir los índices considerablemente.

En base a la necesidad de aumentar la seguridad en las ciudades y tener un mayor control que es lo que sucede internamente en las dentro de las entidades, estos han optado por usar diversos sistemas de seguridad; el cual les da un mayor control sobre las actividades que realizan los ciudadanos. Permitiendo de a poco a poco que desde las pequeñas y medianas empresas opten diferentes maneras de registrar un evento que contemple alertas o alarmas.

Logrando así tener un registro de lo que acontece durante las 24 horas del día. Un ejemplo de sistema de seguridad existente, es el monitoreo por parte de operadores remotos en el famoso sistema de vigilancia Ojos de águila; cuyas grabaciones son usados como pruebas judiciales.

Donde los actos delictivos como secuestros express, robos simples, atracos de sacapintas, incendios, accidentes de tránsito, manifestaciones entre otro delito permiten detener a los sospechosos según las novedades registradas por los operadores que manejan actualmente las cámaras del sistema. [3]

Pese a que el sistema esta implementado en casi todas las ciudades principales del país, el sistema le otorga al operador la capacidad de monitorear 8 videos registrados por las cámaras, pero entre tantas cámaras que están cargo del operador, este no logra detectar si existe sonido producidos por proyectiles o si un individuo solicita algún tipo de ayuda.

Puesto que el operador cuando se enfoca en las cámaras que tiene a su disposición, este no puede estar al 100% observando todas a la vez, sin contar que solo ve lo que enfoca la cámara; pero si existe el sonido de un disparo de fuego o solicitud de ayuda de una persona y esta se encuentra fuera del enfoque y está dentro del rango que las cámaras que vigilan, el operador no será capaz de percibir el audio que tampoco está implementado en el sistema.

En Ecuador la violencia interpersonal supera a la criminal en los casos de muertes violentas; pero esa situación cambia en Guayaquil. “Hay muchos problemas de organizaciones delincuenciales y lucha entre ellas. Tenemos identificados algunos grupos” sostuvo La Policía Nacional. Además, añadió que se está elaborando una medición semanal del uso de armas de fuego en este tipo de delitos. Este porcentaje también ha reducido. En 2012 hubo un 66% de uso de armas de fuego en crímenes; en 2013, 61%; en 2014, 58%; en 2015, 51%; y en el presente año, 43%. La finalidad es alcanzar el 36%. Esto se hace mediante controles permanentes a zonas identificadas”. [4]

“La cámara de Ojos de Águila no captó la balacera” [5] es el título de un diario informativo en Ecuador, contando que solo es un caso que sale a luz de tantos crímenes que existen en la ciudad. Esto nos indica que los sistemas de vigilancia

ubicados en diversos puntos de la ciudad no pueden identificar sonidos balísticos para que el operador tomar acciones inmediatas; dado que el sistema de cámaras de seguridad no las percibe o identifica si una persona utiliza un arma de fuego.

Pese a estas desventajas que presentan los sistemas en las grandes ciudades, diversas compañías han optado por implementar otros tipos de sistemas que permitan detectar la localización de donde surge el disparo de un arma de fuego; estos sistemas están incorporados en los vehículos de los policías, que por medio de triangulación de sensores logran estimar donde se originó el disparo. Por lo general estos sistemas son implementados en las zonas donde se producen enfrentamiento de bandas, para poner un alto al crimen organizado y la tenencia ilegal de armas.

Pero tales sistemas no son confiables del todo, debido a que cuando son otros tipos de sonidos similares a los de un arma de fuego, los sistemas tienden a dar señal de un falso positivo, debido a que no hay un video o una imagen que corrobore que el sonido es producido por impacto proyectil o solicitud de ayuda.

1.2 Justificación

Actualmente las entidades públicas encargadas de la seguridad ciudadana tienen implementado en las diferentes zonas de su urbe, el sistema ojos de águila que no son confiable en su totalidad, pero pese a ello las tienen ubicadas en los puntos más transitados y violentas de la ciudad para un mayor control y así evitar actos delictivos y crímenes violentos.

Cabe destacar que a medida que aumenta el desarrollo tecnológico tanto a nivel investigativo como a nivel empresarial, siempre se está buscando cubrir las exigencias o necesidades que se presentan en una comunidad u organización. Para nuestro caso en particular se busca diseñar e implementar un prototipo que sea adaptable a las cámaras de seguridad y de fácil acceso para las organizaciones públicas como privadas para tener un control de su entorno.

El prototipo implementado, detectará los sonidos producidos por un arma de fuego, además la detección de palabras claves que impliquen algún tipo de solicitud de ayuda por parte de la ciudadanía, y este transmitirá una alarma al

operador que monitorea las diferentes cámaras, indicando en cual se produce el hecho.

La factibilidad de esta solución es que nos permitirá brindarle mayor seguridad a la ciudadanía, permitiendo monitorear incluso el uso de un arma de fuego, debido a que es ilegal portarla si no tiene los documentos necesarios que respalden su tenencia responsable, ya que a medida considerable se ha reportado accidentes por balas perdidas, por lo que hasta el hecho de disparar un arma al aire libre es ilegal.

Con este prototipo se pretende controlar el uso y tenencia responsable de un arma de fuego, adicionalmente el prototipo también podrá alertar al operador si se presentan emergencias como solicitud de ayuda por parte de la ciudadanía ya sean estas como pidiendo auxilio por robo o hasta la notificación de un incendio.

Visto desde otro contexto es necesario e indispensable implementar este prototipo, puesto que cuando suceda un accidente o catástrofe de cualquier índole el sistema de seguridad de cámaras nos permitirá ubicar al personal necesario inmediatamente y tomar las medidas necesarias y pertinente según el caso que se presente para su respectivo rescate, evacuación y entre otros.

1.3 Objetivos del proyecto

1.3.1 Objetivo General.

Implementar un prototipo que mejore el monitoreo en tiempo real de las zonas conflictivas de la urbe mediante el uso de sensores, para identificar cualquier tipo de acto delictivo y solicitud de ayuda por parte de la ciudadanía.

1.3.2 Objetivos Específicos

- Implementar un módulo de reconocimiento de voz, que nos permitirá registrar que tipo de ayuda está solicitando el ciudadano y así informar a las autoridades competentes que tipo de ayuda necesita.
- Identificar los tipos de armas más usados para los actos ilícitos y de fácil acceso para los ciudadanos de manera ilegal, debido a que se estudiara el patrón que tiene cada arma de fuego.

- Analizar el patrón que es emitido por un arma de fuego, para comparar con varias muestras de disparos de diferente calibre y así reconocer que tipo o modelo fue usado para el acto.
- Establecer la comunicación entre el prototipo con el operador que monitorea las cámaras de seguridad, ya que se comunicaran por medio de una conexión remota, para enviar la señal de emergencia.
- Diseñar un prototipo a través de un sensor y lenguaje de programación, notificando al operador que monitorea las cámaras en tiempo real que tipo de incidentes ocurre dentro de la zona cubierta por el sistema de seguridad.

1.4 Alcances y Limitaciones

El prototipo proporcionará la información necesaria al operador en caso de que se presente una eventualidad de gran índole como que tipo de solicitud de ayuda realiza o si existe un uso de armas de fuego en las zonas cubiertas por el sistema de seguridad, para que se pueda tomar las medidas necesarias en tiempo real.

Dentro las limitaciones del proyecto, se debe tener en cuenta lo siguiente:

- Para la solicitud de ayuda, el ciudadano tendrá que gritar fuertemente y legible que tipo de ayuda necesita para que el módulo reconozca y notifique inmediatamente al operador. En caso de que el ciudadano hable o solicite la ayuda necesaria en voz baja como una conversación, este sistema no reconocerá la solicitud.
- Dado que en el mercado existen diferentes tipos de armas de fuego con diferentes calibres, el prototipo solo contará en su base de datos los patrones de las armas de fuego usados en la práctica de tiro. Que son comúnmente usados para los actos ilícitos. Además, los disparos de proyectil serán reconocidos por el sistema dentro de una cobertura no mayor de 40 metros; y para descartar en caso de que se halle una señal con falso positivo, el operador será quien determinara si es o no un impacto de proyectil.

CAPÍTULO 2

2 ANÁLISIS Y DISEÑO

2.1 Planificación

El proyecto está diseñado para informar inmediatamente a un operador remoto de seguridad, mediante el uso de telemetría la información necesaria que percibe nuestro prototipo ubicado junto a un sistema de seguridad.

Para el desarrollo de este proyecto, se debe tener claro el esquema general del prototipo, el cual se muestra en la figura 2.1.

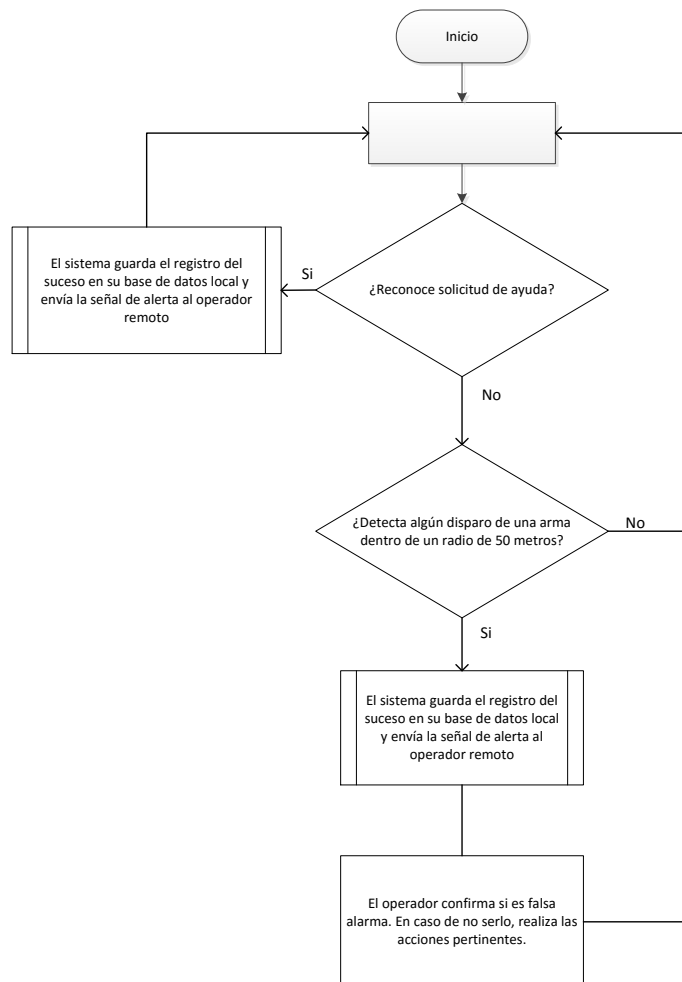


Figura 2.1: Esquema General del prototipo

El diagrama de flujo de la figura 2.1, nos detalla gráficamente cómo funcionará el prototipo. El esquema hace énfasis que es obligatorio la colaboración de una persona para confirmar si el prototipo envió un falso positivo en el reconocimiento de disparo.

Además, gracias a la figura proporcionada por el sistema, podemos intuir que el prototipo está conformado por dos módulos: un módulo para el reconocimiento de voz y otro para el reconocimiento del disparo, ambos independientes entre sí.

2.1.1 Módulo de reconocimiento de voz

El reconocimiento de voz está implementado para identificar los diferentes tipos de solicitudes de ayuda; este se desarrolló por un sistema embebido, de fácil acceso en el mercado, conocido como módulo VR3. Para saber más detalles de este módulo, ir a la sección 2.4.3.

2.1.2 Módulo de reconocimiento de disparo

Para el reconocimiento de un disparo con arma de fuego, modelamos un sistema de procesamiento digital de señales. El sistema emplea un algoritmo que se basa en la teoría del filtro de Wiener. Este algoritmo requiere que toda señal en dominio de tiempo discreto se transforme a dominio de frecuencia; en otras palabras, se analiza el espectro de frecuencia de la señal.

Antes de considerar el procesamiento de señales, se debe conocer el marco teórico que implica todo el sistema.

2.2 Marco Teórico

2.2.1 Señal de audio digital

En la mayoría de aplicaciones, el sonido analógico primero es muestreado para ser convertido a valores entero sobre su respectiva tasa de muestreo. Este proceso se lo conoce como digitalización, que significa convertir una señal analógica a su formato digital (señal discreta).

Las dos características más importantes de una señal digital de audio son la tasa de muestreo y resolución. La tasa o frecuencia de muestreo es el

número de muestras por unidad de tiempo que se toma de la señal analógica, y la resolución está definida por el número de bits usado para representar una muestra.

De acuerdo con el teorema de muestreo (Teorema de Nyquist), para muestrear adecuadamente una señal analógica se necesita hacerlo con una frecuencia de muestreo igual o mayor que el doble de la máxima de las frecuencias de la señal. A mayor frecuencia de muestreo dará lugar a una mejor señal muestreada para analizar. Relativamente, esto requiere un procesador más rápido para procesar la señal, debido a la mayor cantidad de datos que se manejará.

2.2.2 Espectro de frecuencia

El espectro es la representación de una función sobre sus componentes de frecuencia; por lo general suelen ser conocidos como espectro de potencia o densidad espectral de potencia. La diferencia radica en que el espectro de potencia es la magnitud al cuadrado de la transformada de Fourier y la densidad espectral de potencia es una función que tiene unidades de potencia por Hz.

Así que, mientras que el espectro de potencia se encarga de calcular el área bajo la curva de la señal por medio de la transformada de Fourier, la densidad espectral de potencia asigna unidades de potencia a cada unidad de frecuencia, exaltando periodicidades. [6]

Como ya se mencionó, se usa la transformada de Fourier para conocer el espectro de potencia de una señal. En procesamientos de señales se emplea la FFT (Transformada Rápida de Fourier) en vez de la DFT (Transformada Discreta de Fourier) por la diferencia que muestra respecto a eficiencia y rapidez. Esencialmente, el FFT como DFT transforma la señal discreta desde el dominio del tiempo en su dominio de la frecuencia.

A continuación, la figura 2.2 muestra el espectro de potencia de una señal de audio digitalizado $x(t)$ con una frecuencia de muestreo de 44100 Hz.

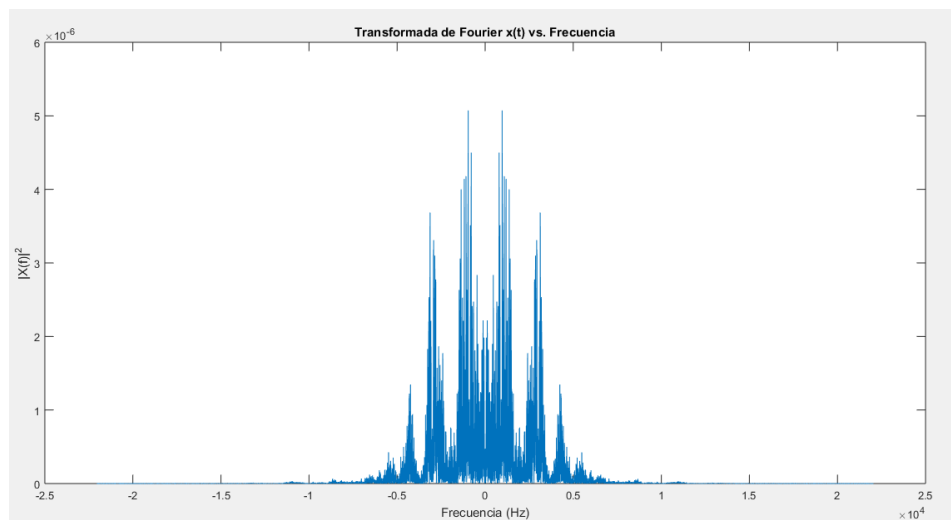


Figura 2.2: Espectro de potencia de la señal discreta $x(t)$

Tal como se puede apreciar en la figura 2.2 el fenómeno de “leakage” o manchado espectral, causa que la función de la gráfica no sea continua; lo cual no es bueno para el análisis posterior. El “leakage” tiene lugar cuando se calcula la transformada de Fourier en señales discretas aperiódicas. Cuando se graba una señal de audio, se obtiene una señal que contiene una gran cantidad de frecuencia por lo que se estaríamos ante la presencia de una señal aperiódica en el dominio de tiempo.

Usando ventanas podemos mejorar la situación. Las ventanas son funciones matemáticas que se aplica para evitar la discontinuidad en el espectro, asociados con intervalos de observación finita. Es recomendable usar una ventana para truncar una larga secuencia de señal en secuencias de tiempo corto. Para secuencias de tiempo corto, la señal puede ser tratado como periódica, y fuera de la ventana se considera todos los valores como ceros. Luego se calcula la FFT a los datos de la señal truncada. Esto es conocido como Short Time Fourier Transform (STFT). Se mantiene la ventana en movimiento a lo largo del eje de tiempo hasta que haya truncado a través de todo el espectro. De esta forma, la ventana no solo reduce el “leakage” del componente en frecuencia, sino que suaviza el espectro. El resultado de todo el proceso mencionado se denomina como el espectrograma de la señal.

La definición del espectrograma se puede resumir como una sucesión de espectros. La figura 2.3 muestra a su lado derecho el espectrograma de la señal $x(t)$ (usada para obtener el espectro de potencia en la figura 2.2), y a su izquierda la sumatoria de los espectros en sus respectivos componentes de frecuencia.

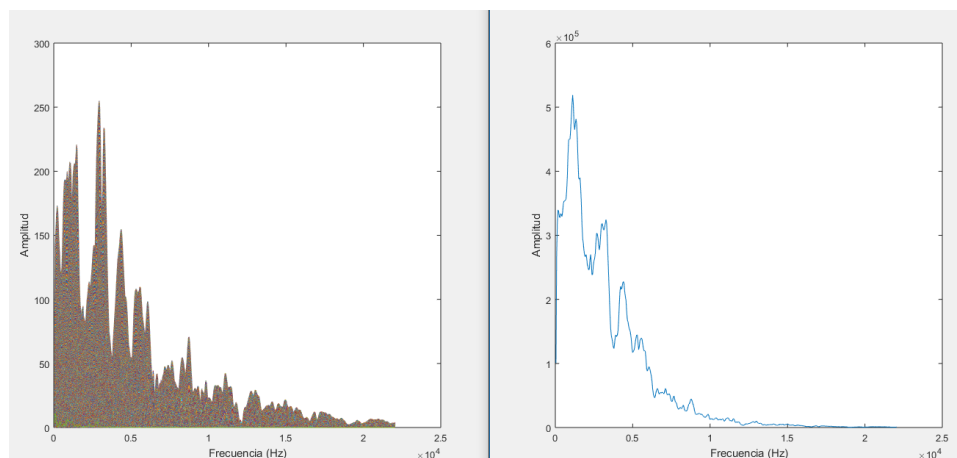


Figura 2.3: Espectrograma y Espectro de frecuencia de la señal discreta $x(t)$

La gráfica del lado derecho de la figura 2.3 representara a la función espectro de frecuencia de la señal $x(t)$, el cual se procede a normalizar. La normalización en una señal consiste en llevar su amplitud más alta a un nivel objetivo, la unidad. Con la normalización buscamos suprimir el efecto de ruido que se encuentra en la señal. La figura 2.4 nos muestra el espectro de frecuencia normalizado de la señal $x(t)$.

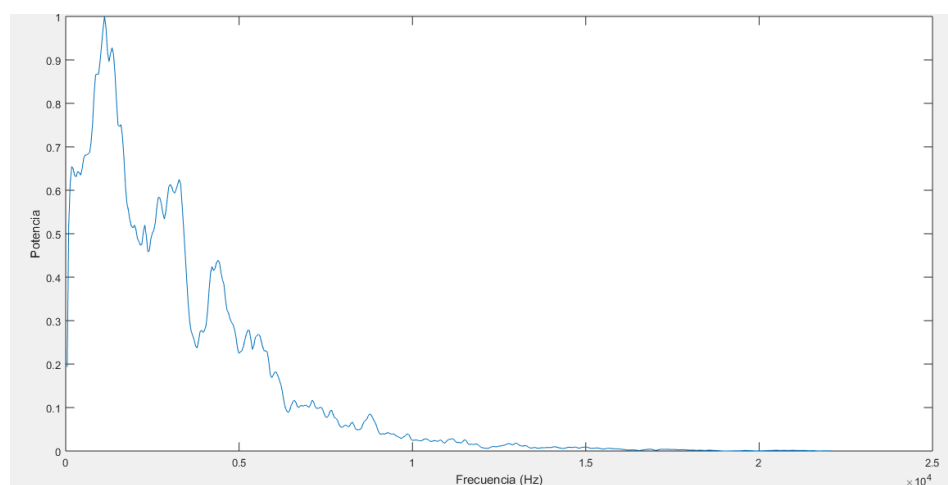


Figura 2.4: Espectro de frecuencia normalizado de la señal $x(t)$

En contraste con la figura 2.2, la figura 2.4 presenta la parte real del espectro de potencia de $x(t)$, normalizado (donde la amplitud más alta es del valor uno) y sin el efecto de “leakage”, que es lo ideal para el análisis posterior.

Emplearemos todo este proceso (de espectrograma hasta la normalización), para definir las señales que se involucran en el algoritmo básico del filtro de Wiener: señal de entrada y señal deseada.

2.2.3 El filtro de Wiener

Propuesto por Norbert Wiener, es un filtro lineal cuyo propósito es, utilizando métodos estadísticos, reducir el ruido presente en la señal de entrada de tal modo que la señal de salida del filtro se aproxime lo más posible (en el sentido cuadrático medio) a una señal deseada (sin ruido) [7].

El filtro de Wiener tiene una variedad de aplicaciones en el procesamiento de señales, imágenes, sistemas de control y comunicaciones digitales, para la eliminación de ruido en una señal. Un estudio realizado demuestra que el algoritmo del filtro también puede trabajar para reconocimiento de voz [8].

Fundamentándonos en ese estudio, adaptamos nuestro modelo de reconocimiento de disparo con el algoritmo básico del filtro de Wiener. A continuación, una breve explicación de la teoría.

El filtro de Wiener trabaja de la siguiente forma:

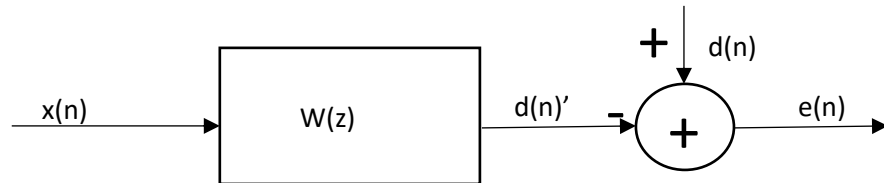


Figura 2.5: Filtro de Wiener

La Fig. 2.5 nos muestra que la señal de entrada en el filtro de Wiener es $x(n)$; asumiendo que los coeficientes del filtro es $w(n)$, la señal de salida $d(n)'$ es:

$$d(n)' = w(n) * x(n) = \sum_{l=0}^{p-1} w(l)x(n-l) \quad (2.1)$$

Por lo que el error de estimación vendría ser:

$$e(n) = d(n) - \sum_{l=0}^{p-1} w(l)x(n-l) \quad (2.2)$$

El propósito del filtro de Wiener es elegir el orden adecuado y encontrar sus coeficientes, para que el sistema puede obtener la mejor estimación. En otras palabras, con los coeficientes apropiados el sistema puede minimizar el Error Cuadrático Medio.

$$\xi = E\{e(n)e^*(n)\} = E\{|e(n)|^2\} = E\left\{\left|d(n) - \sum_{l=0}^{p-1} w(l)x(n-l)\right|^2\right\} \quad (2.3)$$

Para encontrar el mínimo valor del Error Cuadrático Medio (ξ), se deriva la ecuación 2.3 respecto al transpuesto conjugado de $w(k)$ a cero, lo que da resultado a la ecuación 2.4.

$$E \left\{ e(n) \frac{\delta e^*(n)}{\delta w^*(k)} \right\} = 0 \quad (2.4)$$

Luego de haber derivado, obtenemos la ecuación 2.5 conocida como el "Principio de Ortogonalidad".

$$E\{e(n)x^*(n-k)\} = 0, k = 0,1,2, \dots, p-1 \quad (2.5)$$

De la ecuación 2.2 y la ecuación 2.5, conocemos:

$$E \left\{ \left[d(n) - \sum_{l=0}^{p-1} w(l)x(n-l) \right] x^*(n-k) \right\} = 0 \quad (2.6)$$

$$E\{d(n)x^*(n-k)\} - \sum_{l=0}^{p-1} w(l) E\{x(n-l)x^*(n-k)\} = 0$$

$$r_{dx} - \sum_{l=0}^{p-1} w(l)r_x(k-l) = 0$$

$$\sum_{l=0}^{p-1} w(l)r_x(k-l) = r_{dx}, k = 0,1,2, \dots, p-1 \quad (2.7)$$

Cabe mencionar que el resultado de una auto correlación (r_x) es una matriz Toeplitz, una matriz cuadrada cuyas diagonales son constantes.

La ecuación 2.7 es conocida como la ecuación de Wiener-Hopf.

$$wR_x = r_{dx} \quad (2.8)$$

Teniendo R_x y r_{dx} , se puede obtener directamente los coeficientes óptimos del filtro (w). Con los coeficientes óptimos de filtro se puede obtener el mínimo Error Cuadrático Medio.

De la ecuación 2.3, obtenemos lo siguiente:

$$\xi = E\{e(n)e^*(n)\} = E\left\{e(n)\left[d^*(n) - \sum_{l=0}^{p-1} w^*(l)x^*(n-l)\right]\right\}$$

$$\xi = E\left\{e(n)d^*(n) - \sum_{l=0}^{p-1} w^*(l)e(n)x^*(n-l)\right\} \quad (2.9)$$

Usando el principio de ortogonalidad (ecuación 2.5) en la ecuación 2.9, se obtiene la ecuación del mínimo Error Cuadrático Medio.

$$\xi_{min} = E\{e(n)d^*(n)\} = E\left\{\left[d(n) - \sum_{l=0}^{p-1} w(l)x(n-l)\right]d^*(n)\right\} \quad (2.10)$$

Donde el valor mínimo se encuentra en $n=0$, tal como se aprecia en la ecuación 2.11.

$$\xi_{min} = r_d(0) - \sum_{l=0}^{p-1} w(l)r_{dx}^*(l) \quad (2.11)$$

Si queremos usar la ecuación de Wiener-Hopf es necesario conocer dos condiciones dadas: una es la señal deseada $d(n)$ y la otra es la señal de entrada $x(n)$. Con estas señales se procede a buscar la auto correlación respectiva (R_x y R_d), y la correlación cruzada (r_{dx}). Asumiendo que ambas señales son procesos estacionarios en el sentido amplio (WSS).

Nuestro modelo de reconocimiento consta de 7 señales deseadas y una señal de entrada, todas en sus respectivos espectros de frecuencia normalizado. Cabe señalar que las señales deseadas o predefinidas se obtuvieron de siete grabaciones realizadas del estallido de un arma (Glock Automática) de calibre 9mm.

Las grabaciones cuentan con una duración de 200 ms cada una, y se clasifican por el entorno y distancia respecto al micrófono; en un espacio abierto obtenemos una señal de audio para cada distancia de 5, 10 y 40 metros, mientras en un lugar cerrado tenemos una señal de audio para 5, 10, 20 y 40 metros.

De acuerdo a lo visto con la ecuación de Wiener – Hopf, se procede a calcular los coeficientes del filtro para las siete señales predefinidas y se encuentra los valores promedios de ξ_{min} con respecto a los siete coeficientes del filtro.

Comparando los valores promedios del mínimo Error Cuadrático Medio (ξ_{min}), el sistema dará el juicio del cual de las señales referidas en el modelo se asimila más con la señal de entrada $x(n)$. Mientras más pequeño sea el valor promedio ξ_{min} , mejor estimación tenemos.

2.3 Software

2.3.1 Matlab

Actualmente Matlab es una de las mayores herramientas usadas a nivel investigativo, dado que provee un entorno matemático que permite desarrollar problemas algebraicos, hasta la extracción y manipulación de datos que provienen de una señal y poder realizar simulación, modelamiento matemático para fines educativos o empresariales.

Sin embargo, este software tuvo su origen a inicio de las décadas de los 90, cuando un programador pretendía resolver problemas matemáticos complejos usando lenguaje sencillo de programación, dado que en ese entonces existían otras herramientas para resolver la misma problemática, pero su grado de aprendizaje era más complejo que el usar Matlab.

Actualmente Matlab es empleado para entornos de desarrollo e investigativo, que son utilizados en diversas aplicaciones para el procesamiento de señales de audio, video a tal punto que puede simular

sistemas y modelos matemáticos. Además de que puede ser adaptable a diferentes sistemas embebidos para controlar módulos o sensores.

Siendo así Matlab cuenta con una gran diversidad de herramientas que me permiten procesar señales, dentro de cuales tenemos análisis de las mismas, hasta aplicación de filtro digitales y analógicos que incluyen respuestas de frecuencia, dominio hasta de fases; teniendo así diversos compiladores para los diversos sistemas operativos que se maneja hoy en día.

2.3.2 Simulink

Simulink es un entorno de diagrama de bloques para simulación multidominio y diseño basado en modelos. Es compatible con la simulación, generación automática de código, pruebas y verificaciones continuas de sistemas embebidos.

Simulink está integrado en MATLAB, dando acceso inmediato a una amplia gama de herramientas que le permiten desarrollar algoritmos, analizarlos y visualizar simulaciones, crear lotes de scripts para su procesamiento, personalizar el ambiente de modelado, definir señales, parámetros y datos de prueba. [9]

2.4 Hardware

2.4.1 Sistemas Embebidos

Hoy en día los sistemas embebidos son términos muy conocidos entorno al área electrónico, debido a que gran parte de los proyectos educativos e investigativos, por ende, estos sistemas están incluidos en aparatos o dispositivos que permiten controlar o manipular alguna acción de manera automática dependiendo de la funcionalidad del sistema y su respectiva programación.

Básicamente un sistema embebido es la combinación tanto en hardware como en software, convirtiéndolo en una pequeña estructura electrónica analógico-digital. Teniendo como esencia realizar actividades concretas

acorde a la necesidad del problema, además que estos tipos de sistemas se encuentran limitados por su capacidad.

Dado que los sistemas se encuentran restringidos por su capacidad de procesamiento acorde a la memoria que viene incluido en cada uno de los diferentes tipos de los mismos, estos tienen grandes ventajas de adaptarse diferentes tipos de módulos, sensores e inclusive se pueden adecuarse a otro tipo de sistemas embebidos, convirtiéndolo un sistema heterogéneo.

Actualmente a los sistemas heterogéneos embebidos podemos encontrarlo muy a menudo en la vida diaria, como por ejemplo una computadora, electrodomésticos, dispositivos móviles, sistemas de monitoreo de vigilancia y entre otros dispositivos electrónicos.

Dentro de las características de los sistemas embebidos se encuentran integrados en ellos diversos tipos de tecnología, seguridad y hasta una variedad en lenguajes de programación, que de tal manera esto hace que su costo puede variar en el mercado. Dado que su diseño esta implementado por un conjunto de procesadores que manejan sus respectivas interfaces de salidas/entradas y además permiten una manipulación más factible de los datos, esto variara de acuerdo al tipo de sistema que este manipulando.

2.4.2 Arduino

Dentro de las áreas investigativas y educativas se han producido diferentes prototipos electrónicos para el procesamiento de datos, además conforme a los avances tecnológicos se han desarrollado sistemas con plataforma Open Source como es el caso de Arduino.

Si bien es cierto esta plataforma de código abierto nació como una iniciativa hacia los estudiantes universitario para abaratar los costos de implementación de sus trabajos; además que la universidad que lo estaba desarrollando tenía un déficit económico, a tal punto que estaba yendo a la quiebra.

El primer prototipo estaba solo diseñado por una placa de circuito eléctrico, operado por un microcontrolador que podía trabajar con pequeños sensores, led y resistencias; pero conforme a la complejidad de los proyectos, los diseñadores de esta plataforma se vieron obligados a hacer más robusto y que tenga mayor soporte a diversos módulos y sensores que iban apareciendo.

Por lo tanto, el grupo de diseñadores fue creciendo paulatinamente por lo que entre la mejora del primer diseño se vio mejorado en diversas fases como el aumento de potencia para adaptarse a otros dispositivos, el entorno de programación del procesador y la interfaz del hardware convirtiéndolo en el primer prototipo que saldría a la venta para otras universidades.

Arduino se convirtió en una plataforma de prototipos electrónica de código abierto basada en hardware y software flexibles y fáciles de usar. Pensado para artistas, diseñadores y cualquier interesado en crear objetos o entornos interactivos. Dado que cuenta con una recepción de entradas desde una variedad de sensores y pudiendo controlar luces, motores y otros artefactos. [10]

El microcontrolador de la placa se programa usando el *Arduino Programming Language* (basado en Wiring) y el *Arduino Development Environment* (basado en Processing). Los proyectos de Arduino pueden ser autónomos o se pueden comunicar con software en ejecución en un ordenador. Permite que las placas se pueden ensamblar a mano o encargarse pre ensambladas; el software se puede descargar gratuitamente de la página oficial de Arduino. Los diseños de referencia del hardware (archivos CAD) están disponibles bajo licencia open-source, por lo que es libre de adaptarlas a las necesidades de cualquier proyecto. [10]

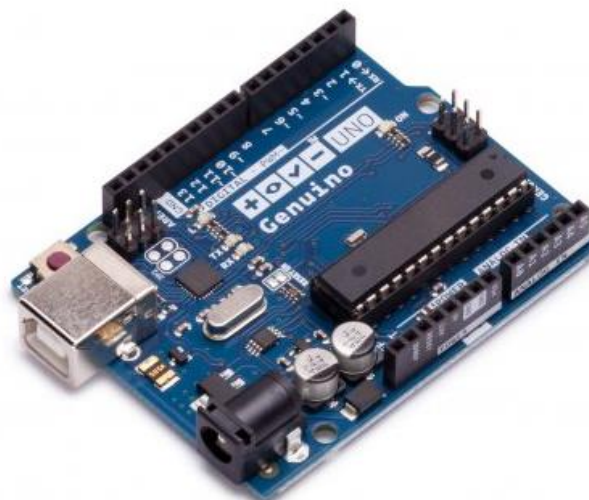


Figura 2.6: Arduino Genuino Uno

Tal como se observa en la figura 2.6, el Arduino Genuino Uno Rev3 es una placa electrónica basada en ATmega328P, con un microcontrolador de 8 bits con memoria y 2 KB de memoria RAM con 32 KB de flash. Conteniendo todo lo necesario para apoyar el microcontrolador; basta con conectarlo a un ordenador con un cable USB o la corriente con un adaptador de CA a CC o una batería para empezar. [11]. Para visualizar mejor sus características lo podemos ver en la Tabla 1 que se detalla a continuación.

Especificaciones Técnicas	
Microcontrolador	ATmega328
Voltaje operador	5V
Voltaje de entrada	7-12V
Voltaje de entrada (límite)	6-20V
Pines digitales I/O	14
Pines digitales PWM I/O	6
Pines de entrada analógica	6

Corrientes DC por Pin I/O	40 mA
Corrientes DC por Pin 3.3V	50 mA
Memoria Flash	32 KB
Memoria Flash desde Bootloader	0.5 KB
EEPROM	1 KB
Velocidad del Reloj	16 MHz

Tabla 1: Especificación técnica de Arduino Uno

2.4.3 Módulo VR3

Uno de las funcionalidades del proyecto consiste en reconocer palabras cuando un ciudadano este solicitando ayuda, para lo cual usaremos el módulo VR3. Este módulo de reconocimiento de voz facilitara las palabras claves que deseamos reconocer y ejecutara la acción programada de acuerdo a la palabra escuchada.

El módulo VR3 puede ser configurada de diferentes maneras, dado que es compatible con el módulo CP2303 y la placa Arduino Uno. De acuerdo a cuál módulo vaya a ser adaptado, su configuración puede variar entorno a su lenguaje de programación, dado que puede ser en código hexadecimal por el módulo CP2303 o usando lenguaje C para la placa Arduino.

Además, soporta hasta 80 comandos de voz en absoluto. Número máximo de comandos de voz 7 podrían trabajar al mismo tiempo. Cualquier sonido podría ser entrenado como comando, donde el usuario tiene que entrenar el módulo primero antes de que el reconocimiento realice cualquier acción con el comando de voz. [12]

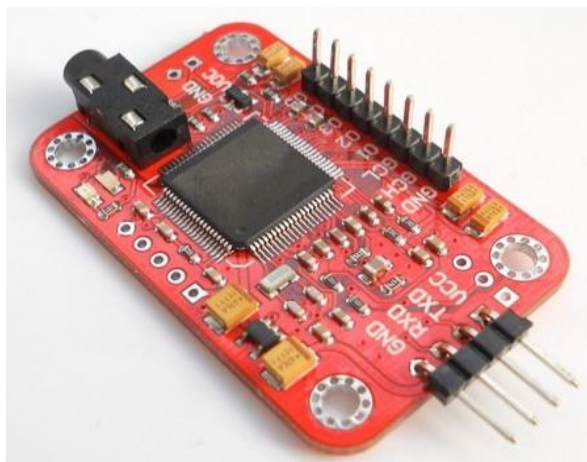


Figura 2.7: Modulo de reconocimiento de voz VR3

En la figura 2.7 se visualiza el diseño de este pequeño sistema micro embebido que tiene 2 formas de control: Puerto serie (función completa), los pines de entrada generales (parte de la función). Los pines de salida generan diversos tipos de ondas, mientras que este pequeño sistema reconozca por voz el comando correspondiente. [13]. Para tener una explicación más clara de cómo opera este módulo se puede apreciar a continuación la tabla 2.

Especificaciones Técnicas	
Voltaje de Alimentación	4.5 – 5.5 V
Corriente	< 40 mA
Interfaz Digital TTL	5 V
Interfaz Análoga	3.5 mm para micrófono
Dimensión	30 mm x 47.5 mm
Precisión de Reconocimiento	99% bajo entorno ruidos reducidos

Tabla 2: Especificación técnica del módulo VR3

2.4.4 Raspberry Pi 3 Modelo B

Es una popular placa de computadora del tamaño de una tarjeta de crédito que soporta sistemas operativos integrados Linux, como Raspbian. Este pequeño ordenador es desarrollado en Reino Unido por la fundación Raspberry Pi, con la intención de fomentar la enseñanza de ciencias computacionales en las escuelas, visualice la gráfica 2.8.

Es la tercera generación de Raspberry Pi. Comparado con sus predecesores, ofrece las siguientes funciones:

- A 1.2GHz 64-bit quad-core ARMv8 CPU
- 802.11n Wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)

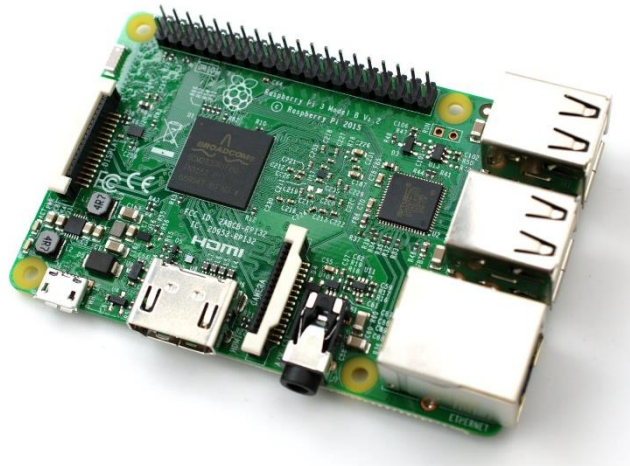


Figura 2.8: Raspberry Pi 3 Modelo B

Esta tarjeta tiene la capacidad de asimilar las funciones básicas que haría una computadora de escritorio. Aparte incorpora funcionalidades como servidor web, bases de datos, entre otras.

De uso escolar o general, la Raspberry Pi puede actuar también como un sistema embebido para el procesamiento de señales en tiempo real. El paquete de soporte de Simulink para Raspberry Pi, te permite desarrollar algoritmos en Simulink y desplegar a al hardware mediante generación

automática de código. A continuación, el procesamiento es realizado de forma autónoma en la Raspberry Pi.

Para el proyecto, esta placa cumple con el rol de detectar un disparo de arma de fuego mediante el procesamiento de la señal de audio.

2.4.5 Tarjeta de Sonido Estéreo USB Externa

Sabemos que la tarjeta de sonido es un dispositivo que controla la salida y entrada de audio, y que se conecta a la placa madre de la computadora o puede ir integrada en la misma. La Raspberry Pi cuenta con una tarjeta de sonido integrada en la placa que habilita la salida de audio, pero no una entrada, por lo que se vio forzado la adquisición de una tarjeta de sonido estéreo USB externa.



Figura 2.9: Tarjeta de sonido Estéreo USB Externa

Tal como se observa en la figura 2.9, este adaptador convierte un puerto USB en un puerto de audio habilitado para la conexión de auriculares y audífonos. Su uso es esencial en nuestro proyecto para la adquisición de datos, propicio para el procesamiento de señal en el reconocimiento de un disparo con arma de fuego.

Compatible con la mayoría de versiones de los sistemas operativos Windows, Linux y Mac, el adaptador se integra en prácticamente cualquier entorno de ordenador y ofrece una gran calidad de sonido y uso práctico [14].

CAPÍTULO 3

3 IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

3.1 Arquitectura del sonido

Para el desarrollo de este primer nodo sensor o prototipo se usará en gran parte tecnologías open source, para abaratar costos en mantenimiento de hardware, permitiendo así resolver dudas con comunidades que trabajan bajo estos entornos; evitando así obtener licencias por los softwares propietario que limitan el desarrollo.

Mediante la combinación de software Python y Simulink de Matlab en conjunto con los sistemas embebidos como Raspberry Pi, Arduino Uno y el Módulo VR3 se desarrollaron scripts y modelos para el procesamiento de señales que permita reconocer el sonido de un disparo e identificar la solicitud de ayuda que es receptada por este sistema.

El sistema de identificación de sonido está diseñado tal como se muestra en la siguiente Figura 3.1.

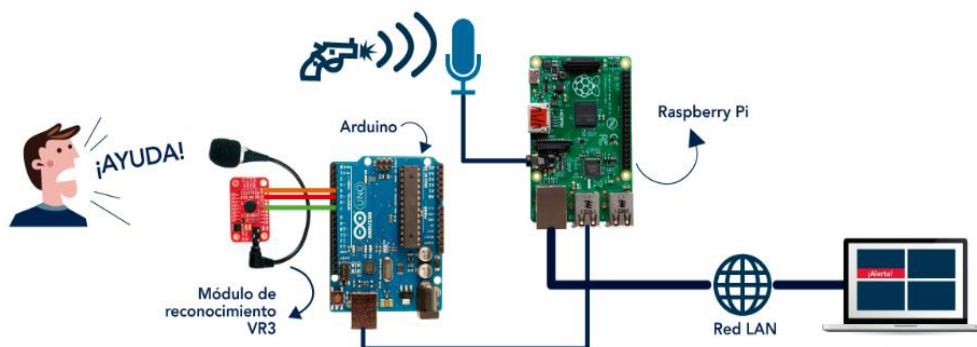


Figura 3.1 Sistema de identificación

3.2 Desarrollo del nodo detector de sonido

El proyecto está dividido en tres fases, en la que el nodo auxiliar estará receptando los datos, procesando las señales y por último la comunicación con

el centro de monitoreo, para una explicación más detallada se describe a continuación.

3.2.1 Primera Fase

El prototipo del nodo estará recibiendo los datos por los micrófonos conectados al módulo VR3 y a la Raspberry Pi; en esta fase el Raspberry Pi estará suministrando la energía necesaria para el Arduino y su respectivo módulo. El Raspberry Pi puede conectarse directamente a la toma de corriente con su respectivo cargador o fuente extra que le proporcione el suficiente voltaje para su funcionamiento.

Una vez que este energizado el sistema del Raspberry, este podrá alimentar al Arduino, para que así los módulos que trabajan con este puedan ser suministrados con el voltaje y la corriente necesaria. Dado que el módulo que detectará las palabras suministradas utilizará una librería de Arduino para su recepción de datos y así el Raspberry Pi puede enviar los datos hacia el centro de operación remoto.

El módulo para la detección de solicitudes de ayuda es conocido como Voice Recognition Module 3, dado que permite reconocer palabras de acuerdo a su configuración y entrenamiento que recibe previamente para aprender las palabras. Por lo tanto, el módulo VR3 estará conectado al Arduino Uno quien recepta las palabras enviadas por el módulo.

Por ende, el Arduino detecta las palabras reconocidas por el módulo VR3, por lo que este pequeño sistema ejecutará la acción que se le programe por lo detectado en el módulo; por lo que el Arduino realizará la función de retransmitir la información recibida hacia el Raspberry Pi, quien tendrá alojada una pequeña base datos y lo almacenará con su respectiva fecha y hora.

3.2.2 Segunda Fase

Esta fase consiste en el desarrollo e implementación del sistema para el reconocimiento de un disparo con arma de fuego.

La siguiente figura 3.2 muestra la configuración del módulo de reconocimiento y algunos dispositivos y paquetes requeridos:

- Hardware: Un micrófono electret omnidireccional y una tarjeta de sonido estéreo USB externa
- Software: DSP System Toolbox y Simulink Support Package for Raspberry Pi en Matlab/Simulink

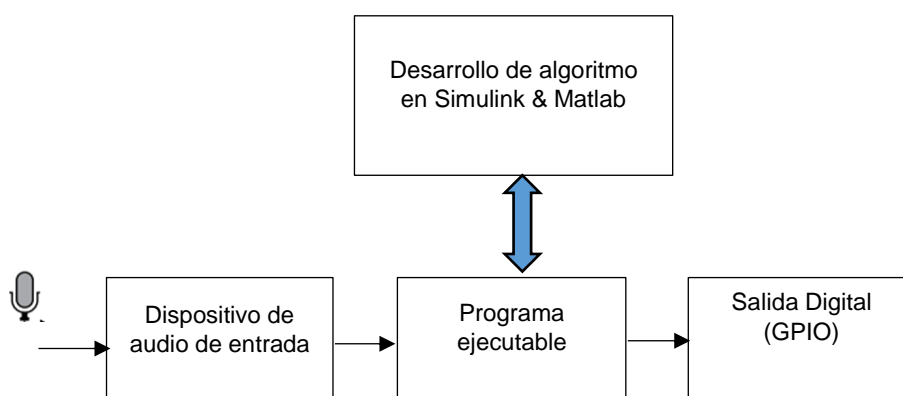


Figura 3.2: Configuración del módulo de reconocimiento de disparo

Como se muestra en la figura 3.2, el micrófono está conectado con la tarjeta de sonido y el sonido es convertido en una señal de datos muestreado. La señal de audio digital tiene una resolución de 16 bits y una frecuencia de muestreo de 16000 Hz. Los datos del audio son representados como una matriz en Matlab y un frame en Simulink. El algoritmo de procesamiento de datos es desarrollado en el entorno Simulink/Matlab y descargado en la Raspberry Pi como un programa ejecutable.

La definición de la frecuencia de muestreo se basó en los siguientes parámetros como la respuesta de frecuencia del micrófono, el procesador del Raspberry Pi y la latencia del sistema. No fue posible encontrar la respuesta de frecuencia del micrófono debido a que se desconocía el nombre del fabricante, pero gracias al gráfico de la figura 2.4, podemos apreciar que la potencia parece atenuarse desde la frecuencia de 10 KHz

en adelante. Por lo que, en primera vista, la frecuencia de muestreo ideal sería 22050 Hz (Teorema de Nyquist). Si el sistema de reconocimiento hace la comparación a una señal digital de audio de una duración de 0.2 segundos, esto equivaldría un buffer de tamaño 4410 ($0.2 \cdot 22050$). Si tomamos en cuenta la capacidad de procesamiento del Raspberry mas la latencia del sistema (algoritmo de reconocimiento) lo mejor sería disminuir el tamaño del buffer. Por consiguiente, la frecuencia de muestreo es de 16000 ($0.2 \cdot 16000$ muestras) discriminando así sonidos con frecuencias superiores a 8000 Hz.

Como se muestra en la siguiente figura 3.3, se ha desarrollado un modelo de Simulink para el reconocimiento de disparo, por medio del sonido que genera el estallido de un arma. El bloque ALSA Audio Capture que está configurado con una frecuencia de muestreo de 16000 Hz, envía los datos de audio estéreo en un frame (dimensión 16×2) de enteros de 16 bits en un intervalo de 0,001 segundo. Para el procesamiento digital se debe convertir los datos capturados de dos canales de audio (una matriz de enteros 16×2) en datos de un solo canal (un vector tipo double de 16×1).

El resto del modelo consta de los siguientes bloques de función:

1. El bloque del buffer está configurado con un tamaño de 3200. Su función consiste en guardar un frame de 0.2 segundos de datos del audio.
2. El grupo de tres bloques trabajan como un detector de picos y segmentación. Se mide que la amplitud máxima de cada frame supere el valor de 0.8 para considerar la posible presencia de un sonido generado por el disparo con el arma. Este ajuste depende de múltiples factores, incluyendo la configuración de la ganancia del micrófono, la distancia esperada de la fuente de sonido al micrófono y el ruido de fondo esperado. Por ejemplo, si uno habla fuerte y muy cerca del micrófono, el sistema lo valida como un posible estallido del arma. Una vez que un sonido válido es

detectado, un impulso ascendente se genera y habilita el bloque Triggered Subsystem después de un retraso de 198 frames (198*0.001 segundos), con el propósito de guardar en el buffer solo los datos del sonido que genera el disparo.

3. El bloque Spectrogram se encarga de encontrar el espectro de frecuencia normalizado de la señal de datos alojada en el buffer, mediante el uso de espectrograma sobre la señal. Para ser más claro, esto se explica en la parte 2.2.2.
4. Estos bloques son constantes que contiene los valores de siete espectros de frecuencia normalizados denominados como señales deseadas. Cada señal deseada tiene un id que la diferencia de las demás.
5. Este bloque contiene el algoritmo del filtro de Wiener, explicado detalladamente en la sección 2.2.3. Sus entradas son la señal deseada y la señal de salida que da el bloque número 3.
6. El bloque Comparador de Error da el juicio del cual de las señales referidas en el modelo se asimila más con la señal de datos alojada en el buffer.
7. Luego de haber elegido la señal deseada el bloque 6, se procede a guardar en la base de datos: el id de la señal deseada, el valor promedio del ECM (Error Cuadrático Medio), el máximo del ECM y el espectro de frecuencia normalizado de la señal de datos del buffer.

Sistema de reconocimiento de disparos

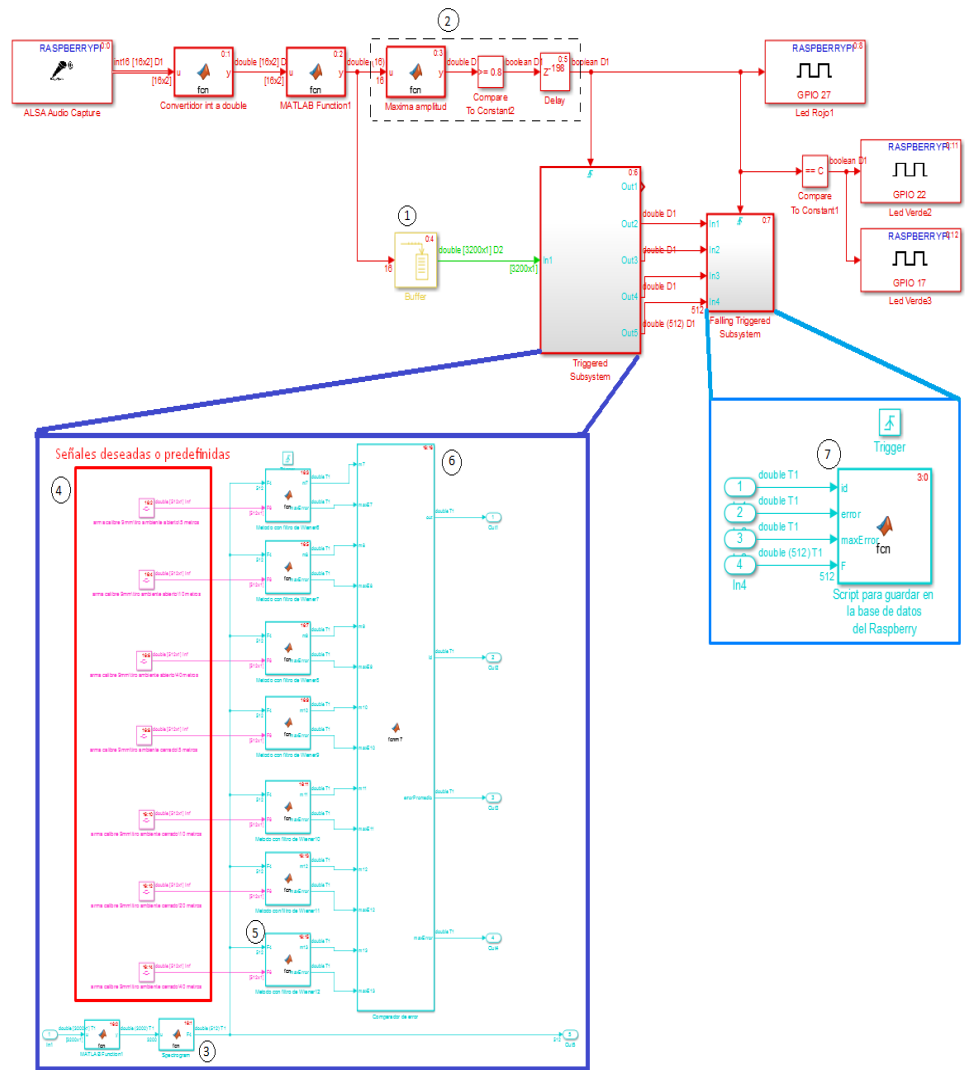


Figura 3.3: Modelo en Simulink

Cuando el modelo empieza a ejecutarse en el Raspberry, se enciende los dos leds color verde conectado en los pines GPIO 22 y GPIO 17 tal como se aprecia en la figura 3.4. Mientras que led rojo en el pin GPIO 27 al encenderse, indicará que se guardó en la base de datos los resultados del algoritmo de reconocimiento de disparo como se observa en la figura 3.5.



Figura 3.4: El modelo se encuentra en ejecución



Figura 3.5: Led Rojo indica que se guarda en base de datos

3.2.3 Tercera Fase

La comunicación entre el nodo auxiliar y el centro de monitoreo se realizará mediante una sincronización de base de datos que existirá entre estos dos sistemas. Debido a que ambos alojarán información en sus respectivas bases de datos con respecto a las solicitudes de ayuda y características de proyectil.

El nodo auxiliar tendrá una base de datos donde alojará la información ya procesada en caso de presentarse un evento y este sistema según el evento haya dado como afirmativo a la alarma producida por el mismo.

Por lo que su base de datos que tendrá alojada será configurada en modo maestro.

Mientras que el centro de monitoreo tendrá alojada otra base de dato con características similares en cuanto a la base del nodo, y esta guardará los registros sucedidos con sus respectivas fechas para llevar una bitácora de los eventos sucedidos. Cabe destacar que la base de datos en el centro de monitoreo será configurada en modo esclavo.

Por lo que relevante que ambos estén sincronizados, para que la base de datos en modo esclavo solo pueda consultar la información necesaria a la base datos en modo maestra; y así este hacer su respectiva consulta de registro e informar de manera inmediata al operador remoto.

3.3 Funcionamiento del nodo

3.3.1 Energización del sistema

El sistema principal que representa al Raspberry en conjunto con el Arduino y el Módulo VR3, representaría al nodo sensor que estaría trabajando en conjunto con el sistema de seguridad; donde todo sistema estará energizado con un voltaje de 5 V.

El Raspberry va a estar energizado por un cargador externo que le proporcionará el voltaje apropiado según las especificaciones técnicas, así será alimentado para su operación necesaria. Dado que este cuenta con puertos USB, solo será usado un puerto USB para la energización del Arduino; como el Arduino cuenta con pines de alimentación, será suficiente conectar los pines de 5 V y GND con los pines del Módulo VR3 para que todo este energizado.

3.3.2 Recepción de señales

Para la recepción de señales como el disparo y la solicitud de ayuda usaremos micrófonos omnidireccionales, cabe aclarar que vamos a disponer de dos micrófonos: uno para el Raspberry que receptará la señal producido por un disparo y este hará su debido procesamiento, mientras que el otro será usado para conectarlo en el módulo VR3 que detectará la

solicitud de ayuda e informará inmediatamente al Arduino que palabra detectó.

3.3.3 Transmisión de datos

El módulo VR3 transmitirá el dato según la palabra percibida por el micrófono al Arduino Uno, mediante una comunicación UART (Rx y Tx); por lo que Arduino receptorá los datos para efectuar la operación deseada y envío de datos mediante el puerto USB para la transmisión de datos al Raspberry, quien tendrá alojada una base de datos en modo maestro. Dado que la Raspberry cuenta con una tarjeta de red inalámbrica, esta tendrá que estar en la misma red con el centro de operación remoto. Convirtiendo así al Raspberry en nodo transmisor.

3.3.4 Recepción de datos

Dado que el Raspberry se convierte en nodo transmisor, el centro de operaciones remoto será el receptor, para lo cual el monitoreo lo simularemos mediante un servidor quien va a recibir los datos alojados en la base de datos Raspberry y lo sincronizara con la base de datos alojado en el servidor, para lo cual este va a estar configurado en modo esclavo, permitiendo así consultar los datos que se almacena en el nodo.

3.3.5 Centro de monitoreo

El centro de monitoreo va a ser el servidor, para lo cual va a tener configurado y activado los servicios de Web (Apache y PHP), DNS (bind9) y base de datos (Mysql). Dado que el servidor alojara el sistema operativo Debian Jessie que es una distribución libre de Linux.

Por lo cual el centro de monitoreo puede tener conectado varios nodos, para que estos estén en constante sincronización con la base de datos.

Dado que el nodo y el centro de operaciones remoto están sincronizado, el servidor estará sincronizándose constantemente con el nodo, para determinar si detecto algún disparo o sonido similar o si detecto alguna solicitud de ayuda. Para lo cual el servidor va a estar consultando las

tablas que van a estar replicados por el nodo y copiadas en el servidor, obteniendo así la información más reciente por el suceso suscitado.

Para la lo cual nuestro centro de monitoreo va a ser una página web desarrollada con PHP y alojada en un servidor web Apache, para más seguridad se proporcionará el servicio DNS al servidor, permitiendo direccionar la dirección IP con la página. Como va ser solo de monitoreo, nos mostrará los valores enviados por el nodo que proceso la señal en la página y será el operador remoto conectándose desde otro equipo en modo cliente con la URL o dirección para visualizar sus resultados.

3.4 Instalación y Configuración

Para probar el funcionamiento del nodo detector de sonidos, se instaló el nodo en un lugar remoto del centro de operaciones para tomar las señales como el disparo y solicitudes de ayuda. Destacando que debe haber comunicación entre el nodo y el servidor.

El modelo creado en Simulink de Matlab debe ser cargado en el Raspberry para la recepción de señales de disparo, y a la vez se desarrolló y ejecuto un script para recibir los datos que trasmite el Arduino por el puerto de USB; recordando que los datos que recolecta el Arduino y el modelo de Simulink guardara los datos en tablas diferentes de la base de datos en modo master.

Además, revisar si la base de datos del servidor está configurada en modo esclavo, para que se sincronice las tablas de datos que almacena la información en el Raspberry; y así mismo la página web pueda consultar la información de su base de datos.

3.5 Pruebas y Soluciones

Una de las limitaciones que presentó el proyecto es la dificultad de conseguir un arma de fuego, debido que es ilegal portarlas sin un permiso. Por lo que solicitamos ayuda al club Rivera del Lago, un club de tiro donde se realiza práctica de deportes relacionados con el manejo profesional de armas.

Por lo cual, en la primera visita, la organización accedió ayudarnos y brindarnos todas las facilidades para las grabaciones iniciales en la que sacaríamos las

señales deseadas para el sistema de reconocimiento de disparo. Por consiguiente, en la segunda visita a las instalaciones de las mismas, se procedió a realizar las pruebas en cuanto a la primera versión del modelo de Simulink. Los resultados no fueron favorables y se observó un consumo excesivo de CPU que provocaba la ejecución del programa. Las posibles causas de los errores presenciado ese día fue el uso de una frecuencia de muestreo de 44100 Hz, una base grande de señales predefinidas y una segmentación incorrecta.

Para solucionar los errores suscitados, los cambios realizados al modelo fueron:

- La nueva frecuencia de muestreo a 16000 Hz
- Cambio en la base de datos de señales predefinidas (quitar de los registros de reconocimiento la escopeta de calibre 22mm dejando solamente el arma Glock automática 9mm)
- Corrección de la segmentación.

El cambio de frecuencia de muestreo implicaba que las señales deseadas debían ser cambiadas; por consiguiente, debimos ir por tercera vez al club para hacer las grabaciones, ahora con una frecuencia de 16 KHz.

```

pi@raspberrypi: ~
top - 12:11:05 up 53 min, 3 users, load average: 1.93, 1.05, 0.69
Tasks: 134 total, 2 running, 132 sleeping, 0 stopped, 0 zombie
%Cpu(s): 25.0 us, 0.2 sy, 0.0 ni, 74.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem:  947748 total,  316968 used,  630780 free,  11888 buffers
KiB Swap: 102396 total,  0 used,  102396 free.  86300 cached Mem

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  MEM%   TIME+  COMMAND
 3516 root      rt   0 512448 142564 5056 S 100.2 15.0  1:53.04 proyecto_v4_pru
 3358 pi        20   0   5112   2492 2088 R   0.3  0.3   0:03.19 top
    1 root      20   0  23864   3992 2736 S   0.0  0.4   0:04.54 systemd
    2 root      20   0     0     0     0 S   0.0  0.0   0:00.00 kthreadd
    3 root      20   0     0     0     0 S   0.0  0.0   0:00.37 ksoftirqd/0
    5 root      0 -20     0     0     0 S   0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0     0     0     0 S   0.0  0.0   0:00.33 rcu_sched
    8 root      20   0     0     0     0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0     0     0     0 S   0.0  0.0   0:00.01 migration/0

```

Figura 3.6: Consumo de CPU en la primera prueba de campo

```

pi@raspberrypi: ~
top - 02:56:30 up 3:28, 3 users, load average: 0.02, 0.01, 0.00
Tasks: 128 total, 1 running, 127 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.9 us, 0.5 sy, 0.0 ni, 98.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 947748 total, 171128 used, 776620 free, 9456 buffers
KiB Swap: 102396 total, 0 used, 102396 free. 69140 cached Mem

  PID USER  PR  NI  VIRT  RES  SHR S  %CPU  %MEM    TIME+  COMMAND
 1889 root   rt    0 52684 21668 4456 S   11.5   2.3   0:16.00 proyecto_v4_arm
 1935 pi     20    0  5252   2500  2140 R    1.8   0.3   0:00.92 top
     1 root   20    0 22904  4016  2788 S    0.0   0.4   0:04.47 systemd
     2 root   20    0     0     0     0 S    0.0   0.0   0:00.00 kthreadd
     3 root   20    0     0     0     0 S    0.0   0.0   0:00.02 ksoftirqd/0
     5 root    0 -20    0     0     0 S    0.0   0.0   0:00.00 kworker/0:0H
     7 root   20    0     0     0     0 S    0.0   0.0   0:00.25 rcu_sched
     8 root   20    0     0     0     0 S    0.0   0.0   0:00.00 rcu_bh
     9 root   rt    0     0     0     0 S    0.0   0.0   0:00.01 migration/0
    10 root   rt    0     0     0     0 S    0.0   0.0   0:00.00 migration/1
    11 root   20    0     0     0     0 S    0.0   0.0   0:00.25 ksoftirqd/1
    13 root    0 -20    0     0     0 S    0.0   0.0   0:00.00 kworker/1:0H

```

Figura 3.7: Consumo del CPU con los cambios al modelo Simulink

Con los cambios efectuados, se presenta una mejora en el rendimiento del sistema. Tal como se puede comparar la figura 3.6 con respecto a la figura 3.7. Por lo que el tiempo de demora en el reconocimiento o comparación de sonido es aproximadamente de 7 segundos, y en ese lapso de tiempo el consumo de CPU llega al 100% por un instante y luego baja a su porcentaje inicial.

Observando las mejoras en el módulo, se decide ir por cuarta vez al club para realizar las pruebas finales, pero recibimos la negativa por parte de ellos, argumentando que había un límite de visitas para personas no socias. Aparte el presupuesto con que contábamos para el proyecto no era suficiente para asistir a un polígono privado de tiro. Sin más remedio, tuvimos que optar por simulaciones para realizar las pruebas y analizar los resultados posteriormente.

Las simulaciones se hacen en Matlab y Simulink con una grabación de audio que se hizo con una frecuencia de muestreo de 16000 Hz, y a una distancia de 40 metros entre la fuente del sonido y el micrófono. La señal deseada para el análisis es de un disparo a 40 metros del micrófono como se muestra en la gráfica 3.8. Cabe recordar que el arma que se usó en la grabación es un Glock automática 9mm, ya que se modeló un sistema para el reconocimiento de solo esa arma.

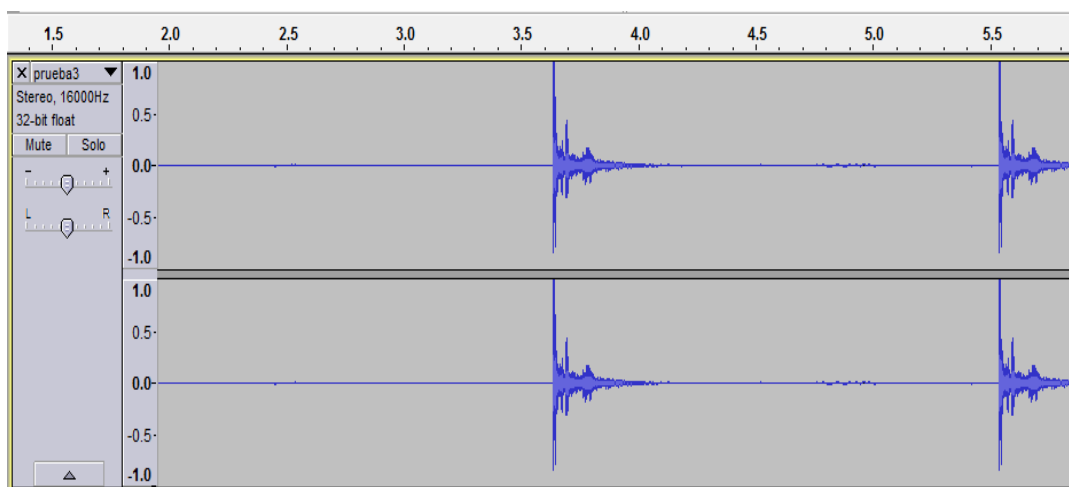


Figura 3.8: Grabación de disparos con una arma calibre 9mm, a 40 metros del micrófono

La simulación en Matlab solo requiere el fragmento de 0.2 segundos del audio en el que suena el disparo, tal como se muestra la figura 3.9.

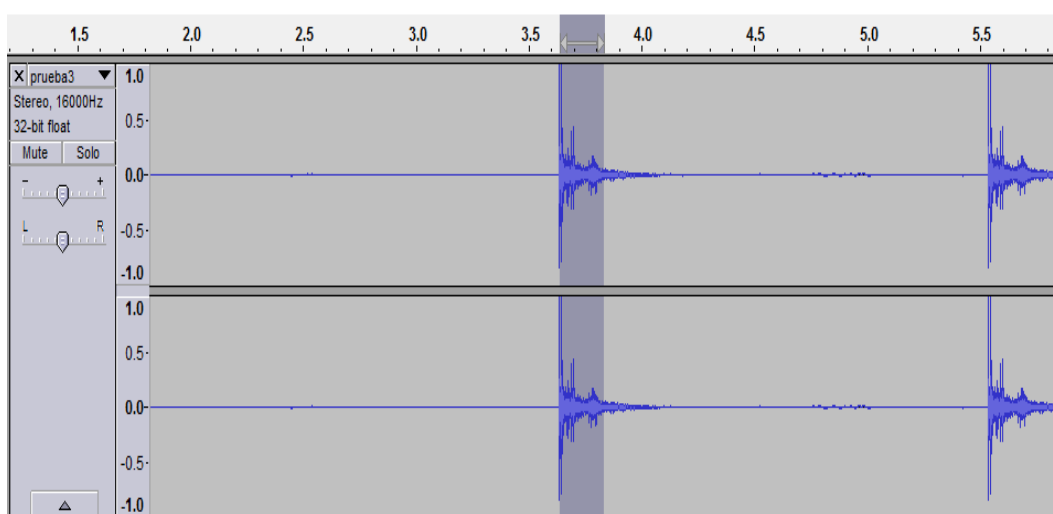


Figura 3.9: Fragmento de 0.2 segundos

3.6 Resultados Obtenidos

3.6.1 Simulaciones en Matlab

Obtenemos el espectro de frecuencia normalizado de la señal digital del audio como se aprecia en la figura 3.10.

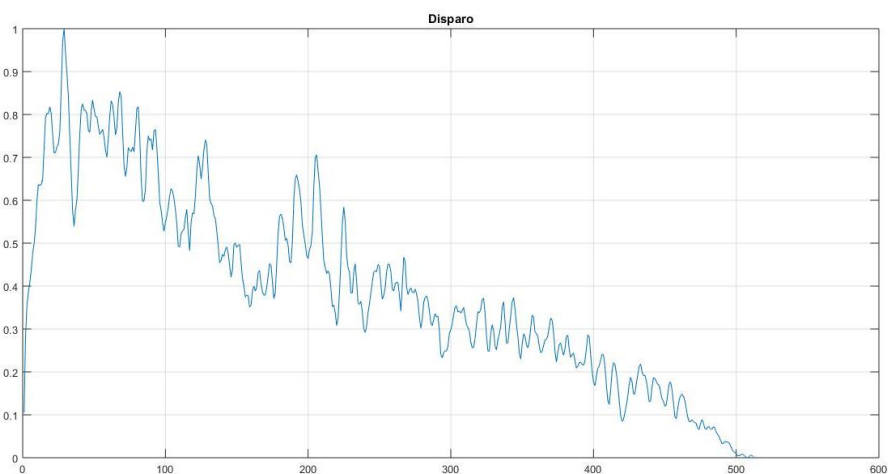


Figura 3.10: Espectro de frecuencia de la señal digital del audio

Comparamos con la siguiente gráfica 3.11 que contiene el espectro de la señal deseada. Visualmente podemos reconocer que existe una gran similitud.

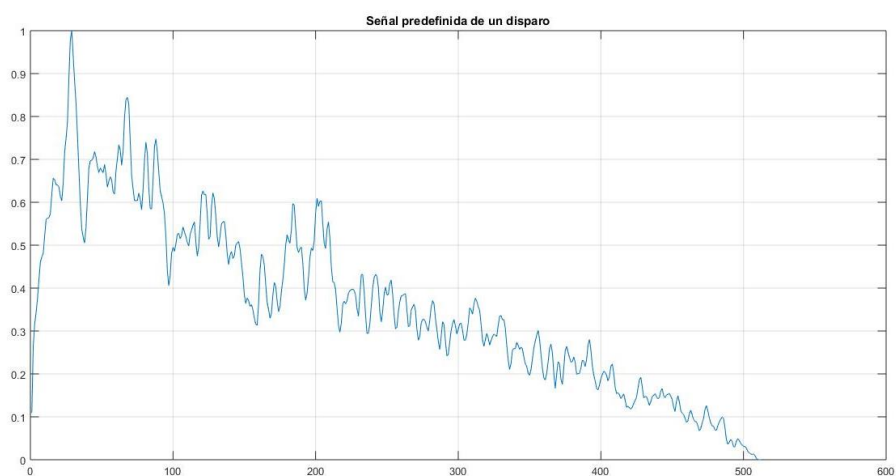


Figura 3.11: Espectro de frecuencia de la señal deseada

Se procede a aplicar el algoritmo básico del filtro de Wiener, el resultado es la gráfica 3.12 que se muestra a continuación.

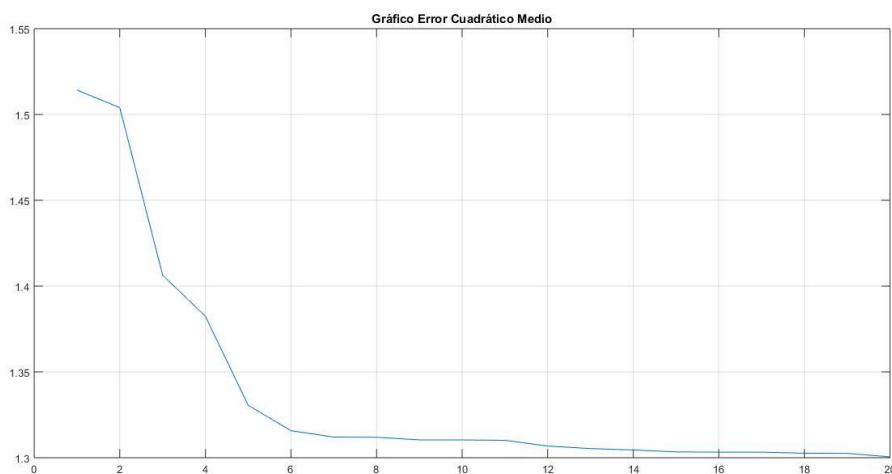


Figura 3.12: Gráfico de Error Cuadrático Medio

El valor promedio del Error Cuadrático Medio, de la figura 3.12, es 1.3370.

A continuación, aplicaremos el algoritmo de reconocimiento a la grabación de sonido de un aplauso cuya duración es de 0.1 segundos. Mantenemos la señal deseada de un disparo a 40 metros del micrófono, pero para este escenario su duración debe ser de 0.1 segundos.

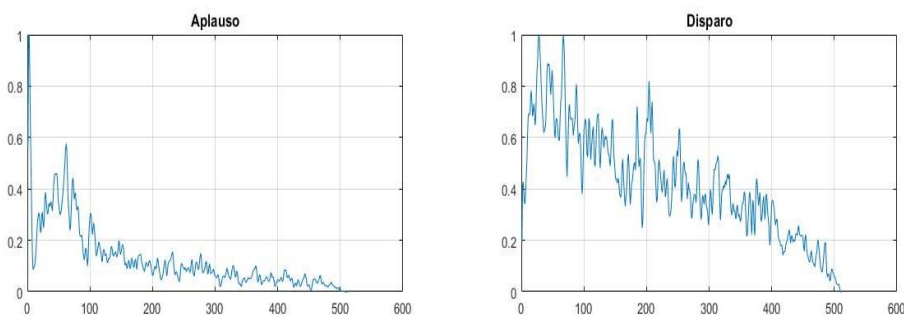


Figura 3.13: Espectros de frecuencia

Podemos apreciar en la figura 3.13, que los espectros de ambas señales son diferentes y lo podemos corroborar con el gráfico del Error Cuadrático Medio.

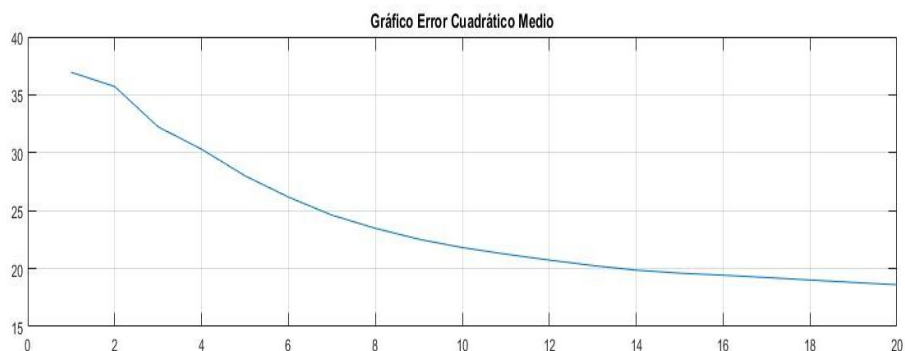


Figura 3.14: Gráfico de Error Cuadrático Medio

El valor promedio del ECM es 23.9299.

Analizando los resultados obtenidos podemos afirmar que mientras más pequeño sea el valor del Error Cuadrático Medio, más se asemejará la señal de entrada con la señal estimada como base; lo que nos permitirá identificar el parentesco entre los patrones de ambas señales.

3.6.2 Simulación en Simulink

Para esta simulación, adaptamos el modelo Simulink que se muestra en la figura 3.15 para que ahora procese los datos de un archivo de audio reemplazando el audio de entrada por el micrófono. Debemos aclarar que el modelo se ejecuta esta vez en un pc, mas no en el Raspberry Pi.

Sistema de reconocimiento de disparos

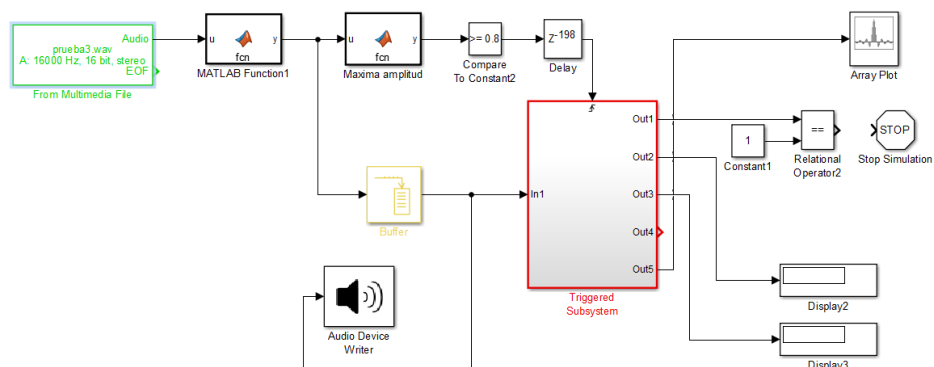


Figura 3.15: Modelo para pruebas en pc

Recordemos que la grabación está representada en la figura 3.8. Se puede apreciar que hubo dos disparos, uno en el segundo 3.3 y el otro en el segundo 5.5 aproximadamente. Por lo tanto, se espera que la simulación nos arroje dos resultados.

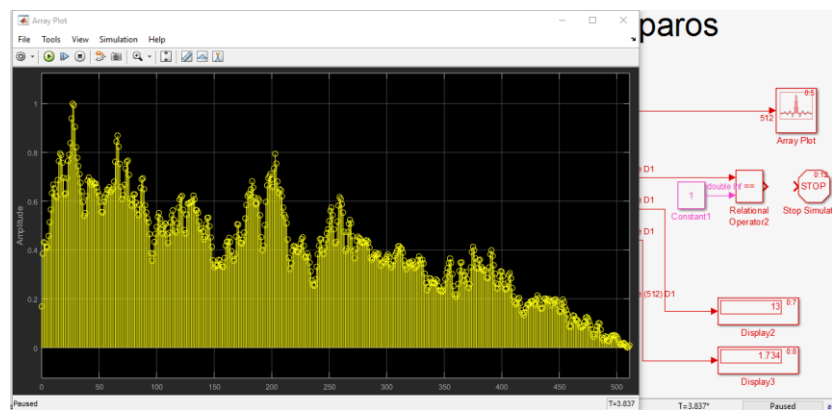


Figura 3.16: Detección del primer disparo

Tanto en la figura 3.16 como la 3.17, nos fijaremos en el valor que muestra el Display 3 ya que este indica el mínimo valor promedio del Error Cuadrático Medio que se obtiene de las siete señales deseadas.

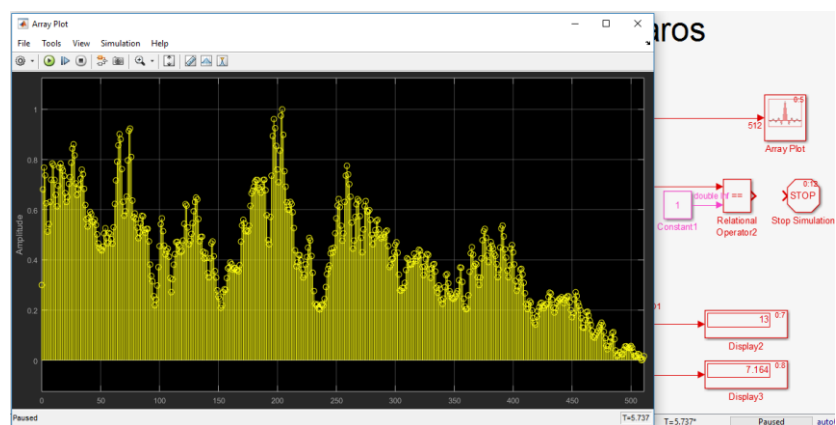


Figura 3.17: Detección del segundo disparo

Los resultados fueron: un valor promedio de ECM de 1.734 para el primer disparo y 7.164 para el segundo disparo.

Analizando el resultado de la simulación en Matlab, destacamos que los resultados en Simulink son diferentes; debido a que se está tratando el procesamiento digital de señales en tiempo real, donde el método de

segmentación implementado influye mucho en el resultado del sistema, por lo que la señal de datos que entrega la segmentación puede contener información no deseada o puede faltar información esencial.

3.7 Costos de implementación

En la tabla 3 se detalla el costo de cada implemento utilizado para la implementación del nodo detector de señales:

Equipo	Cantidad	Costo Total
Raspberry Pi Model 3	1	\$70,00
Arduino Uno	1	\$50,00
Módulo VR3	2	\$94,00
Materiales para el nodo	1	\$10,00
Costo Total		\$224,00

Tabla 3: Costo de Implementar el nodo

CONCLUSIONES Y RECOMENDACIONES

Se logró implementar un primer prototipo del nodo para la detección de señales, utilizando en gran parte tecnología open source y otra parte software propietario como Matlab para el procesamiento de la señal.

Previamente se identificó las palabras usadas comúnmente por parte de un ciudadano cuando este requiera de ayuda y los tipos de armas de fuego más usados para los actos ilícitos. Se define la base de datos de sonido que contiene palabras de socorro y disparos de solo una arma de fuego (para mejorar el rendimiento del sistema se tuvo que eliminar el uso de otras armas, dejando solamente el arma Glock automática 9mm).

Se permitió almacenar la información como solicitudes de ayuda e información que produce un impacto de proyectil, para registrar el evento y presentarlo de inmediato al operador remoto. Debido a la falta de pruebas, el módulo de reconocimiento de disparo no está habilitado para decidir si el sonido captado es proveniente o no del disparo, por lo que se enviará al servidor los resultados provenientes del algoritmo de reconocimiento. Cualquier sonido que al digitalizarse contenga al menos una muestra mayor a 0.8 (valor parametrizable) será procesado por el sistema y posteriormente se mostrará por medio de interfaz gráfica los resultados al usuario quien deberá confirmar si se trata de una falsa alarma.

En la práctica, para un mejor rendimiento del reconocimiento, tanto en solicitudes de ayuda como disparos, el programa de reconocimiento tiene que ser optimizado para el ambiente y configuraciones. Por ejemplo, los niveles de ruido de fondo son diferentes en distintos escenarios de la aplicación. Diferentes micrófonos tienen diversos rendimientos de captura de audio, lo que puede dar lugar a diversos niveles de voz (en el reconocimiento de solicitudes de auxilio).

La efectividad del módulo de reconocimiento de voz para las solicitudes de ayuda va a depender altamente del entrenamiento que se hace al sistema para mitigar los efectos causados por factores externos.

Dado que el proyecto es solo un prototipo, se ve limitado en cuanto al procesamiento de la señal; así que tendrá retardos en su procesamiento, como consultar y verificar con las señales predefinidas en la base de datos. Si se desea mejorar y optimizar el tiempo de retardo para el procesamiento de la señal, se debe utilizar sistemas embebidos más eficientes y robustos en cuanto a su capacidad, dado que el que usamos se ve limitado por ser solo el primer prototipo.

En el modelo de Simulink vale la pena considerar el ajuste de los siguientes parámetros: la cantidad de retardo en la segmentación del sonido y el umbral de la toma de decisiones. La cantidad de retardo en la segmentación del sonido es un parámetro que afecta a la cantidad de muestras antes y después de que se utilice el cruce del umbral detectado por el reconocimiento de disparo. El umbral de la toma de decisiones, que es el interruptor del subsistema de reconocimiento, tiene un impacto sobre la tasa de falso reconocimiento y la tasa de rechazo. En la mayoría de los casos, es difícil de encontrar un umbral adecuado y obtener una buena compensación entre el falso reconocimiento y la pérdida de datos para el reconocimiento.

Cabe destacar que se puede agregar un conjunto de palabras e ir asignando a cada uno a un índice dentro del módulo VR3, para que así el nodo pueda detectar más palabras, previamente se debe entrenar al módulo VR3 para su posterior aprendizaje.

BIBLIOGRAFÍA

- [1] Locating Surveillance and Target Acquisition Association. The Eyes and Ears of the Battlefield [Online].
Disponibile en: <http://www.locatingartillery.org/overview.htm>
- [2] Defense Industry Daily. (2011, Febrero 14). Defense program acquisition news, budget data, market briefings [Online].
Disponibile en: <http://www.defenseindustrydaily.com/sniping-at-us-forces-beginning-to-boomerang-01128/#incoming-fire-detection>
- [3] El Univero. (2006, Septiembre 3). Ojos de águila, videos usados como pruebas judiciales [Online]. Disponible en:
<http://www.eluniverso.com/2006/09/03/0001/10/1AD5DAA087BC42A398C043CC6DD9BCF0.html>
- [4] El Telégrafo. (2016, Mayo 12). Los Esteros es una de las zonas más violentas de Guayaquil Justicia [Online]. Disponible en:
<http://www.eltelegrafo.com.ec/noticias/judicial/13/los-esteros-es-una-de-las-zonas-mas-violentas-de-guayaquil>
- [5] El Comercio. (2011, Junio 13). La cámara de Ojos de Águila no captó la balacera en el norte de Quito [Online]. Disponible en:
<http://www.elcomercio.com/actualidad/seguridad/camara-de-ojos-de-aguila.html>
- [6] National Instruments. (2008, Noviembre 7). ¿Cuál es la Diferencia Entre Calcular el Espectro de Potencia y la Densidad Espectral de Potencia? [Online]. Disponible en:
<http://digital.ni.com/public.nsf/allkb/3A19FE21E3D49EF7862574FA00730EF3>

- [7] Fundación Wikimedia, Inc. (2014, Abril 23). Filtro de Wiener [Online]. Disponible en: https://es.wikipedia.org/wiki/Filtro_de_Wiener
- [8] T. Yang, "The Algorithms of Speech Recognition, Programming and Simulating in MATLAB", Tesis de Licenciatura, Fclty. ENGINEERING AND SUSTAINABLE DEVELOPMENT, Univ. de Gävle, Gävle, Suecia, 2012.
- [9] Matlab. Simulación y Model-Based Design [Online]. Disponible en: <http://www.multon.com.mx/micrositios/matlab/inicio.html#descripción>
- [10] Arduino. ¿QUÉ ES ARDUINO? [Online]. Disponible en: <http://arduino.cl/que-es-arduino/>
- [11] Arduino. Genuino Uno Rev3 [Online]. Disponible en: <https://store.arduino.cc/product/GBX00066>
- [12] BotScience. Módulo de reconocimiento de voz para Arduino [Online]. Disponible en:
http://botscience.net/store/index.php?route=product/product&product_id=77
- [13] Tecnología en Bolivia. Módulo de Reconocimiento de Voz VR3 [Online]. Disponible en:
<http://tecbolivia.com/index.php/venta-de-componentes-electronicos-11/sensores/m%C3%B3dulo-de-reconocimiento-de-voz-vr3-detail>

ANEXOS

A continuación, se detallará el proceso de instalación de los servicios MySQL, PHP, Apache y scripts para guardar información en la base de datos, previamente para la instalación se utilizará el sistema de gestión de paquetes y que es la forma más eficiente y óptima se va establecer un paquete binario proporcionado por el mantenedor de la distribución.

Pero antes de proceder a la instalación de los servicios, es mejor mantener actualizados las listas de paquetes más recientes en nodo y el servidor, para esto utilizaremos el siguiente comando:

#apt-get update

Al escribir este comando, se verificará en sus repositorios si existe una actualización para obtener las listas más recientes de paquetes disponibles para nuestro sistema.

```
root@debian:~# apt-get update
Obj http://ftp.cl.debian.org wheezy Release.gpg
Obj http://ftp.cl.debian.org wheezy Release
Obj http://ftp.cl.debian.org wheezy/main 1386 Packages
Des:1 http://security.debian.org wheezy/updates Release.gpg [836 B]
Obj http://ftp.cl.debian.org wheezy/main Translation-es
```

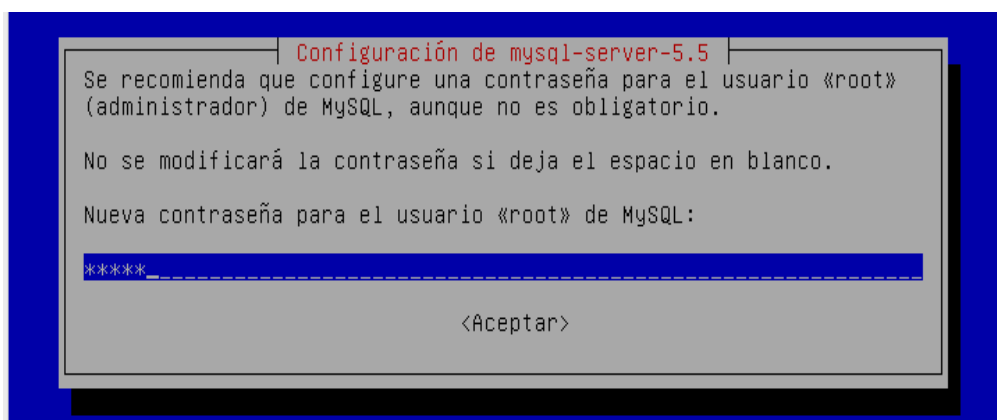
Instalación de MySQL en Raspberry Pi y el servidor

MySQL es el gestor de Base de Datos que se procederá a instalar tanto en el Raspberry (nodo) y el servidor (centro de operaciones remoto). Dado que ambos tienen la distribución Debian de Linux.

Para esto debemos ingresar el siguiente código “apt-get install mysql-server mysql-server mysql-common mysql-client”

```
root@debian:~# apt-get install mysql-server mysql-common mysql-client
```

Cuando estemos instalando MySQL nos va a presentar un cuadro donde tenemos que configurar la contraseña.



Una vez establecida la contraseña para el usuario **root** nos pide que volvamos a ingresarla lo cual me parece una buena idea para evitar cualquier tipo de error.

```
i386.deb) ...
Seleccionando el paquete heirloom-mailx previamente no seleccionado.
Desempaquetando heirloom-mailx (de ../heirloom-mailx_12.5-2_i386.deb) ...
Seleccionando el paquete libhtml-template-perl previamente no seleccionado.
Desempaquetando libhtml-template-perl (de ../libhtml-template-perl_2.91-1_all.d
eb) ...
Seleccionando el paquete mysql-client previamente no seleccionado.
Desempaquetando mysql-client (de ../mysql-client_5.5.31+dfsg-0+wheezy1_all.deb)
...
Seleccionando el paquete mysql-server previamente no seleccionado.
Desempaquetando mysql-server (de ../mysql-server_5.5.31+dfsg-0+wheezy1_all.deb)
...
Procesando disparadores para man-db ...
Configurando libwrap0:i386 (7.6.q-24) ...
Configurando libaio1:i386 (0.3.109-3) ...
Configurando libmysqlclient18:i386 (5.5.31+dfsg-0+wheezy1) ...
Configurando libnet-daemon-perl (0.48-1) ...
Configurando libplrpc-perl (0.2020-2) ...
Configurando libdbi-perl (1.622-1) ...
Configurando libdbd-mysql-perl (4.021-1+b1) ...
Configurando mysql-client-5.5 (5.5.31+dfsg-0+wheezy1) ...
Configurando mysql-server-core-5.5 (5.5.31+dfsg-0+wheezy1) ...
Configurando mysql-server-5.5 (5.5.31+dfsg-0+wheezy1) ...
[ ok ] Stopping MySQL database server: mysqld.
```

Configurar Maestro - Esclavo en replicación de MySQL

La replicación MySQL es un proceso que le permite mantener fácilmente múltiples copias de un conjunto de datos MySQL haciendo que copian automáticamente desde un maestro a una base de datos de esclavos. Para que el proceso funcione se necesitan dos direcciones IP: una del servidor maestro y otra del esclavo.

Las direcciones IP serán así:

Base de datos del nodo 192.168.2.12 - Maestro

Base de datos del servidor 192.168.2.48 - Esclavo

Abrir el archivo de configuración de MySQL en la base maestro.

sudo /etc/mysql/my.cnf nano

Encontrar la siguiente línea

bind-address = 127.0.0.1

Reemplazar la dirección IP estándar con la dirección IP del nodo

bind-address = 192.168.2.12

El siguiente cambio de configuración se refiere a la `id_servidor`, que se encuentra en la sección `[mysqld]`. Puede elegir cualquier número de este punto (que sólo puede ser más fácil comenzar con 1), pero el número debe ser único y no puede coincidir con cualquier otro `servidor-id` en el grupo de replicación. Asegúrese de que esta línea es sin comentar.

server-id = 1

Pasar a la línea de `log_bin`. Aquí es donde se guardan los datos reales de la replicación. El esclavo va a copiar todos los cambios que se registran en el registro. Para este paso simplemente tenemos que comentar la línea que hace referencia a `log_bin`:

log_bin = /var/log/mysql/mysql-bin.log

Por último, hay que designar a la base de datos que se va a replicar en el servidor esclavo. Puede incluir más de una base de datos mediante la repetición de esta línea para todas las bases de datos que se necesitan.

binlog_do_db = NewDatabase

Después de realizar todos los cambios, reiniciamos el servicio

sudo /etc/init.d/mysql restart

Después abrimos la base datos

mysql -u root -p

Tenemos que conceder privilegios al esclavo. Puede usar esta línea para nombrar a su esclavo y establecer su contraseña. El comando debe tener el formato siguiente:

GRANT REPLICATION SLAVE ON * * A 'slave_user' @ '%' IDENTIFIED BY 'contraseña';

FLUSH PRIVILEGES;

Se necesita abrir una nueva ventana

USE NewDatabase;

Después de eso, bloquear la base de datos para evitar cualquier nuevo cambio:

FLUSH TABLES WITH READ LOCK;

A continuación, escriba:

SHOWN MASTER STATUS;

Verá una tabla que debe ser algo como esto:

mysql> SHOW MASTER STATUS;

```
+ -----+ +-----+ +-----+ +-----+
| Archivo | posición | Binlog_Do_DB | Binlog_Ignore_DB |
+ -----+ +-----+ +-----+ +-----+
| mysql-bin.000001 | 107 | NewDatabase | |
+ -----+ +-----+ +-----+ +-----+
```

1 row in set (0.00 sec)

Esta es la posición desde la cual la base de datos de esclavos se iniciará replicante. Anote estos números, ya que resultarán útiles más adelante.

Si realiza algún cambio nuevo en la misma ventana, la base de datos se desbloqueará automáticamente. Por esta razón, se debe abrir la nueva pestaña o ventana y continuar con los siguientes pasos.

Procediendo con la base de datos sigue bloqueado, exportar su base de datos usando mysqldump en la nueva ventana (asegúrese de que está escribiendo este comando en el shell bash, no en MySQL).

```
mysqldump -u root -p --opt NewDatabase> newdatabase.sql
```

Ahora bien, volviendo a la ventana de su original, desbloquear las bases de datos (por lo que se pueda escribir de nuevo). Terminar saliendo de la concha.

```
UNLOCK TABLES;
```

```
QUIT;
```

Configure la base de datos del esclavo

Ingresa en el servidor esclavo, abre MySQL y crear la nueva base de datos que va a replicar desde el maestro (luego sale):

```
CREATE DATABASE NewDatabase;
```

```
QUIT;
```

Importar la base de datos que exportó previamente de la base de datos maestra.

```
mysql -u root -p NewDatabase </path/to/newdatabase.sql
```

Ahora tenemos que configurar el esclavo de la misma manera como lo hicimos el maestro:

```
sudo /etc/mysql/my.cnf nano
```

Tenemos que asegurarnos de que tenemos algunas cosas establecidas en esta configuración. El primero es el server-id. Este número, como se ha mencionado antes tiene que ser único. Dado que se establece en el valor predeterminado (siendo 1), asegúrese de cambiar es algo diferente.

```
server-id = 2
```

Después de eso, asegúrese de que usted tiene los tres siguientes criterios debidamente cumplimentados:

```
relay-log = /var/log/mysql/mysql-relay-bin.log
```

```
log_bin = /var/log/mysql/mysql-bin.log
```

```
binlog_do_db = NewDatabase
```

Reiniciar MySQL una vez más

Abrir el gestor MySQL una vez más y escriba los siguientes datos, en sustitución de los valores para que coincida con su información:

```
CHANGE MASTER TO MASTER_HOST = '192.168.2.12 ', MASTER_USER = 'slave_user ', MASTER_PASSWORD = ' password ', MASTER_LOG_FILE = 'mysql-bin.000001 ', MASTER_LOG_POS = 107;
```

Este comando logra varias cosas al mismo tiempo:

Designa el servidor actual como el esclavo de nuestro servidor maestro.

Proporciona el servidor las credenciales de acceso correctos

Por último, permite que el servidor esclavo sabe por dónde empezar la replicación de; el archivo de registro principal y la posición provenir de los números que anotó anteriormente log.

Con eso, se ha configurado un servidor maestro y el esclavo.

Activar el servidor esclavo:

```
START SLAVE;
```

Usted será capaz de ver los detalles del esclavo de replicación escribiendo en este comando. El \ G reorganiza el texto para hacerlo más legible.

```
SHOW SLAVE STATUS \G
```

Si hay un problema en la conexión, puede intentar iniciar esclavo con un comando para saltar sobre ella:

```
SET GLOBAL SQL_SLAVE_SKIP_COUNTER = 1; SLAVE START;
```

Instalación de Apache y PHP

Hacemos un update de los repositorios y procedemos a instalar Apache y PHP:

```
sudo apt-get update
```

```
sudo apt-get install apache2 php5 libapache2-mod-php5
```


Reiniciamos Apache:

```
sudo /etc/init.d/apache2 restart
```

Una vez esta todo instalado probamos que todo funciona perfectamente, para ello creamos un archivo en la ruta **/var/www** llamado **testphp.php** incluyendo la siguiente función:

```
<?php phpinfo(); ?>
```

Por último, introducimos en el navegador del cliente la siguiente dirección **192.168.2.48/testphp.php**. Si nos aparece información sobre php es que todo ha salido bien y que ya tenemos instalado nuestro servidor web.

PHP Version 5.4.4-14+deb7u3 	
System	Linux raspberrypi 3.6.11+ #474 PREEMPT Thu Jun 13 17:14:42 BST 2013 armv6l
Build Date	Jul 17 2013 21:59:46
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/apache2
Loaded Configuration File	/etc/php5/apache2/php.ini
Scan this dir for additional .ini files	/etc/php5/apache2/conf.d