UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

“Aplicación del reconocimiento de la voz de una persona en un circuito de seguridad”

T E S I S

que para obtener el título de

INGENIERO ELÉCTRICO ELECTRÓNICO

P R E S E N T A

RAMÍREZ SOSA JUAN MANUEL

DIRECTOR DE TESIS:
DR. ABEL HERRERA CAMACHO

Ciudad Universitaria 2007

Indice

Antecedentes

1. Antecedentes

1.1 Comunicación y sus elementos

Los sistemas de comunicación en general transportan información, sea cual sea el sistema de comunicación, el cometido de este es trasmitir e intercambiar en la mayoría de estos información. Las señales de voz en uno de estos sistemas en los cuales hay un sistema dedicado al intercambio de información, que por medio de señales acústicas nos podemos comunicar entre nosotros.

En el transcurso de la historia se han presentado grandes intentos de producir voces humanas por medio de mecanismos, algunos con grandes éxitos en su época. En muchos de los casos eran ingeniosos arreglos de tuberías conectados a un locutor, en otros de los caso arreglos de gran ingenio capaces de producir sonidos vocablos. En el siglo XX con el nacimiento de la telefonía, se ha avanzado de manera acelerada el tema Voz-Maquina, hasta que en nuestra actualidad se han desarrollado métodos de comunicación entre estos.

Para tener un conocimiento un poco mas completo de un sistema de comunicación mencionaré los elementos que lo componen, estos son:

- **Emisor:** en nuestro caso es el cerebro que piensa.
- **Receptor:** es el aparato auditivo que canaliza la señal acústica y la transporta al cerebro.
- **Mensaje:** es la señal sonora que contiene vocablos o la idea a comunicar.
- **Código:** el código es el tipo de idioma en el cual se transmite la idea o mensaje.
- **Canal:** es el medio por donde viaja la onda por lo regular el aire, pero en la actualidad se pueden incluir medios eléctricos como el teléfono.
- **Contexto:** este punto incluye aspectos como el estado de humor, el interés, muchos aspectos subjetivos que modifican la señal de manera notable, así como aspectos físicos, como el comportamiento en la frecuencia, interferencia distorsión, ruido, etc.

1.1.1 Lenguaje

El *lenguaje* tiene sus elementos bien definidos uno de los principales, como debe de suponerse es la lengua, la cual nos permite la comunicación por medio de señales sonoras. La lengua es un sistema de signos que permite la comunicación de los individuos en una comunidad. Tiene carácter social ya que se desarrolla en una comunidad.

El *habla* consiste en seleccionar de entre los símbolos disponibles y organizarlos de una manera coherente, dependiendo de las reglas que rigen a este. Es individual, esto quiere decir que depende del individuo el cual lo articula.

El *lenguaje* tiene modalidades las cuales son zonales, también llamadas dialectos, que se desarrollan en ciertas comunidades.

Los signos lingüísticos se clasifican en dos tipos*: significado y significante*. El significado es el concepto metal el cual desarrollamos con una idea, y el significante es la imagen producida debido al concepto que provoca el significado de la palabra, de manera que para cada individuo estos don conceptos pueden formarse de manera diferente, dependiendo del contacto, experiencia o previo conocimiento del concepto formado por la palabra articulada.

La *palabra* son los elementos libres mínimos de los cuales se compone le lenguaje. La sintaxis son las reglas que rigen al lenguaje, siendo estas muy extensas y claras en la forma de uso, la sintaxis coordinan las palabras para formar frases u oraciones.

Los fonemas son la unidad básica fónica que compone el lenguaje, estas son de manera no unívoca, estos se materializan a través de sonidos.

1.1.2 Fonología y fonética

La fonología estudia el fonema, esto quiere decir que se encarga del estudio detallado, del sonido ideal del lenguaje, también el carácter simbólico y la representación mental. La fonética se refiere a los sonidos, pero de una vista fisiológica, en lo referente a la producción física de la voz, emisión y articulación.

La fonología hace un detallado estudio de la pronunciación en los sonidos del lenguaje hablado y todas sus posibles combinaciones. Además se encarga de observar la pronunciación en los diferentes individuos y haciendo diferencias en la posible entonación, referida a cada hablante, así como el acento. Uno de los grandes propósitos de la fonología es acortar lo más posible la cantidad de fonemas para precisar un idioma de una manera precisa.

La fonética es un campo de estudio dedicado al porque de la producción del habla de una manera más física, esto es: se encarga del análisis acústico, articulado y perceptivo. Tiene como propósito el análisis de los fonemas de una manera más tangible.

Para la trasmisión del mensaje se necesita un medio conductor, en este caso es el aire. Siendo el aire un medio elástico y con una masa en un metro cúbico un poco mayor a un kilogramo.

1.2 Producción de voz y tracto vocal.

De una manera muy generalizada en esta tesis se tratará de explicar un poco el funcionamiento fisiológico de la producción de la voz, esto será de manera generalizada una mención anatómica breve del aparato fonatorio. De alguna manera se va a tratar desde el punto de vista ingeniería el proceso, por lo cual se debe de tener conocimiento de las bases que lo componen, para tener una noción de su funcionamiento. A lo largo del estudio del comportamiento de la voz, llevo a personas en el pasado a tratar de realizar aparatos muy rudimentarios e ingeniosos para levar acabo la construcción de sintetizadores de voz de una manera mecánica, algunos con éxito para su tiempo, otros no tanto, pero al fin aportaciones a la investigación en el área del procesamiento de voz.

Los órganos productores de la voz se pueden dividir en tres regiones: el tracto pulmonar (producido por los pulmones y traquea), la laringe que es una área situada en la parte superior tráquea e inferior a la faringe, y las cavidades bucal y nasal. La modulación de la voz es un proceso realizado por el tracto vocal con objeto de regular el sonido e interponer sonidos adicionales e interrupciones ocasionales.

La voz humana se produce de manera voluntaria por medio de una fuente de energía que llamamos pulmones, en la forma de un flujo de aire, una de las partes que también compone el aparato fonatorio es la laringe que contiene las cuerdas vocales. Los pulmones son una masa esponjosa de una larga área, su capacidad es de 4 a 5 litros en un adulto. Las *cuerdas vocales* en realidad son dos membranas orientadas hacia atrás *(figura I.1)*. Por la parte frontal se unen con el cartílago tiroides, esta parte se puede palpar en los varones regularmente en la parte media del cuello, alguna veces llamada manzana de Adán. Por la parte posterior, esta sujeta a cartílagos aritenoides, los cuales pueden separarse voluntariamente por medio de músculos. La abertura que es formada por las dos cuerdas vocales se le denomina *glotis*.

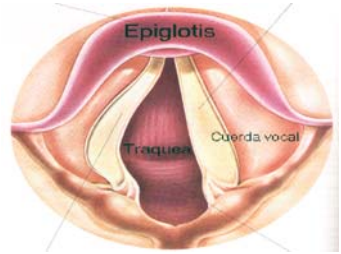

Figura I.1. Vista de las cuerdas vocales, las cuales son una de las principales fuentes productoras de la voz

En el caso de la respiración el glotis está de manera separada y adopta una forma triangular de forma que el aire pasa libremente y produce un sonido denominado *susurréo*, en este caso el aire pasa por esta apertura, creando una turbulencia, que ocasiona un ruido de banda ancha, el cual sirve como señal excitadora.

La *fricación* es algo similar al *susurréo*, pero con la diferencia que existe un lugar de articulación en el tracto vocal. Como un ejemplo por mencionar, la "*f*" de la palabrea en ingles "finger". De manera análoga que el *susurréo*, la *fricación* es de una amplitud mas baja que y tiene un espectro en frecuencia mas rico en las altas frecuencias, además que los sonidos fricativos son mas sonoros que los susurros.

Cuando la glotis empieza a cerrarse produce una turbulencia en el aire procedente de los pulmones, produciendo un ruido llamado aspiración. Al cerrarse más, el aire comienza a vibrar más y las cuerdas vocales producen un movimiento en forma de lengüetas y se producen de esa forma sonidos tonales o periódicos. La frecuencia producida por el individuo depende de demasiados factores como, la masa de las cuerdas vocales, la tensión que se les aplique y la velocidad que se le aplique al aire que las atraviesa proveniente de los pulmones, entre otros factores. A mayor tamaño menor frecuencia, de manera que en los hombres, los cuales tienen en promedio más grande las cuerdas vocales tengan la voz mas grave, en consideración con las mujeres que siendo estas más pequeñas pueden producir voces más aguadas. Así, para lograr emitir sonidos en el registro extremo de la voz se necesita un esfuerzo mayor. También aumenta la frecuencia al incrementar la velocidad del flujo de aire, razón por la cual al aumentar la intensidad de emisión se tiende a elevar espontáneamente el tono de voz.

Cuando el tracto vocal está cerrado y una persona continúa exhalado, la presión aumenta y esto causa un pequeño transitorio. La combinación de un silencio corto, seguido de una ráfaga de ruido crea una excitación aperiódica. Este sonido se puede clasificar en dos partes, si el transitorio es abrupto y limpio, se le denomina oclusiva o puede ser de manera plosiva como la "p" de la palabra "spin", si el sonido cae en un sonido muy parecido al fricativo se le denomina africativo, como sucede en la "j" de "reject".

La vibración es cuasiperiódica y ocurre en muchos lugares del tracto vocal, como en la "r" que se produce entre la lengua contra el paladar. A este tipo de sonido se le denomina *vibración.*

Finalmente, es posible bloquear la glotis de manera que no se produzca sonido alguno. Sobre la glotis se encuentra la epiglotis, el cual es un cartílago sobre la faringe que obstruye el paso del alimento ingerido en el aparato respiratorio.

A la porción compuesta de las cavidades faríngea, nasal y oral se le denomina cavidad supraglótica, mientras que la cavidad debajo de la laringe que contiene la traquea, los bronquios y los pulmones es llamado *cavidades infraglóticas*.

1.3 Sonidos y su clasificación.

Hay un sin fin de puntos, por los cuales se puede llevar a cabo una clasificación de los diferentes comportamientos de las señales de voz, tomando en cuenta diferentes criterios algunos de estos criterios se presentan a continuación:

a) **Según su carácter vocálico o consonántico:** desde un punto de vista mecánico acústico, los sonidos vocales son emitidos por la sola vibración de las cuerdas (Fonación) vocales sin ningún obstáculo entre la laringe y las aberturas oral y nasal.

Las consonantes por el contrario se emiten interponiendo un obstáculo formado por los elementos articulatorios.

b) Según su nasalidad: los sonidos en donde el aire pasa por la cavidad nasal se les denomina, sonidos nasales, de manera homologa, los sonidos en donde el aire es expulsado por la boca se le denominan orales.

c) Según su carácter tonal (sonido sonoro) o no tonal (sonido sordo): los sonidos en los cuales participan las vibraciones de las cuerdas sonoras se denominan sonoros o tonales. La tonalidad como la palabra indica es un sonido periodo, en nuestro caso como son señales de voz, son sonidos cuasi periódicos.

d) Según el lugar y del modo de articulación: este el proceso mediante el cual el aparto fonatorio interpone un obstáculo, en el flujo de aire. Para llevar acabo este tipo de articulación se hace uso de "los órganos articulados" que son: labios, dientes y las diferentes partes del paladar (alvéolo, paladar duro, paladar blando o velo), la lengua y la glotis. Dependiendo del punto de articulación se tienen los siguientes fonemas.

- **Bilabiales**: ésta es la oposición de ambos labios al paso del aire.
- **Labiodentales:** oposición de los dientes superiores con el labio inferior.
- **Linguodentaes:** es cuando la posición de la punta de la lengua se encuentra en una forma de obstrucción con los dientes superiores.
- **Alveolares:** es la oposición de la lengua con la zona alveolar en el techo de la boca.
- **Palatales:** oposición de la lengua con el paladar duro.
- **Velares:** es cuando se presenta una oposición de la lengua con el paladar duro.
- **Glotales:** es la oposición de la misma glotis al paso del aire deformando su sonido.

e) Según la posición de los órganos articulatorios: en el caso de las vocales, la articulación depende de los órganos que producen la resonancia el la articulación de los fonemas, como es el caso de la posición de la lengua, de la posición de los labios o del paladar blando.

f) Según la duración: la duración de los fonemas solo tiene importancia a nivel semántico en el idioma español, como es el caso de la acentuación.

1.4 Modelo digital de la voz

Para tener un buen modelo de la voz se deben de tener los siguientes elementos: los cambios en la señal de excitación, la respuesta al tracto vocal y el efecto que se le da con los labios en la radiación del sonido.

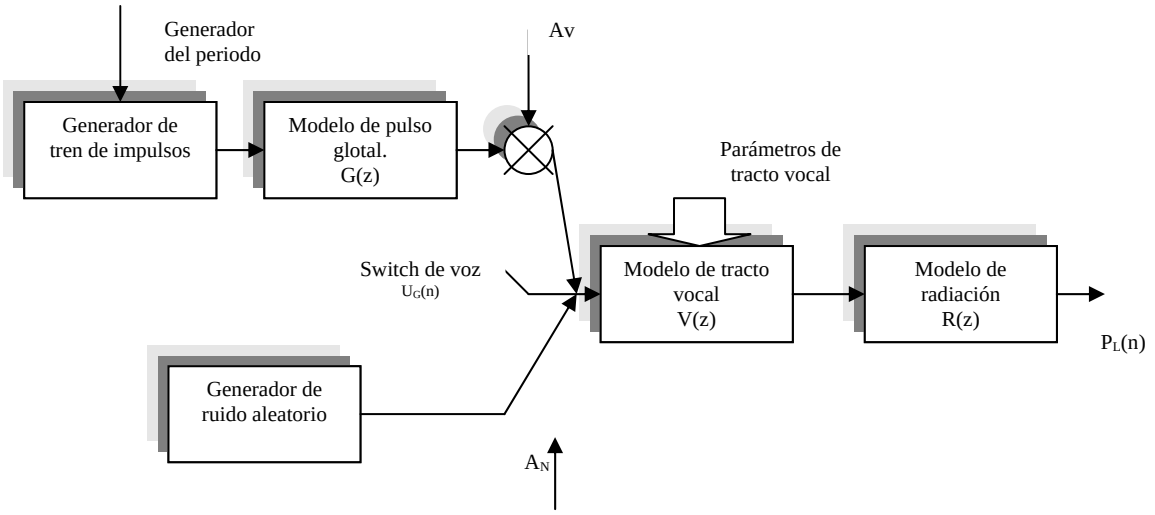

Figura I.2. Esquemático del modelo digital de la voz

En el diagrama de la figura I.2 se pueden observar tres tipos de excitación, de lo anterior se puede llegar a una reducción de solo dos: señales periódicas (sonoras) y ruido turbulento (sonidos sonoros), con una distribución gamma o laplaciana y espectro plano.

Los sonidos sonoros, provenientes de un modelo cambiante de la glotis, es un tren de impulsos espaciados a frecuencias iguales del tono fundamental. De manera que el tracto vocal puede ser modelado por la función de transferencia:

$$H(z) = \frac{G}{1 - \sum_{i=1}^{N} \alpha_i z^{-i}} \qquad \textbf{...(I.1)}$$

Donde G representa el factor de ganancia total y α_i , las ubicaciones de los polos de la función de transferencia. Los polos corresponden a las resonancias o formantes de la voz.

Cuando hablamos, la radiación sonora que producimos tiene una propiedad, esta es que a bajas frecuencias la presión del sonido es proporcional a la derivada de la velocidad volumétrica. Lo anterior dicho se puede representar matemáticamente de la forma que a continuación se anota.

$$R(z) = 1 - z^{-1} \qquad \textbf{...(I.2)}$$

Los tres efectos pueden ser representados por una sola función de trasferencia todo polos llamada espectro de envolvente. Esta función contiene o refleja la información principal de la voz tomando puntos estratégicos para su modelo y casi todos los sistemas lo tratan de generar o recuperar en el caso de esta tesis se tratara en una primera instancia de modelar para asi en un sistema independiente, hacer el reconocimiento de un hablante.

1.5 Sonidos y la voz

1.5.1 Las vocales

En el español los sonidos vocálicos corresponden a las cinco vocales del alfabeto. Todas esta son articulaciones abiertas de corta duración, ya sea que estén acentuadas o no. Aproximadamente las formantes vocálicas tienen aproximadamente las frecuencias que se muestran en la tabla 1. Analizando la tabla se puede verificar que entre mayor sea el tracto vocal, incrementa el formante F0, de manera análoga entre mas adelante y levantada se encuentre la lengua mayor será el formante de frecuencia F1, como son los casos extremos o más ilustrativos de /**i**/ y /**u**/.

En /**a**/ el primer formante es aproximadamente 1,170 Hz y F2 esta cerca de 2,900 Hz. Para /**e**/ F1 YF2 tiene baja energía y existe una transición en el formante F0 al final de la vocal. Para la vocal /**i**/ los formantes F1 y F2 ocupan frecuencias alrededor de 2,900 y 4,000 Hz. Para /**o**/, F2 está en el intervalo de la tabla I.1 y tanto F0 como F1 ocupan frecuencias alrededor de 570 y 700 Hz. Para /**u**/ también se pierde F2 en la tabla I.1 y F1 tiene una pendiente decreciente pronunciada.

Vocal	F0	F1	F2
a	900	1300	2100
e	375	2200	2550
i	325	2300	3900
o	400	550	4300
u	325	425	4500

Tabla I.1. Los tres primeros formantes de los fonemas vocálicos.

1.5.2 Diptongos

Un diptongo se refiere al monosílabo que empieza en o cerca de la posición articulatoria de un vocal y se mueva hacia la posición de otra vocal. La vocal que tiene mayor apertura en el tracto vocal se le denomina con el nombre de *núcleo silábico*, a la que tiene menor apertura se le denomina silaba marginal.

En la clasificación de los diptongos se tienen diptongos crecientes y diptongos decrecientes. En nuestro lenguaje hay dos diptongos semicrecientes [j] y [w], además de ocho diptongos crecientes.

- [ja] “hac*ia*”, [je] “t*ie*mpo”, [jo] “t*ie*mpo”, [ju] “c*iu*dad”.
- [wa] “ag*ua*”, [we] “s*ue*lo”, [wi] “r*ui*do”, [wo] “antig*uo*”.

Para diptongos decrecientes el margen silábico se denomina se llama semivocal. Las semivocales se diferencian de las semiconsonantes en el diptongo por su posición y por la manera en la cual es articulado. Existen dos semivocales [i] y [u] y seis diptongos crecientes:

- [ai] “aire”, [ei] “seis”, [oi] “hoy”.
- [au] “causa”, [eu] “feudo”, [ou] “lo unió”.

1.5.3 Consonantes

Es importante dar una clasificación de las categorías de las consonantes, para tener el conocimiento un poco más amplio del comportamiento de la voz y así de alguna manera entender su comportamiento para un análisis posterior. De manera que a continuación presento una clasificación un poco más generalizada y de manera que se toma en consideración más importante, el tracto vocal, ya que éste se puede modelar matemática mente como se muestra en la figura I.2.

- *Africativas.* Existe un cierre en el tracto vocal, seguido de una expiración la cual produce una turbulencia.
- *Aspiradas.* El tracto vocal esta cerrado de manera inicial en el punto de articulación y posterior mente se exhala aire antes de emitir el siguiente sonido.
- *Fricativas.* El tracto vocal al iniciar esta abierto parcialmente en el punto de la articulación y el velo esta cerrado.
- *Laterales.* El tracto vocal esta cerrado inicialmente en el punto de articulación, pero se encuentra abierto a los lados.
- *Nasales.* El tracto vocal se encuentra cerrado en el punto de la articulación y el velo está abierto.
- *Plosivas.* El tracto vocal esta cerrado, el pasaje nasal de igual manera y se realiza una exhalación limpia y constante.

- *Demivocales.* El tracto vocal se encuentra de manera semiabierta y sin ninguna turbulencia.
- *Vibrato.* En el punto de la articulación se presenta una abertura y cerradura oscilatoria, seguida de una exhalación gradual que continua con una turbulencia.

Todos los fonemas que corresponden a alguna de las clasificaciones como vocales, diptongos, semivocales y nasales se les llama de manera grupal como *sonorantes*. Este tipo de sonidos (sonorantes), se distinguen por el hecho de que son sonidos sonoros que excitan el tracto vocal con pulsos cuasiperiódicos, producidos por las cuerdas vocales.

Casi en su totalidad los idiomas en general, ocupan vocales nasales, plosivas y fricativas, sin embargo el número de estas y el lugar de su articulación varían enormemente dependiendo del idioma.

Plosivas

Existen seis formas de sonidos plosivos: /b/, /d/, /g/, /k/, /p/ y /t/, sus repectivas diferencias se representan el la tabla I.2.

Las consonantes de carácter plosivo se realizan de dos diferentes posiciones: después de una consonante nasal y una pausa par las consonantes /b/ y /g/, y después de una consonate nasal lateral en el caso de /d/. En cualquiera de las demás posiciones, las plosivas se mueven a fricativas. Por ejemplo, la /b/ es plosiva en la frase "u*n b*ote" y pasa fricativa en "ese bote".

Un espectro corto, seguido por un periodo largo sin energía arriba de las componentes sonoras se le denomina sonido plosivo.

Fonemas	Forma de articulación	Lugar de articulación	Ejemplo
/b/	voice	Labial	"vaca"
/d/	voice	Dental	"domino"
/g/	voice	Velar	"guerra"
/k/	unvoice	Velar	"camino"
/p/	unvoice	Bilabial	"ópera"
/t/	unvoice	Dental	"tela"

Tabla I.2. Consonantes Plosivas.

Fricativas

En el idioma español existen los siguientes sonidos fricativos: /f/, /θ/, /s/, /ʃ/ y /x/. esta clasificación de sonido consonantes, tiene la característica de que tiene una forma de onda cuasi-aleatoria con suaves cambios.

Los sonidos fricativos tienen una componente baja en frecuencia, cercana a los 300 Hz, resultante de la vibración de las cuerdas vocales. Esta franja vocal cambia abruptamente al primer formante del sonido subsecuente. Las formantes principales se encuentran en:

- Para /f/ y alófonos en 0-400, 1400-2200, 2900-4000 y 6000-800 Hz.
- Para /θ/, /s/ y alófonos en 0-500, 2600-3600 y 5000-8000 Hz.
- Para /ʃ/ y sus alófonos en 0-600 y 2200-3000 Hz, es este caso predomina la intensidad de las frecuencias centrales.
- Para /x/ y sus alófonos en 0-900 Hz, estas frecuencias son las predominantes.

Africativas

En nuestro idioma español el único sonido fricativo, es el de la /c/. Este sonido corresponde a la "ch", y se clasifica como un sonido sordo palatal. Un claro ejemplo es de la palabra

"muchacho". La característica sorda (con un alto contenido en altas frecuencias), se presenta en la palabra mencionada anteriormente.

Nasales

Los tres fonemas que componen esta categoría son: /m/, /n/ y /ŋ/, con la característica de que poseen un sonido sonoro. Las características de las transiciones de las nasales son idénticas a las características de las plosivas /p/, /t/ y la fricativa /ʃ/, sin embargo, la energía de éstas es mayor a altas frecuencias y se extiende hasta los 8,000 Hz.

Semivocales

Existen los fonemas laterales como es el caso de /ʎ/ y /l/, que corresponden a las letras "ll" y "l" respectivamente. El primero precede a fonemas fricativos interdentales sordos, por ejemplo "un dulce", la segunda cuando procede a una constante dental, como un ejemplo a mencionar "el toro".

También existen sonidos vibrantes /r/, el cual corresponde a la letra "r" y el fonema /ř/ que corresponde a la letra "r", esta ultima con la cualidad, que corresponde a una palabra que comienza con esta o le sigue una "n" o una "l" o en su respectivo caso corresponde a la letra "rr" la cual tiene un sonido mas vibratorio.

Los espectros de las llamadas vibrantes presentan interrupciones intermitentes en la señal por intervalos de silencio.

Procesamiento de Voz

2. Procesamiento de Voz

2.1 Preénfasis

El filtro Preéfasis es un sistema digital utilizado en el procesamiento de señales, implementado ya sea de primer orden, utilizando un filtro analógico paso-altas con una frecuencia de corte de 3dB en algún lugar entre 100 Hz y 1 kHz (la posición exacta no es crítica), o de manera digital se puede implementar esta misma aplicación de forma siguiente:

$$y[n] = x[n] - a * x[n-1] \qquad \textbf{...(II.1)}$$

de donde y[n] denota la muestra actual de salida del filtro preénfasis, ese la muestra x[n] es la muestra actual de nuestra señal, x[n-1] es la muestra anterior de la entrada de la señal y a es una constante elegida entre .9 y 1, siendo esta constante de manera no critica.

Aplicando una transformación z a nuestra función nos queda de la forma que se muestra a continuación:

$$H(z) = \frac{Y(z)}{X(z)} = 1 - a * z^{-1} \qquad \textbf{...(II.2)}$$

en donde se puede apreciar z^{-1} denota el retraso de nuestra señal de entrada.

De una manera general éste filtro preénfasis da un levantamiento de +6 dB/octava en le rango apropiado de tal forma que en el espectro medido tenga un rango dinámico similar a través de toda la banda de frecuencias.

2.2 Filtros por ventaneo

Una de las técnicas más utilizadas para el diseño de filtros digitales IIR están basadas en las transformaciones de sistemas IIR de tiempo continuo en sistemas IIR de tiempo discreto. Esto se debe a que los sistemas en tiempo continuo fueron los primeros en ser desarrollados y por lo tanto se encontraron muy avanzados cuando surgieron los sistemas digitales, así en sus inicios se vio la dificultad de desarrollar nuevos métodos para la implementación de los mismo por lo que se opto en tomar los mismos e implementarlos por medio de transformaciones matemáticas a los sistemas discretos.

En contraste los filtros FIR (finite impulse response) se diseñaron para implementaciones discretas. Por lo que estos sistemas están explícitamente diseñados para la respuesta en frecuencia es el tiempo discreto. Además, los filtros FIR asumen una fase constante, evitando así el problema de las factorización del espectro que complica el diseño directo de los filtros IIR, asumiendo de esta forma una fase constante.

El método más simple para diseñar un filtro FIR es el método de ventana. Y se representa por la ecuación:

$$H_d(w) = \sum_{n=0}^{\infty} h_d(n) e^{-jwn} \qquad \textbf{...(II.3)}$$

Donde h[d] es la secuencia de la respuesta al impulso que puede ser expresada en términos de $H[e^{jw}]$ como:

$$h_d(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} H_d(w) e^{jwn} dw \qquad \textbf{...(II.4)}$$

En general la respuesta $h_d(n)$ es infinita pero esta es truncada mediante el ventaneo, en el punto: $n = M - 1$ y multiplicando por la ventana rectangular o unitaria se obtiene:

$$h(n) = \begin{Bmatrix} h_d(n) & n = 0,1,...M-1 \\ 0 & otro \quad caso \end{Bmatrix} \qquad \textbf{…(II.5)}$$

Muchos de los sistemas que se consideran ideales para su estudio son definidos pro respuestas constantes o funcionales con discontinuidades en los límites entre las bandas. La ecuación (3) puede interpretarse como la representación en Fourier de la respuesta en frecuencia periódica de $H(e^{jw})$, siendo la secuencia *h[n]* los coeficientes de Fourier.

En la siguiente tabla se da un esbozo de las ventanas más utilizadas (Hamming, Hanning, Blackman, Rectangular, Bartlett). En esta comparación se puede apreciar, que mientras la ventana se acerque más a cero, como sucede con las ventanas Hamming, Hanning y Blackman, los lóbulos laterales (segunda columna), se reducen enormemente; en contraste, el lóbulo principal es más ancho(tercera columna) y por lo tanto estas tienen más transiciones en las discontinuidades de $H(e^{jw})$.

Tipo de ventana	***Amplitud pico del lóbulo lateral (relativa dB)***	***Ancho aproximado del lóbulo principal***	***Error pico aproximado 20 log$_{10}$(dB)***	***Ventana de Kaiser equivalent***	***Ancho de la transición de la ventana de Kaiser equivalente***
Rectangular	-13	4π/(M+1)	-21	0	1.81 π /M
Bartlett	-25	8 π /M	-25	1.33	2.37 π /M
Hanning	-31	8 π /M	-44	3.86	5.01 π /M
Hamming	-41	8 π /M	-53	4.86	6.27 π /M
Blackman	-57	12 π /M	-74	7.04	9.19 π /M

2.2.1 Efectos del ventaneo

- El ancho de banda de la ventana depende de su longitud, sin embargo, el nivel de lóbulos secundarios es independiente de N.

- N tiene que ser mayor que un periodo de “pitch” (para evitar una variación excesiva en la energía en tiempo corto de la señal de voz), pero más pequeño que la duración de un sonido para reflejar adecuadamente los cambios en la señal de voz.

No existe un valor de N adecuado para todos los casos, ya que la frecuencia fundamental (“pitch”) puede variar entre 20 muestras (500 Hz con fs= 10 kHz) para un niño o una mujer de voz aguda, hasta 250 muestras (40 Hz con fs= 10 kHz) para un hombre de voz grave.

2.3 Parámetros en el dominio del tiempo.

2.3.1 Energía

Uno de los objetivos que tiene el procesamiento digital, es obtener la información más representativa de las señales digitalizadas, en éste caso nos concierne el estudio de las señales de voz en lo que respecta al trabajo presentado en el desarrollo de esta tesis. El desarrollo de algoritmos de programación nos ha dado pauta para determinar cuando una señal captada en un sistema es de voz o no. Además, se puede llegar a determinar si las señales son sonoras, sordas o silencio (ruido de fondo).

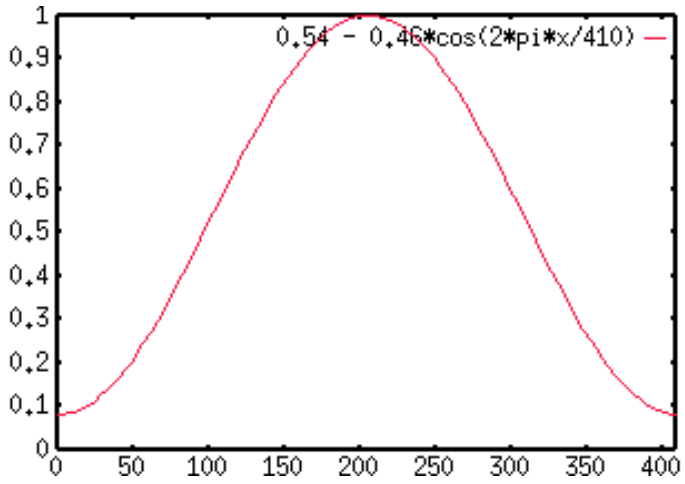

Figura II.1. Ventanas Hamming uno de los filtros más utilizadas en procesamiento de voz

En esta parte conceptual se verán métodos. Lo que significa que los métodos que describirán involucran directamente la forma de onda de la señal. En contrate los métodos en el dominio de la frecuencia, los cuales utilizan (implícita o explícitamente) alguna forma involucran alguna representación del espectro de la señal.

Uno de lo métodos implementados en términos del dominio del tiempo para determinar el inicio y fin de la señal de voz, toma por parámetros, la taza promedio de cruces por cero, la energía y la función de autocorrelación. Estos algoritmos en el dominio del tiempo son muy utilizados debido a su simplicidad.

2.3.2 La voz en el dominio del tiempo

Una de las principales hipótesis en el procesamiento digital de voz es que las propiedades de las señal de voz son parte de un proceso estocástico no estacionario; no obstante, sus propiedades cambian “lentamente” (5-10 sonidos por segundo). Esta hipótesis lleva a una variedad de métodos de procesamiento en tiempo corto, en las cuales se aíslan y procesan pequeños segmentos de la señale de voz como si fueran segmentos de un sonido con propiedades fijas, debido a la consideración mencionada anteriormente de que cambia lentamente nuestra señal. A menudo éste tipo de segmentos utilizados en el procesamiento de voz son comúnmente llamados *tramas de análisis*.

Uno de los ejemplos más ilustrativos de lo que se discutió con anterioridad es *la energía en tiempo corto*. La energía de una señal en tiempo corto esta definida como:

$$E = \sum_{m=-\infty}^{\infty} x^2(m) \qquad \textbf{...(II.6)}$$

La anterior ecuación da poca información acerca del comportamiento de las señales de voz, ya que para nuestra aplicación tiene poco significado o utilidad. Una definición más simple e ilustrativa en tiempo corto se define como:

$$E_n = \sum_{m=n-N+1}^{n} x^2(m) \qquad \textbf{...(II.7)}$$

De manera que la ecuación anterior nos dice, que la energía en un punto *n* es la sumatoria de los cuadrados de la señal desde n-N+1 hasta el punto donde se desea conocer la energía de la señal en tiempo corto.

En la siguiente figura II.2 se muestra el cálculo de la secuencia de energía en tiempo corto de una señal.

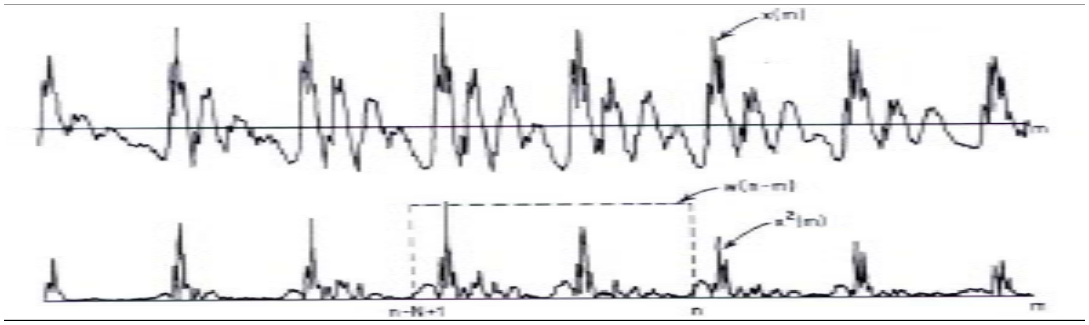

Figura II.2. Ilustración del cálculo de la energía en tiempo corto.

2.3.3 Cruces por cero

En tiempo discreto de dice que en análisis de una señal cuando ocurre un cruce por cero cuando una muestra sucesiva tiene un signo algebraico diferente. La frecuencia en la que ocurren los cruces por cero depende en gran medida del contenido en frecuencia de la señal. Esto es particularmente cierto para las señales de banda angosta.

Las señales de voz son consideradas señales de voz de banda ancha y por consecuencia la interpretación de cruces por cero es menos precisa. Pero por medio de las propiedades espectrales se puede obtener un estimado, usando una representación en la tasa promedio de cruces por cero en tiempo corto.

Existen un sin fin de consideraciones que se deben de tomar en cuenta en el momento de hacer la implementación basada en la tasa promedio de cruces por cero. Aunque en el algoritmo básico en la detección de cruces por cero solo se necesita analizar el cambio de signo en pares de muestras sucesivas, se debe de tener especial cuidado en el proceso de muestreo. De manera obvia, la taza de cruces por cero se encuentra frecuentemente afectada por el offset de DC de nuestra señal en el convertidor analógico digital, el ruido de 60 [Hz] de la señal y cualquier ruido que pudiera estar presente en nuestra señal.

El problema de localizar el inicio y fin de un segmento de voz inmerso en ruido es de suma importancia en la implementación del reconocimiento de palabras aisladas para el análisis en tiempo real en el reconocimiento de las palabras aisladas, ya que es necesario localizar las señales de voz que corresponden a cada palabra. Para esto se pude recurrir al uso de la energía en tiempo corto para la detección, sin embargo en ambientes con ruido es necesario tomar otras consideraciones.
El poder determinar cual es el inicio y el final de una palabra, nos da ciertas ventajas en los sistemas de reconocimiento:

- Procesar menor número de información, eliminando las partes innecesarias de ruido.

- Comparar únicamente los patrones de información o las partes útiles de la señal.
- Evitar confusiones a causa del ruido o señales de fondo.

Algunos de los problemas que se presentan en la implementación del algoritmo de inicio y fin de palabra:

- Espurias de ruido que se pueden confundir con la señal.
- Silencios contenidos dentro de las palabras que tienen fonemas plosivos (ej. /t/, /p/, /k/) que pueden confundirse con un falso principio o fin.
- Los fonemas fricativos (ej. /f/, /th/, /h/, etc.), ya que tienen baja energía.
- Sonidos cortos (ej. /t/, /p/, /k/).
- Detección de fonemas nasales al final de la palabra (baja energía y cruces por cero)
- Respiraciones del locutor, que pueden confundirse por su duración.
- Los micrófonos tienen resonancia después de de pronunciar una palabra (sobre todo en vocales).
- Los niveles de ruido pueden confundirse con la señal de voz.

2.4 Algoritmo de detección de inicio y fin (L. R. Rabiner, M. R. Sambur)

Ante las dificultades para la detección del inicio y fin, se ha desarrollado un método que consiste en considerar las características de los sonidos:

Sonidos sonoros (*voiced*)	Tiene alto contenido en energía. Ocupan las frecuencias bajas del espectro de la voz humana.
Sonidos no sonoros (*unvoiced*)	Tienen bajo contenido de energía. Ocupan las frecuencias superiores del espectro de la voz humana.

De esta forma se puede implementar un detector que incluya ambas características, análisis de energía y frecuencia, cruces por ceros y magnitud promedio (o energía en tiempo corto).

2.4.1 Algoritmo de Detección de Inicio y Fin de palabra:

Detección de inicio

1. Por cada trama de 128 muestras, calcular las funciones: cruce por ceros { *Z[n]* } y la magnitud promedio de la señal { M[n] }, estas funciones se definen a continuación:

$$M_n = \sum_{m=0}^{N-1} |x[m]|$$

$$Z_n = \frac{\sum_{m=0}^{N-2} \left| sign(x[m+1]) - sign(x[m]) \right|}{2N}$$

2. Para obtener las estadísticas del ruido ambiental, se considera que las primeras diez ventanas son ruido, con lo cual se tiene:

$$Ms_n = \{M_1, M_2, \cdots, M_{10}\}$$
$$Zs_n = \{Z_1, Z_2, \cdots, Z_{10}\}$$

3. Calcular la media y la desviación estándar para las características del ruido y obtener los siguientes umbrales:

Umbral	***Nombre del umbral***	***Valor***
UmbSupEnrg	Umbral Superior de Energía	0.5*max{M_n}
UmbInfEnrg	Umbral Inferior de Energía	μ_{Ms} + 2*σ_{Ms}
UmbCruCero	Umbral de cruces por cero	μ_{Zs} + 2*σ_{Zs}

4. Recorrer la función ***M_n*** incrementando en una unidad a ***n*** de 11 hasta que ***M_n*** > ***UmbSupEnrg***. En éste punto estamos garantizando presencia de señal. A éste punto lo marcaremos como ***In***.

5. Resulta lógico pensar que el inicio se la señal se encuentra en algún punto anterior a ***In***, por lo que ahora recorremos la función ***M_n*** desde ***n*** = ***In*** hasta que ***M_n*** < ***UmbInfEnrg***. Éste punto lo marcaremos como ***Ie*** y lo reconocemos tentativamente como el inicio de la señal, determinado por la función de magnitud.

6. Ahora decrementamos ***n*** desde ***n*** = ***Ie*** hasta ***n*** = ***Ie – 25*** o en su defecto ***n*** = ***11***, verificando si sucede alguna de las siguientes condiciones en la función de cruces por cero, ya que lo que ahora buscamos es la posibilidad de que un sonido *no sonoro* preceda a un sonido *sonoro*.

   - Sí { ***Z_n*** < ***UmbCruCero*** } significa que no encontramos alguna porción de la señal con aumento importante de frecuencia en 25 ventanas anteriores, por lo tanto el inicio es ***Ie***.

   - Sí encontramos que { ***Z_n*** > ***UmbCruCero*** } menos de tres veces seguidas significa que solo fue una espiga de ruido, el punto de inicio sigue siendo ***Ie***.

   - Si encontramos que { ***Z_n*** > ***UmbCruCero*** } al menos tres veces seguidas hemos encontrado un sonido *no sonoro*, entonces buscamos el punto ***n*** para el cual { ***Z_n*** > ***UmbCruCero*** } la primera de las más de tres veces, es decir, el punto para el cual la función ***Z_n*** sobrepasa el umbral, indicando el comienzo del sonido *no sonoro* y desplazamos el inicio de la palabra a ***Iz***.

Para la detección de fin de la palabra, hacemos lo mismo pero en sentido inverso a partir del punto (4) de la sección anterior, como si detectáramos un inicio con la señal invertida en el tiempo.

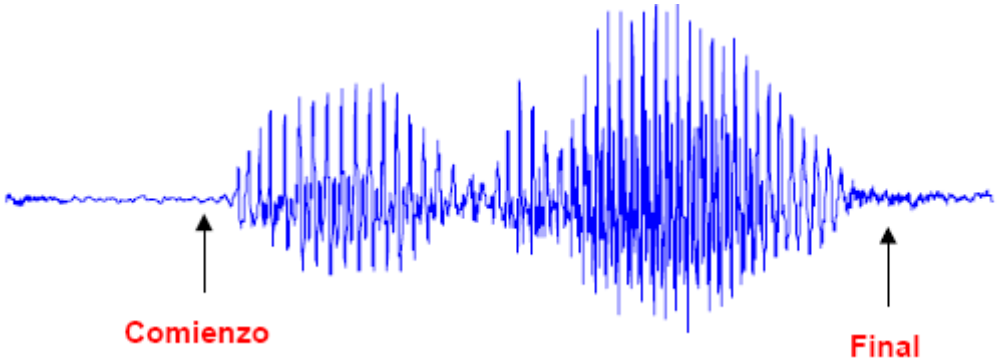

Figura II.3. Detección de inicio y fin de la señal de voz.

2.5 Autocorrelación en tiempo corto

La autocorrelación de una señal en tiempo corto se define como:

$$\phi(k) = \sum_{m=-\infty}^{\infty} x(m)x(m+k) \qquad \textbf{...(II.8)}$$

Si la seña es aleatoria o periódica entonces la manera apropiada para expresar la función es:

$$\phi(k) = \lim_{N\to\infty} \frac{1}{(2N+1)} \sum_{m=-N}^{N} x(m)x(m+k) \qquad \textbf{...(II.9)}$$

En cualquiera caso la representación de la señal por medio de la autocorrelación es una de las formas más convenientes de mostrar ciertas propiedades de la señal. Una de las características de la función autocorrelación por mencionar algunas, son:

- Es una función par
- Posee un valor máximo en k=0
- La cantidad de $\phi(0)$ es igual a la energía (ecuación 7)para señales determinísticas o la potencia promedio para las señales aleatorias o periódicas.

Por lo que se puede decir que la autocorrelación contiene en sus elementos, en caso especial a la energía. Otra parte importante de esta función es que se puede demostrar la periodicidad de la señal por medio de la misma, además de que sin importar el origen de la señal, se puede estimar la periodicidad, localizando el primer máximo de la función de autocorrelación. Por esas propiedades, la función de autocorrelación es de gran utilidad en los diseños de procesamiento digital de voz.

Una de las características de la función de autocorrelación, es que proporciona una medida de la relación de la señal con una copia desfasada de sí misma.

La función de autocorrelación para una trama de *N* muestras es:

$$r(k) = \sum_{m=0}^{N-1-k} s(m)s(m+k) \qquad para\ k = 0,1,2,...,p \qquad \textbf{...(II.10)}$$

2.6 Extracción de la información del hablante de la señal de voz.

Hablando en términos generales, el reconocimiento del hablante esta íntimamente relacionada con la fisiología y el comportamiento del hablante. De manera más explicita, las variaciones el tamaño de las cavidades del tracto vocal produce variaciones en la resonancia característica del espectro de la señal de voz. Algunas variaciones en el tamaño de las cuerdas vocales esta relacionado con cambios en el tono promedio o de la frecuencia fundamental. Variaciones en el tamaño de las cavidades nasales, producen diferencias espectrales en sonidos nasales de voz. No comúnmente las variaciones anatómicas, como son la configuración o estructura de los dientes y el paladar, producen variaciones en la voz, como son sonidos no típicos de los labios y nasalidad anormal.

Correlacionas de comportamiento de la identidad del orador en la señal de voz son más difícil de especificar. El comportamiento de la información del las señales de voz, esta asociado con características individuales en el sonido de la articulación de la voz, en características de transición de sonido a sonido, en características del contorno del tono, ritmo y tiempo.

Medidas explicitas de las características del hablante son comúnmente difíciles de realizar. Por ejemplo, las características de nasalidad y aliento son difíciles de medir ya que la dificultad en la cuantización, segmentación, y la medida de la voz nasal, o en la correlación y cuantización del aliento en los eventos de la voz con señales características. Las fuentes de la medidas de la voz son mencionadas algunas veces como buenos candidatos para caracterizar a lo hablantes. Sin en cambio, otras medidas como el “pitch” o tono, no son fáciles de extraer de una señal de voz. Medir la forma de onda del tracto vocal requiere cuidadosa filtración de la señal implicando buenos conocimientos y control de la señal grabada y transmisión característica. A pesar de que muchas características de la voz son difíciles de medir y muchos otros son muy difíciles de especifican, nosotros sabemos que muchas características pueden ser capturadas implícitamente en las medidas de la señal además de que pueden ser mejoradas fácilmente. Como son las medidas en tiempo corto, energía, y frecuencia fundamental son relativamente fáciles de obtener. Además, esta medida puede resolver diferencias en las características del hablante que regularmente sobrepasa la discriminación humana.

2.6.1 Variabilidad

El factor más significante que afecta el mejoramiento del reconocimiento del hablante es la variación en la señal característica prueba a prueba. La variación surge en el momento de ser grabadas las muestras y transmisión de al voz, y la más importante, las variaciones generadas por el hablante. Estas variaciones pueden ser de forma voluntaria o involuntaria. Es posible también para los hablantes cambiar la pronunciación, afectando las características espectrales de los sonidos individuales de la voz. Un camino simple para cambiar las características espectrales es para variar la cantidad de esfuerzo significativo de expresión a expresión. Hablantes no pueden repetir una expresión precisamente de la misma manera de prueba a prueba. Aun cuado es bien conocido que las señales de la misma expresión repetidas en una sesión tienen una correlación más alta que expresiones grabadas en diferentes tiempos.

El típico modelo de reconocimiento de voz, opera en uno o dos modos de entradas, texto dependiente o texto independiente. En el modo de texto dependiente debe de proveer modelos del mismo texto para el entrenamiento y prueba de reconocimiento. En el modo de texto independiente no esta considerado proveer textos en pruebas de reconocimiento, sin en cambio el texto puede ser mostrado para especificar el vocabulario.

2.6.2 Operación básica.

Los ingredientes básicos del reconocimiento del habla son mostrados en la figura II.4 siguiente. Una expresión de entrada de un hablante desconocido es analizada para extraer los rasgos característicos. La medida de rasgos es comparada con rasgos prototipos obtenidos de un

modelo conocido del hablante.

El típico modelo de reconocimiento de voz, opera en uno o dos modos de entradas, texto dependiente o texto independiente. En el modo de texto dependiente debe de proveer modelos del mismo texto para el entrenamiento y prueba de reconocimiento. En el modo de texto independiente no esta considerado proveer textos en pruebas de reconocimiento, sin en cambio el texto puede ser mostrado para especificar el vocabulario.

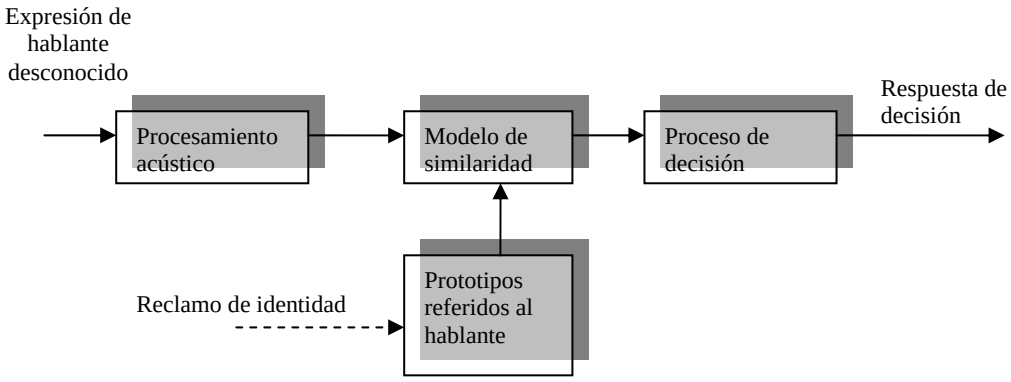

Figura II.4. Componentes básicos del reconocimiento del habla

2.7 Análisis de predicción lineal

Una de las técnicas más utilizadas en análisis de señales de voz es el análisis de predicción lineal. Sus principales ventajas son la precisión de las estimaciones obtenidas y su rapidez de cálculo. La idea básica de éste tipo de análisis es la suposición de que el valor de una muestra de la señal puede ser predicho en base a una combinación lineal de los valores de muestras anteriores. Minimizando el error entre el valor de la muestra y la predicción se puede obtener un conjunto único de valores para los coeficientes de la combinación lineal que refleja las características de la señal. Estos coeficientes se suelen denominar *coeficientes de predicción* o *coeficientes LPC*.

2.7.1 Coeficientes de predicción lineal (LPC)

La idea principal de éste método de codificación de voz es que dada una muestra de voz en un tiempo **n, s(n)**, puede ser aproximada por una combinación lineal de **p** muestras de voz precedentes.

Al minimizar la diferencia de la suma de los cuadrados (sobre un intervalo finito), entre las muestras de voz actuales y las predichas se obtiene un número único de coeficientes de predicción, de forma matemática tenemos una expresión de la forma que se muestra a continuación:

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + ... + a_p s(n-p) \qquad \textbf{...(II.11)}$$

Para determinar éste conjunto de coeficientes de predicción se utiliza el método de autocorrelación (el cual fue antes mencionado), y se llega a una matriz de la forma:

$$\begin{bmatrix} R_n(0) & R_n(1) & . & . & R_n(p-1) \\ R_n(1) & R_n(0) & . & . & R_n(p-2) \\ . & . & . & . & . \\ . & . & . & . & . \\ R_n(p-1) & R_n(p-2) & . & . & R_n(0) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ . \\ . \\ \alpha_p \end{bmatrix} = \begin{bmatrix} R_n(1) \\ R_n(2) \\ . \\ . \\ R_n(p) \end{bmatrix}$$

donde $R_n(\sigma)$ es la autocorrelación de la señal en tiempo corto, y a_k son los coeficientes de predicción lineal (LPC's) que resuelven el sistema. A esta matriz se le denomina **Toeplitz**, para conocer los coeficientes (LPC), es solo resolver el sistema se ecuaciones de dimensión **p *x* p,** o resolver por algún método numérico conocido.

Esta técnica de análisis está estrechamente relacionada con el modelo de producción de voz presentado. Si denominamos *U(z)* y *S(z)* respectivamente a las transformadas *Z* de la excitación *u(n)* y de la señal *s(n)*, y *H(z)* a la función de transferencia del tracto vocal, podemos escribir las relaciones

$$H(z) = \frac{S(z)}{U(z)} = \frac{\sigma}{1 - \sum_{k=1}^{p} a_k z^{-k}} \qquad \textbf{...(II.12)}$$

En la que se ha supuesto que el filtro digital *H(z)* contiene únicamente polos. Esta suposición no es cierta en general, sin embargo, escogiendo un orden de predicción *p* suficientemente elevado, se puede obtener una representación adecuada incluso para los sonidos nasales cuyo espectro presenta polos y ceros en su función de transferencia.

Para determinar *H(z)* es necesario calcular los valores de los coeficientes de predicción y de la ganancia . Los coeficientes k de predicción se determinan minimizando la energía *E* de la señal error de predicción (diferencia entre el valor real de la señal y el predicho por el filtro) sobre un intervalo temporal de muestras de la señal. La motivación principal para esta aproximación es que para una señal generada, ya sea con excitación periódica o con ruido gausiano, los coeficientes obtenidos k coinciden exactamente con los *a k* originales.

2.7.2 Análisis de las características de los LPC's

Para realizar un análisis de los vectores de observación de las muestras de voz, es por medio de un análisis espectral de la señal. Esto se lleva acabo por medio de una segmentación de la señal en muestras de 128, las cuales se analizan considerándolas de manera individual, además de haber eliminado las tramas de ruido de la señal por medio del algoritmo de detección de inicio y fin de la señal. Uno de los análisis espectrales utilizados en el reconocimiento de palabras aisladas y en general en el procesamiento de voz, es el denominado como "predicción lineal" o también conocido como LPC por sus siglas en ingles "Linear Prediction Coefficients". El modelo completo del procesamiento en el que una trama de N_A muestras de la señal de voz es procesada, junto con un vector característico O_t. Los pasos en el procedimiento son los siguientes:

- *Preénfasis*: La señal de voz digitalizada, con una taza de muestreo de 8 Khz, es procesada por un filtro preènfasis, con el propósito de aplanar espectralmente la señal.

- *División de tramas*: La sección de N_A elementos o muestras de voz consecutivas, en éste caso 128 muestras de la trama de voz corresponden a 16 ms, son utilizadas para hacer comenzar el reconocimiento y es procesada. Cada señal de voz es segmentada en *n* número de tramas y cada segmento se considera una señal a la cual se le deben extraer sus características principales para llevar acabo el reconocimiento.

- *Ventana:* Cada trama al ser segmentada, por consiguiente, es multiplicada por una ventana cuadrada, por lo que su espectro en el domino de la frecuencia es afectado. En éste caso se debe de implementar una ventana Hamming para minimizar lo efectos provocados por la segmentación.

- *Análisis de autocorrelación:* cada conjunto de muestras de voz se autcorrelacionan, para obtener un conjunto de *p+1* coeficientes, donde *p* es el orden de análisis de los *LPC.*

- *Análisis LPC/Cepstral:* Para cada trama se calcula un vector de coeficientes LPC a partir del vector de autocorrelación utilizando el método Levinson-Durbin. Se calcula un vector LPC hasta el Q-esimo componente donde, donde Q>p.

- *Peso cepstral:* El ceptrum se define como la transformada inversa de Fourier, del modulo del espectro en la escala logarítmica.

El análisis de LPC tiene una importante ventaja en la estimación de los parámetros de voz tanto en su espectro, las formantes de la frecuencia y el ancho e banda. Ignorando el tracto nasal, vibraciones y otros factores, el tracto vocal puede ser modelado como una secuencia de cilindros concatenados de diámetro variante en el tiempo, el cual es excitado en uno de sus extremos por una forma de onda sintética, un generador de impulsos de voz sonoro, o un generador de ruido sordo. En la figura II.5 se muestra el concepto generalizado del método Rabiner-Schafer, el cual consiste en la concatenación acústica de tubos y su relación con el tracto vocal.

Si el número de las secciones de los cilindros es apropiadamente escogido y otras características son cuidadosamente seleccionadas, la caracterización matemática en el dominio del tiempo de la formulación, se lleva acabo una formulación y un acercamiento más próximo en la representación del tracto vocal.

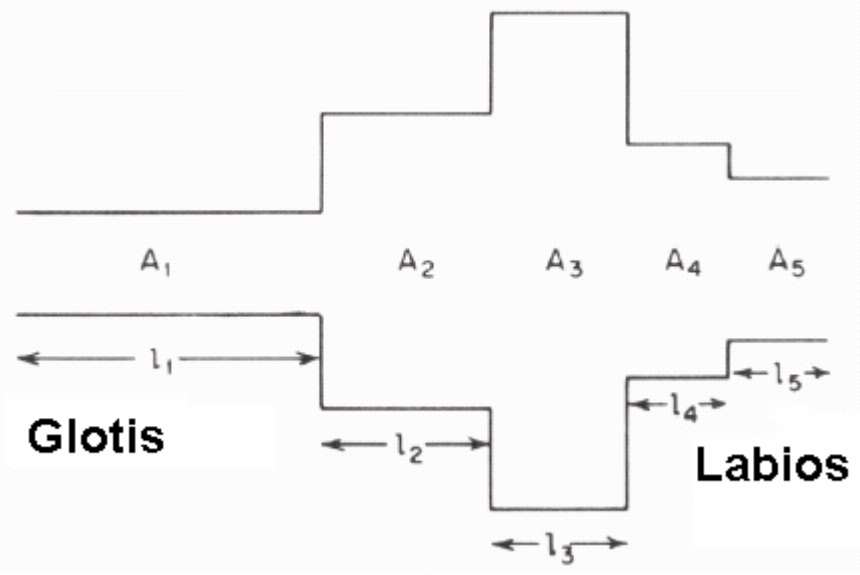

Figura II.5. Emulación por medio de concatenación
de tubos acústicos.

Uno de los algoritmos más utilizados para el cálculo de los coeficientes de predicción lineal es el denominado Levinson-Durvin. Éste es uno de los algoritmos más implementados en el calculo de los LPC, indispensables en la determinación de los LPC-Ceptral. Con éste tipo de

codificación se han implementado gran número de aplicaciones, como reconocedores de palabras aisladas, sintetizadotes de voz humana en diferentes lenguajes, implementación de comandos de voz como en el caso de esta tesis y un sinfín de aplicaciones. El algoritmo matemático implementado o para implementar en un algoritmo de programación es el siguiente:

Inicialización

$$E^{(0)} = r(0)$$

para $i = 1,2,..., p$

$$k_i = \frac{r(i) - \sum_{j=1}^{i-1} \alpha_j^{(i-1)} r(|i-j|)}{E^{(i-1)}}$$

$$\alpha_i^{(i)} = k_i$$

para $j = 1,..., i-1$

$$\alpha_j^{(i)} = \alpha_j^{(i-1)} - k_i \alpha_{i-j}^{(i-1)}$$

$$E(i) = (1 - k_i^2) E^{(i-1)}$$

En mucha ocasiones, es deseable combinar la estimación LPC con la utilización de los coeficientes cepstrum para la caracterización del espectro de una señal de voz. En estas situaciones es conveniente obtener los coeficientes cepstrum asociados a un determinado conjunto de coeficientes de predicción. La relación entre los dos conjuntos de coeficientes puede derivarse en base a la respuesta al impulso del filtro, resultando en la siguiente expresión:

$$c_m = a_m + \sum_{k=1}^{m-1} \left(\frac{k}{m} \right) c_k a_{m-k} \quad 1 \le k \le m \qquad \textbf{...(II.13)}$$

donde los a_m son los coeficientes LPC previamente calculados con el algoritmo Levinson-Durvin y los c_m son los llamados coeficientes LPC-Cepstral, que son los coeficientes Cepstral derivados del análisis LPC. Estos coeficientes no son iguales a los obtenidos directamente a partir de las señales de voz, pero son una muy buena aproximación y han mostrado generar un resultado semejante, además de que su obtención es relativamente aceptable.

2.7.3 Cuantización vectorial

La idea principal de la VQ, por sus siglas en inglés, es que dado un conjunto de *n* vectores en un espacio vectorial de *k* dimensiones y éste se puede dividir en *L* regiones no uniformes, en el que cada región representada por un solo vector, también de *k* dimensiones, denominado centroide o cuantizador, el grupo de vectores representados por un centroide se le denomina *clúster* y el conjunto de *L* centroides se le denomina *codebook*.

La forma en que se realiza la cuantización vectorial es a través de algoritmos de agrupamiento los cuales acomodan los grupos de vectores de acuerdo a los que presentan características similares. En este caso se utiliza el algoritmo de agrupamiento de k-medias para generara buenos resultados y por que nos permite generar de forma determinada el numero de centroides.

El conjunto de vectores a los que se les aplica el algoritmo se obtiene reuniendo todos los vectores de coeficientes (LPC o LPC-Cepstral), que corresponden a un determinado segmento de cada una de las señales de entrenamiento que representa a la misma palabra.

2.7.4 Algoritmos de k-medias

1) Escoger *K* centroides iniciales $\mathbf{z_1}(1)$, $\mathbf{z_2}(1)$, ..., $\mathbf{z_k}(1)$.

2) En la iteración *l*, asignar las muestras a los grupos:
Asignar:

$$\mathbf{x} \quad a \quad \chi_i(l) \quad si \quad d(\mathbf{x}, \mathbf{z}_i(l)) \quad \leq \quad d(\mathbf{x}, \mathbf{z}_j(l)) \quad j = 1,2,...,K \quad j \neq i$$

donde *d(x, z_j(l))* es la distancia (ó distorsión) de *Itakura*

3) Calcular los nuevos centros de grupo:

$$\mathbf{z}_i(l+1) = \frac{1}{N_i} \sum_{x \in \chi_i(l)} \mathbf{x} \qquad i = 1,2,...,K$$

donde N_i es el número de muestras asignadas a $_j(l)$.

4) Si $\mathbf{z_i}(l+1) = \mathbf{z_i}(l)$ para *i = 1, 2, ..., K,* el algoritmo ha convergido y debe terminarse. En caso contrario, regresar al paso 2.

Una característica de éste algoritmo es que los centroides o cuantizadores los cuales se obtienen por medio del algoritmo de K-medias, no son los óptimos globales; es decir, se obtienen cuantizadores óptimos locales. Estos dependen de varios factores, como son: asignación inicial de centroides (principalmente), orden de la toma de muestras, propiedades geométricas de los datos, tipo de distorsión empleada, etc... Una forma de poder alcanzar los óptimos globales, es probar con una gran variedad de centroides iniciales y seleccionar los cuantizadores finales que tengan la menor distorsión, con respecto a los vectores a los cuales representan. Solución poco factible, porque existen un gran número de combinaciones para los cuantizadores iniciales. Otra forma es, elegir los centroides iniciales de forma aleatoria, para buscar una distribución homogénea [Deller, 1987].

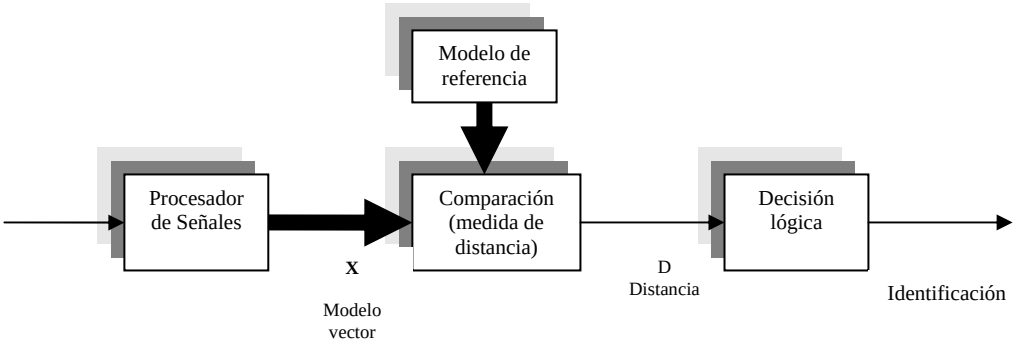

Figura II.6. Representación general del problema de reconocimiento

Los coeficientes LPC-Ceptral son uno de los parámetros más utilizados en el reconocimiento del habla, además de que han demostrado ser un método más robusto comparándolo con el algoritmo LPCC. La transformación o codificación de los coeficientes de preedición denominados como LPC-Cepstral se lleva acabo haciendo la transformada de Fourier del logaritmo de la magnitud del espectro. Considerando una secuencia, x(n), teniendo una transformada X(ω). Los Cepstral c(n), están definidos por la transformada inversa de Fourier C (ω), donde C(ω) = log X(ω).

2.7.5 Distancia Itakura-Saito

En muchos casos del procesamiento de voz, es necesario tener otra medida de la distancia que existe entre dos vectores LPC. La distancia Euclidiana no es apropiada para medir los parámetros de dos LPC's individuales, en vectores que estén relacionados. Esto es debido a que los vectores LPC dependen del peso de la matriz de autocorrelación correspondiente a cada LPC.

La distancia Itakuara-Saito es una de las distancias de máxima similitud entre un vector de coeficientes X_1 y un vector de coeficientes X_2. . Esta distancia se deriva utilizando una interpretación intuitiva del rango de predicción en el error de la energía. Fue obtenida originalmente, utilizando la máxima probabilidad existente entre argumentos similares. La distancia de *Itakura* es, probablemente, la medida de distorsión más empleada para encontrar la similitud entre dos vectores LPC [Deller, 1987].

$$d(x,y) = \log\left(\frac{xR_y x^T}{yR_y y^T}\right) \qquad \textbf{...(II.15)}$$

De la anterior ecuación se puede observar que esta distancia no es una distancia lineal entre los vectores de LPC calculados por el sistema y los centroides de nuestro *codebook*. En donde R_yx y R_yy son las matrices de auto correlación multiplicadas por la señal de voz la cual es obtenida por un convertidor analógico digital a la entrada de nuestra tarjeta de desarrollo y nuestra base de datos calculada previa mente y almacenada en nuestro sistema. xR_yx^T la energía de salida del filtro inverso, tomado como referencia con la entrada, y yR_yy^T la energía mínima posible de salida, del filtro LPC, con respecto a la entrada de la voz. Entonces tenemos que la distancia de *Itakura* se obtiene mediante la ecuación 15.

2.7.6 LPC-Cepstral

Una vez que se han obtenido los parámetros LPC, es posible obtener de estos primeros, los coeficientes LPC-Ceptral. Los coeficientes Cepstral son los coeficientes de la representación de la magnitud del logaritmo de la transformada inversa de Fourier.

Para realizar el cálculo de estos coeficientes se debe de obtener primero los LPC. Así que teniendo estos coeficientes, previamente presentado el algoritmo en la ecuación 13, a los coeficientes LPC se les aplica el siguiente algoritmo de robustecimiento.

$$c_m = a_m + \sum_{k=1}^{m-1}\left(\frac{k}{m}\right)c_k a_{m-k} \quad 1 \le k \le m \qquad \textbf{...(II.16)}$$

Además:

$$c_m = \sum_{k=m-p}^{m-1}\left(\frac{k}{m}\right)c_k a_{m-k} \quad m \ge p \qquad \textbf{...(II.17)}$$

En el reconocimiento de voz se ha comprobado que el método algorítmico LPCC, obtiene información más relevante contenida en la señal. Esta compacta representación, puede ser usada eficientemente para comparar señales. Para esta comparación ente señales ha sido definida una medida de distancia la cual es llamada Distancia Euclidiana.

Por sugerencia de Rabiner, propone un calculo extra, para mejorar el reconocimiento y robustecer el algoritmo LPC cepstral.

$$Q = \frac{3p}{2} \qquad \textbf{...(II.18)}$$

De esta manera y teniendo un nuevo coeficiente de calculo se aplica el siguiente método de calculo de coeficientes, por lo cual se obtendrán en este algoritmo no *p* puntos en el vector LPC, si no *Q* elementos. Este cálculo extra robustece nuestro algoritmo, ya que nos da más información o una información extra. Este *Q* es aplicado en la ecuación en la ecuación 17.

2.7.7 Distancia Cepstral

Los coeficientes cepstral proveen un buen cálculo de la distancia espectral entre dos campos. En el caso de los elementos o coeficientes de predicción LPC-Cepstral la distancia utilizada para el reconocimiento es la distancia Euclidiana. También denominada como error cuadrático entre dos vectores y muy utilizada para calcular la distancia entre dos vectores. La distancia Euclidiana se define como:

$$d(X_1, X_2) = \|X_1 - X_2\| = \sqrt{\sum_{j=1}^{n} \left(X_{1j} - X_{2j}\right)^2} \qquad \textbf{...(II.19)}$$

Sistema de desarrollo para el DSP

3. Sistema de desarrollo para el DSP

Un procesador de señales como lo es la familia TMS320C6x, son procesadores de propósito dedicado especializados con un tipo de arquitectura y un set de instrucciones apropiado para el procesamiento de señales. La notación C6x es utilizada para designar a la familia de DSP's de la familia TMS320C6000. La arquitectura de este circuito lógico programable esta diseñada para realizar de una manera eficiente, un gran numero de cálculos matemáticos en corto tiempo. Basado en un set de instrucciones de palabras largas, VLIW por sus siglas en ingles (Very Long Instructions Words), es considerado unos de los procesadores más poderosos.

Los procesadores de señales son utilizados en un amplio rango de aplicaciones, en la vida diaria los hemos ocupado un sin fin numero de aplicaciones en aparatos electrodomésticos, además de implementaciones de control, voz, imagen etc. Estos circuitos lógicos programables se pueden encontrar en celulares, fax/modem, discos duros, radios y más. Este procesador se ha convertido en el producto de preferencia para un sin fin de aplicaciones para el usuario común. La implementación de los DPS's ha tenido un gran éxito debido al resultado del bajo costo del hardware y el software.

3.1. Arquitectura del DSP.

3.1.1. Características de un DSP TMS320C62x/C67x

Este procesador digital de señales de la serie c6000 de Texas Instrument, maneja como un máximo de ocho instrucciones de 32 bits por ciclo de instrucción. Este dispositivo con el cual se desarrollo la tesis cuenta con un compilador de C para el desarrollo de aplicaciones y un optimizador de ensamble para simplificar la planificación y programación del lenguaje ensamblador.

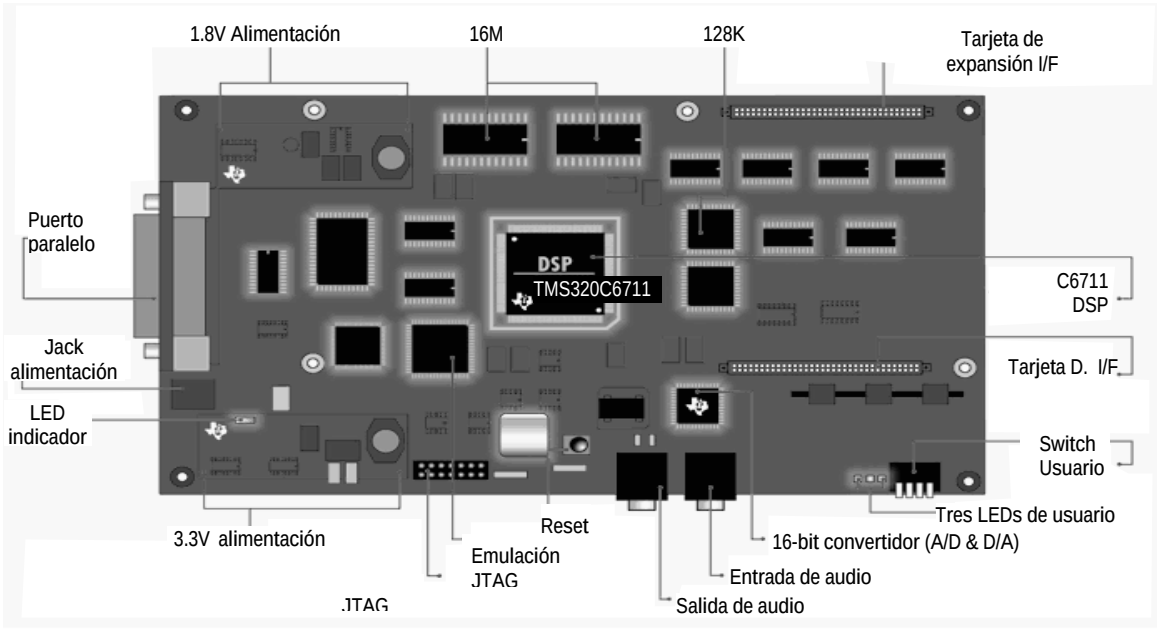

Figura III.1. Tarjeta de desarrollo DSK C6711

Esta familia de DSP cuenta con un cuenta con un CPU muy avanzado, denominado VLIW, esto es que contiene 8 unidades funcionales por medio de las cuales muchas instrucciones son capturadas y procesadas simultáneamente, además contiene 2 multiplicadores en el hardware y 6 ALU's (unidades lógico aritméticas). Lo cual hace a este tipo de componentes una buena opción para optimizar los procesos y algoritmos matemáticos programados e implementados en tiempo real. Proporciona soporte eficiente de memoria para una variedad de aplicaciones de 8, 16 y 32 bits de datos según las necesidades del usuario. Ejecuta el código en unidades

funcionales independientes. Puede manejar unidades aritméticas hasta de 40 bits adicionando resolución en nuestros sistemas de aplicación.

El rango máximo de instrucciones manejado por estos dispositivos es de 1336 MIPS lo cual nos ayuda a implementar sistemas robustos de programación, aunque hoy en día existen DSP's de la compañía Texas que alcanzan velocidades mucho mayores para este tipo de aplicaciones. También alcanza 1G FLOPS (operaciones en punto flotante) a 167 MHz. Soporta interfases para expansión de memoria externa de 32 bits (SDRAM, SBSRAM, SRAM y otras memorias asíncronas), para aumentar el rango de memoria externa y maximizar el desempeño del sistema.

Este componente fue desarrollado con una arquitectura Von Newman la cual es muy utilizada en un micro procesador solo puede hacer una lectura o escritura en la memoria durante cada ciclo de instrucción. Las aplicaciones típicas en un DSP requieren demasiados accesos a memoria por ciclo de instrucción. La memoria de programa y la de datos es separada y ambas pueden ser acezadas al mismo tiempo.

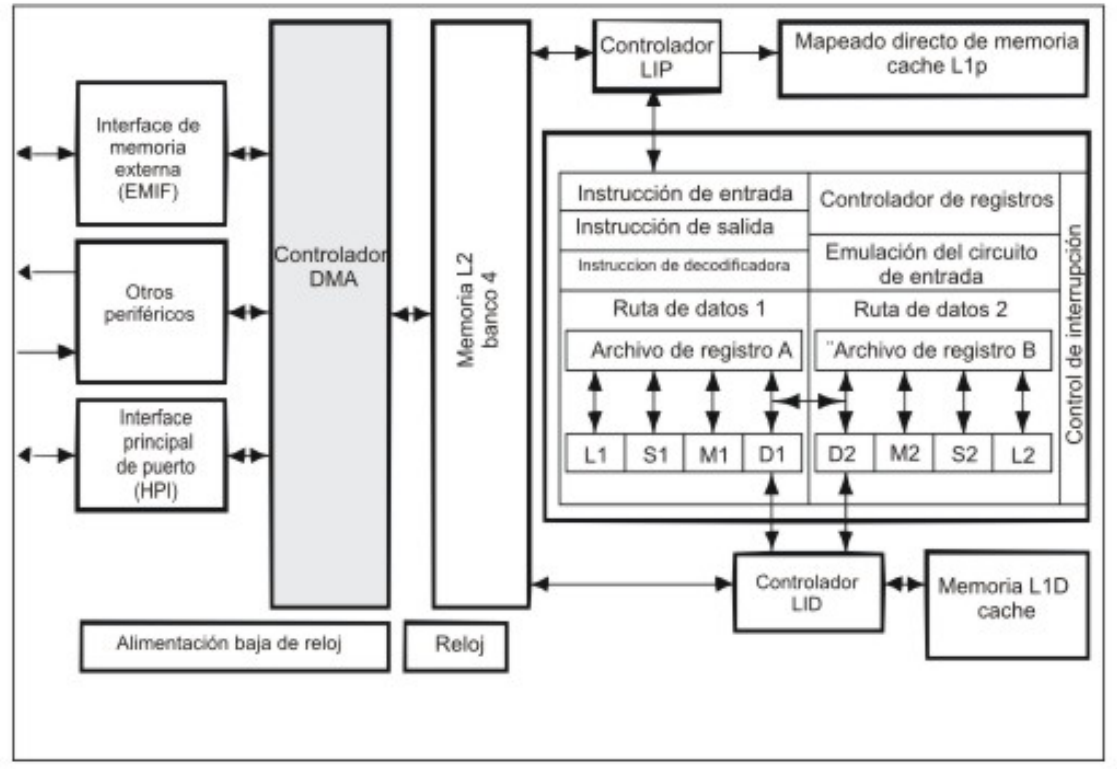

Figura III.2. Diagrama Básico de bloques del TMS320C62X/C67X

La tarjeta de desarrollo implementada para el desarrollo del sistema trabaja a 150 MHz, es de una arquitectura de punto flotante. Estos dispositivos pueden tener velocidades de reloj de 150 167, 200, 250 MHz y actualmente la familia C64xx a alcanzado la implementación de un reloj en las tarjetas de desarrollo de 1Ghz. Con este tipo de procesadores en la actualidad se pueden llevar acabo desarrollos tan complejos como celulares, televisiones de alta definición y un sin fin de aplicaciones tecnológicas e innovaciones las cuales solo quedan restringidas a la imaginación.

Estos circuitos integrados constan de tres partes: CPU, periféricos y memoria. Una de las características especiales que le da gran poder a estos tipos de procesadores es que pueden llevar acabo hasta ocho instrucciones por ciclo de instrucción, ya que consta con ocho unidades funcionales operando de manera paralela, estos son dos conjuntos similares compuestos de cuatro unidades básica cada una de ellas.

3.1.2.Bus interno y memoria

La memoria interna esta organizada en dos bancos, con dos buses independientes. Esto es con dos cargadores de instrucciones, esta es una de las razones por las cuales este procesador es tan potente, ya que en un ciclo de instrucción puede llevar acabo de manera paralela dos instrucciones. En este tipo de arquitectura no hay conflictos en el acceso de memoria aun cuando esta sea demasiado extensa, esto es que un dato puede ser acezado aun cuando este se encuentre en un campo de memoria diferente.

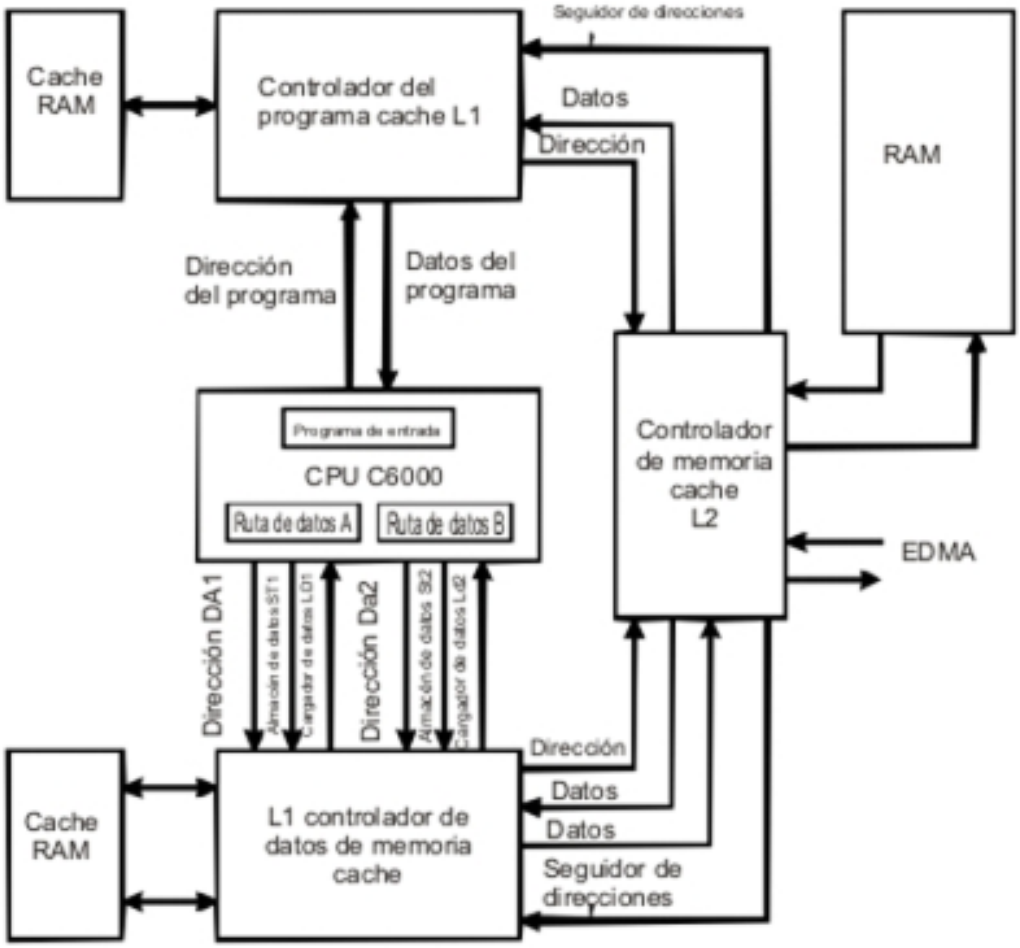

Figura III.3. Distribución de la memoria y bus interno

UNIDAD DE CONTROL

- Consta de tres elementos:
  - Unidad de búsqueda.
  - Unidad de despacho de instrucciones.
  - Unidad de decodificación de instrucciones.
- Operación segmentada.
- Cada unidad es capaz de procesar hasta ocho instrucciones por ciclo.
- Fetchpacket (FP) = paquete de 8 instrucciones.

3.1.3.Camino de datos del CPU (PATH)

Los caminos de datos del CPU consisten en dos registros de propósito general (A y B), ocho unidades funcionales (.L1, .L2, .S1, .S2, .M1, .M2, .D1, .D2), dos caminos de lecturas de memoria (LD1, LD2), dos caminos de almacenamiento (ST1, ST2) y dos caminos cruzados entre los archivos de registro (1X, 2X) y dos caminos de direccionamiento de datos (DA1, DA2).

3.1.4.Unidades funcionales

El cpu consiste en ocho unidades de procesadores, de los cuales se dividen en dos partes:

- . L Operaciones lógicas aritméticas, en punto fijo y punto flotante.
- . S Saltos, manipulaciones de bits, en punto fijo y punto flotante (ALU)
- . M Operaciones de multiplicación, en punto fijo y punto flotante (ALU)
- . D Carga, almacenamiento, y operaciones aritméticas, en punto fijo (ALU)

De esta manera puede ejecutar ocho operaciones en un ciclo de instrucción, como resultado de sus ocho unidades funcionales, seis de las cuales pueden ser en punto flotante y dos de ellas en punto fijo.

UNIDADES .L (L1 y L2)

- Operaciones aritméticas de y de comparación de 32 y 40 bits.
- Operaciones Lógicas de 32 bits.
- Normalización de enteros y cuenta de bits.
- Saturación aritmética para operaciones de 32/40 bits.

UNIDADES .S (.S1 y .S2)

- Disponen de ALUs de 32 bits y desplazadotes de 40 bits.
- Aritmética de 32 bits, operaciones lógicas y campos de bits.
- Desplazamientos de 32/40 bits.
- Saltos (.S2 solamente si se usa un registro como desplazamiento)
- Transferencias hacia/

UNIDADES M (.M1 y .M2)

- Multiplicadores hardware:
  - 16 x 16 bits con resultados de 32 bits (‘C62x).
  - 32 x 32 bits con resultados de 64 bits (‘C67x).
  - Operandos son de 32 bits => Varias opciones
  .
- Multiplicación en punto flotante (‘C67x).
- Multiplicación con desplazamiento a izquierda y saturación.

UNIDADES .D (.D1 y .D2)

- Carga y almacenamiento con desplazamiento constante de:
- 5 bits.
- 15 bits (Sólo en .D2).
- Sumas/restas de 32 bits.
- Cálculo para direccionamiento lineal y circular.

3.1.5.Archivos de registro de propósito general

Hay dos archivos de registro de propósito general (comúnmente llamados acumuladores), en los caminos de datos de la familia de procesadores de señales C6x.cada uno de los registros contiene 16 registros de 32 bits respectivamente. Este tipo de registros como en los microcontroladores, son utilizados para como registros auxiliares para llevar acabo operaciones aritméticas, para manejar o manipular datos o punteros de direccionamiento de los mismos.

Los acumuladores o archivos de propósito general soportan datos de 32 y 40 bits de punto fijo. Los datos de 32 bits pueden estar contenidos en cualquiera de los registros de propósito general; los 32 bits menos significativos son colocados en la parte baja (LSB) de nuestro

acumulador y los ocho más significativos (MSB) son colocados en la parte baja del registro próximo superior (que es siempre un registro impar). En el microprocesador utilizado en este proyecto que es de la familia C67x, se utiliza ese par de registros para utilizar en el programa valores de punto flotante de doble precisión, los cuales son de 64 bits, por lo cual necesitamos juntar dos registros para crear uno con el doble de tamaño.

3.1.6.Modos de direccionamiento

Como en las versiones previas a este procesador de señales el direccionamiento es lineal, aunque hay direccionamientos circulares. El modo de direccionamiento a utilizar se debe de especificar en el registro mapeado del modo de direccionamiento (AMR).

Con todos lo registros se puede llevar acabo un direccionamiento lineal, solo con ocho de ellos, estos registros son del A4 al A7, que son usados por la unidad .D1, la cual como se menciono anteriormente es utilizada para realizar cálculos aritméticos y del B4 al B7 que de manera análoga son parte del registro .D2.

Los procesadores de la familia C6x tiene un tipo de arquitectura denominada *carga/almacenamiento*, lo cual quiere decir que la única manera por la cual se puede acceder a la memoria es por medio de las instrucciones de carga o almacenamiento.

3.1.7.Registro de control de interrupciones

- CSR (registro de control de interrupciones)
- IER (registro de habilitación de interrupciones)
- IFR (registro de banderas de interrupción)
- ISR (registro de peticiones de interrupción)
- ICR (registro de limpio de interrupciones)
- ISTP (tabla de apuntadores del servicio de interrupción)
- IRP (regreso del apuntador de interrupción)
- NRP (regreso del apuntador de interrupción no enmascarable)

3.1.8.CSR (Control Status Register)

Esta compuesto de un bit de habilitación de la interrupción global (GIE) y otro de bit control/status.

GIE: habilitado (1) o deshabilitado (0) todas las interrupciones, exceptuando la interrupción de reset y NMI (Interrupciones no enmascarables)

Código para habilitar las interrupciones globales enmascarables.

```
MVC   CSR,B0;     cargamos CSR
OR    1,B0,B0;    listos para limpiar GIE
MVC   B0,CSR;     limpiamos el registro GIE
```

3.1.9.IER (Registro de habilitación de interrupciones)

En este registro mapeado podemos habilitar o deshabilitar las interrupciones.
Bit 0: corresponde al bit de la interrupción provocada por el reset, siempre (1)
NMIE: para habilitar el NMI

Código para habilitar la interrupción INT9

```
MVK    200h,B1 ;        prendemos el bit 9
MVC    IER,B0 ;         carga de IER
OR     B1,B0,B0 ;       preparando para prender el bit IE9
MVC    B0,IER;colocar el bit  9 en IER
```

3.2. Code Composer Studio

El CCS acelera y mejora el proceso de desarrollo para los programadores que implementan aplicaciones en tiempo real o cuando se realizan desarrollos embebidos, el diseño a implementar se facilita mas. El CCS por medio de un ambiente grafico nos da herramientas con las cuales se puede llevar acabo la configuración, construcción, depuración, análisis del programa, la implementación de mensajes del programa (tracing) por medio de los cuales podemos establecer una comunicación tipo terminal de computadora para monitorear nuestro programa en forma textual. El Code Composer Studio esta compuesto de varios elementos lo cuales se explicaran de una manera muy general en los siguientes puntos.

3.2.1.Características de la edición de programas en CCS

El CCS permite la edición de programas tanto en lenguaje C, así como en lenguaje ensamblador. También, se puede observar el correspondiente en código de un programa desarrollado en C en su correspondiente código en ensamblador.

3.2.2.Herramientas de generación de código

Los TMS320CXXX actualmente en el mercado, manejan un sistema de desarrollo en C/C++ como lenguaje optimizado a la arquitectura, así como un lenguaje ensamblador también optimizado, ligador y herramientas asociadas a ellas. Utilerias para construir librerías, librerías de soporte en tiempo real, utilerías de conversión en hexadesimal, listador de referencias cruzadas y el listrador de referencias absolutas. Para la implementación de algún proyecto el procesador utilizado en el proyecto soporta las siguientes herramientas, para el desarrollo del lenguaje ensamblador.

- Ensamblador (Assembler)
- Archivador (Archiver)
- Ligador (Linker)
- Listado Absoluto (Absolute lister)
- Listado de referencias cruzadas (Cross-reference lister)
- Utilería de conversión hexadecimal (Hex conversion utility)

3.2.3.Criterios de elección C vs ASM

- Complejidad del programa
  - En lenguaje C se facilita la programación así como el manejo de registros en comparación con el lenguaje ensamblador.
- Velocidad
  - En comparación con el lenguaje ensamblador el tiempo de ejecución de C o C++ es más.
- Numero de programadores
  - Para el desarrollo de un proyecto en el lenguaje C, se requieren menos personas ya que se facilitan las implementaciones asi como el tiempo de desarrollo.
- Prioridad costo producto / costo desarrollo

- Por lo antes mencionado, al utilizar el lenguaje C o C++ se llevan acabo de una manera más rápida.
- Además de que serian menos personas programando debido a que se facilita cuando se utiliza un lenguaje como C en comparación con ensamblador.

- Conocimiento previo
  - Ya que C es un lenguaje que la mayoría de los conocen o que deben dominar el cambio de ASM a C es mínimo.

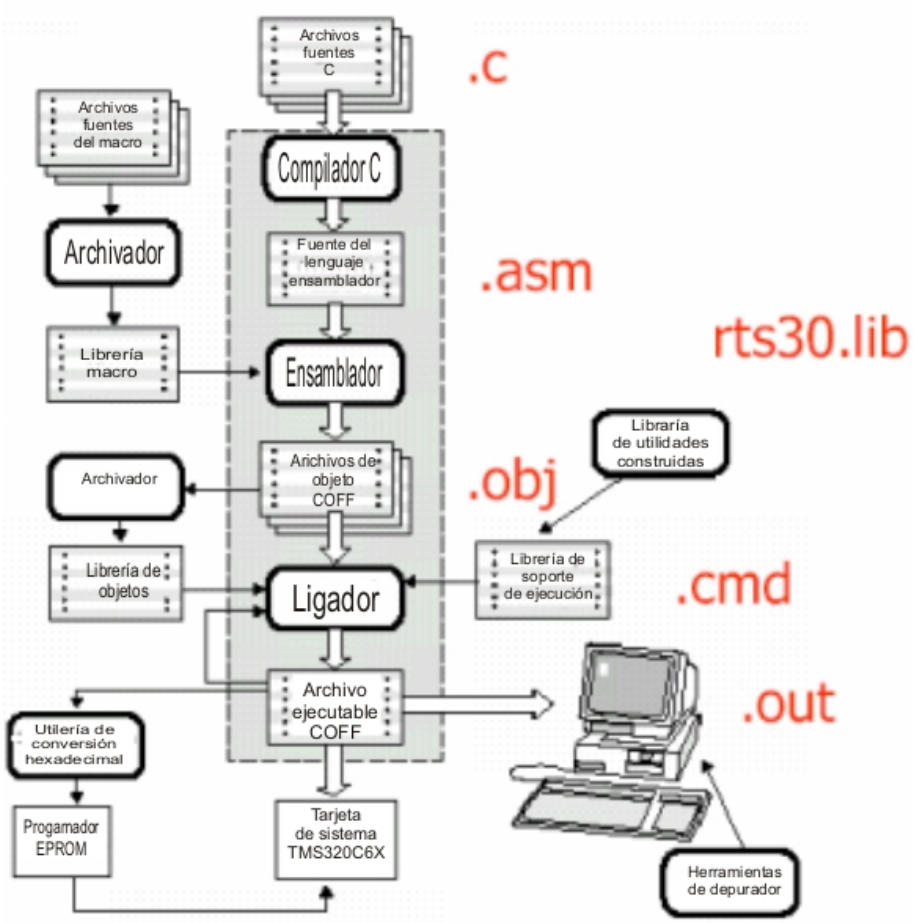

Figura III.4
Flujo para el desarrollo de programas en el procesador de señales TMS320C6000

En la figura III.4 se obsesa el desarrollo del software del TMS320C6000. La parte remarca da es el camino más común del desarrollo de un programas; las demás partes del diagramas son opcionales pero no dejan de ser importantes en algunas implementaciones, siendo estas funciones periféricas que enlazan los procesos del desarrollo

3.2.4.Descripción de herramientas

- *Ensamblador optimizado*: este tipo de lenguaje permite escribir código en ensamblador lineal sin preocuparse en la asignación de los registros. Este tipo de código asigna los registros y usa una optimización de loop's, para convertir el código lineal en código paralelo.

- *Compilador C/C++*: en este tipo de lenguaje de desarrollo el sistema de desarrollo acepta código en C/C++ y lo traslada a un lenguaje ensamblador para su respectiva plataforma o modelo de procesador.

- *Ensamblador:* esta herramienta de desarrollo traduce los programas de lenguaje ensamblador a lenguaje maquina.

- *Linker o ligador:* combina los archivos objeto a un solo archivo para el DSP, de esta manera solo se descarga este último al procesador. El ligador acepta archivos, librerías y posibles módulos de proyectos hechos previamente.

- *Archivador:* este permite reunir varios archivos en uno solo denominado librería. De manera ilustrativa, puede juntar vario macros dentro de una librería de macros.

- *Listado absoluto:* herramienta utilizada para la depuración de programas que aceptan archivos objeto ligados como una entrada y crea archivos .abs que son lenguaje ensamblador para producir un listado que muestra las direcciones absolutas del código objeto al cual esta referido.

- *Hex conversion utility:* convierte los archivo de tipo COFF, los cuales son archivos cuyo formato objeto se puede seleccionar de los siguientes: TI-Tagget, ASCII hexadecimal, Intel, Motorola-S o Tektronix. El archivo ya convertido puede ser transmitido y grabado en una memoria para su utilización en alguna aplicación.

3.2.5.Resumen y conclusión de C vs ASM

Ensamblador:

- Máximo control del procesador
- Si algoritmo complejo – Aumento dificultad
- Costoso mantenimiento y actualización de aplicaciones

C:

- Menor control del procesador
- Disminuye tiempo desarrollo algoritmos
- Menor rendimiento algoritmos
- Sencillo mantenimiento, actualización y documentación de aplicaciones

3.2.6.Entrono de desarrollo Code Composer Studio

El Code Composer Studio (CCS) es un entorno de desarrollo el cual permite editar código C y ensamblador, también se puede visualizar código en C con su equivalente en ensamblador mostrando las sentencias de C después.

El editor proporciona algunos soportes para las actividades siguientes, las cuales facilitan el desarrollo del software:

- Marca los bloques de programación en lenguaje C en corchetes y paréntesis
- Resalte de palabras clave, comentarios y cadenas en el código
- Obtener ayuda sensible al contexto
- Niveles de sangría búsqueda y reemplazo en uno o mas archivos
- Deshaciendo y rehaciendo una o mas acciones

Características de construcción y aplicaciones

Como en casi todos los entornos IDE, en este no es por de más mencionar que se pueden crear proyectos de trabajo, los cuales son utilizados para la construcción de una aplicación. Como se ha mencionado anteriormente, pueden ser desarrollados en diferentes lenguajes, debido a los conocimientos y recursos con los cuales se cuente, así como la optimización que se desee en el proyecto; además en el proyecto se pueden anexar librerías, objetos, archivos, archivos de comando del linker (ligador) o archivos incluye.

3.2.7.DSP/BIOS

El DSO/BIOS es un pequeño *finware* que corre sobre el procesador digital de señales. Este software permite a los desarrolladores tener las siguientes dos capacidades:

- Monitoreo y control en tiempo real: esto es el monitoreo de nuestras variables así como el control de estas en el comportamiento de nuestro programa en tiempo real.
- Optimización en tiempo real. Esto es un sistema planificado de multi-hilos o multiprocesos, optimizado.

Los plug-ins del CCS proveen con el DSP/BIOS un monitoreo y análisis del procesador además de un análisis de nuestra aplicación en tiempo real. Puede utilizarse para que de una manera visual por medio de una interfaz grafica se lleve acabo las pruebas y monitoreo de una aplicación realizada en nuestro DSP con el mínimo impacto en el desarrollo de la aplicación.

- *Trace*: despliega eventos escritos a registros asignados y refleja dinámicamente el control de flujo durante la ejecución del programa.
- *Monitoreo de ejecución*: rastrea estadísticas que reflejan el uso de los recursos.
- *Archivos de flujo*: ligador de archivos en la PC a objetos de Entrada/Salida en el programa residente

Configuración del DSP/BIOS

En el entorno de desarrollo CCS el usuario puede crear sus propios archivos de configuración por medio de los cuales definen objetos que son utilizados en las API's, estos archivos pueden facilitar el mapeo de memoria y el de los vectores en las rutinas que atienden las interrupciones.

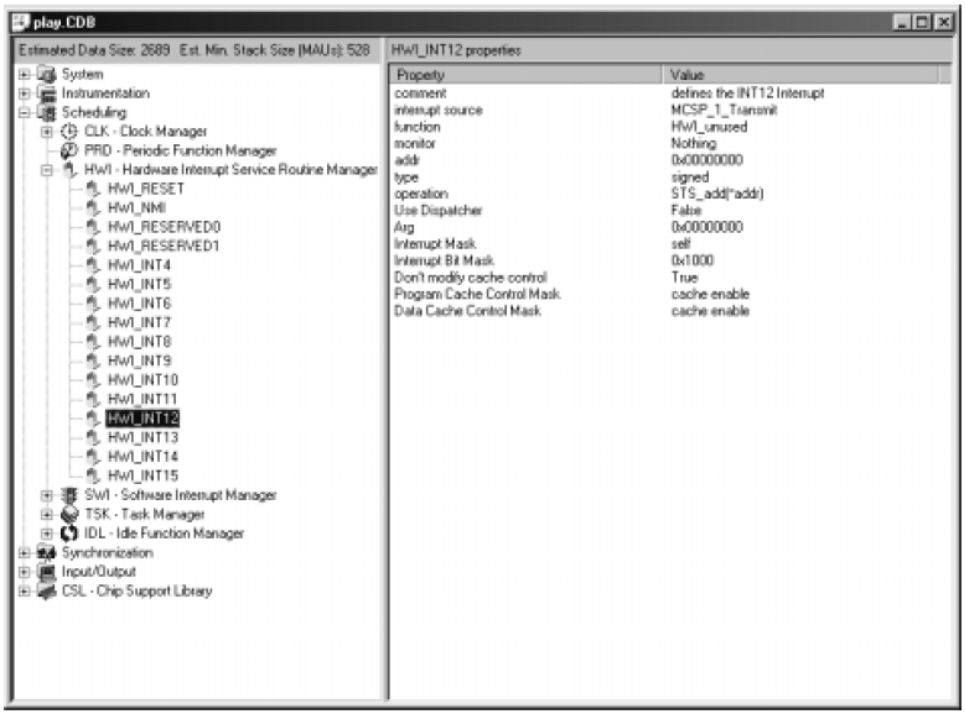

Figura III:5. Ventana de configuración de los objetos del DSP/BIOS

Cunado se abre un archivo de configuración del DSP/BIOS en el CCS, este muestra una ventana en la cual de manera grafica nos permite establecer y crear propiedades para objetos de tiempo real que son utilizadas en la denominadas API's DSP/BIOS. Estos eventos, los cuales son manejables desde esta ventana incluyen interrupciones en tiempo real por software, tuberías de I/O, eventos LOG, etc.

Módulos del DSP/BIOS

Módulos para el sistema de configuración	
GBL	Manejador de ajuste global
MEM	Manejador de memoria

Módulos para el monitoreo y control en tiempo real	
LOG	Manejador de mensajes log
STS	Manejador del acumulador estático
TRC	Modulo Trace para enviar mensajes a la pantalla de depuración
Módulos de planificación en tiempo real	
HWI	Modulo de interrupciones de hardware
SWI	Modulo de interrupciones de software
IDL	Modulo de funciones Idle y modulo de procesos de lazo
CLK	Modulo manejador del sistema de reloj
PRD	Modulo de funciones periódicas, que permite activar funciones cíclicas
Módulos de comunicación en tiempo real	
PIP	Modulo de tubería de datos (pipe), en este se implementan rutinas de flujos de datos de entrada y salida
HST	Modulo de host entrada/salida, este proporciona soporte a las rutinas de hardware
RTDX	Manejador de flujo de datos en tiempo real entre la PC y al tarjeta de desarrollo

3.3. Problemas en tiempo real

Después de la construcción del proyecto, se puede utilizar Execution Graph in el CCS, para tener de manara grafica un *Display* en el cual se pueden visualizar el comportamiento de los Threats o hacer una revisión del comportamiento estático de los mismos en un momento determinado. Para utilizar la aplicación denominada Execution Graph en necesario que el thread “logging” sea habilitado en el RTA del panel de control. Para utilizar la Static View para extraer la información en el tiempote ejecución o en un instante el thread “accumulators” del panel de control RTA debe de ser habilitado. Es importante mencionar que esta aplicación solo da información de los procesos en tiempo real realizados en un cierto tiempo. Es importante que el usuario determine si el programa da la información correcta o se acerca ala realidad los datos arrojados por el mismo o no.

3.4. Intercambio de datos en tiempo real

Los DPS de la familia de Texas Instrument C6x proporcionan emulación en el chip, habilitadas por el CCS, para la ejecución de programas de control y monitoreo de la actividad del programa en ejecución como de ha mencionado anteriormente.

El hardware de emulación proporciona una variedad de capacidades:

- Iniciación, detención o reset del DSP.
- Carga de código o datos del DSP.
- Intercambio de datos en tiempo real (RTDX) entre el host y el DSP
- Examina registros o memoria del DPS
- Puntos de ruptura o interrupción del programa o dependencia de datos.
- Soporte de conteo, incluyendo perfiles exactos de cliclos

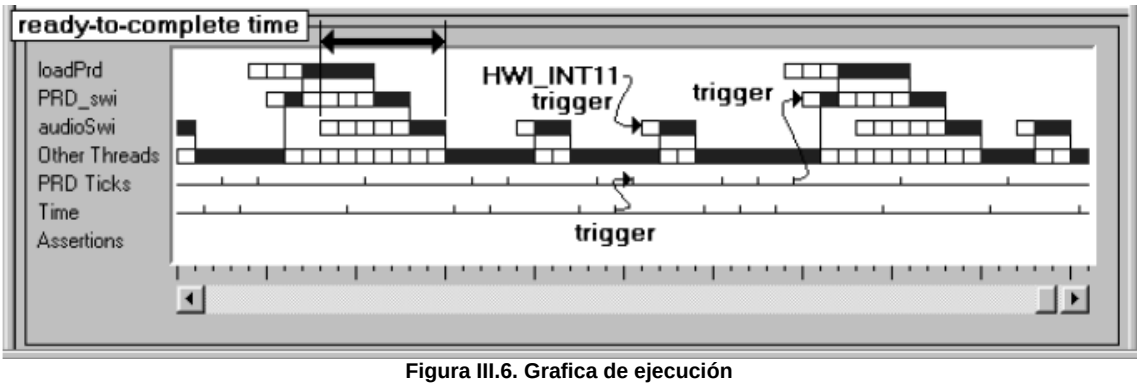

Figura III.6. Grafica de ejecución

Este tipo de sistema (RTDX) permite, visibilidad continua en el proyecto de la operación de las aplicaciones, además de transferir datos entre la computadora host y la tarjeta de desarrollo, sin parar la aplicación en proceso. Este proceso permite al desarrollador un ahorro de tiempo muy significativo ya que las señales de entrada al DSP pueden ser suministradas por medio de la PC, de esta manera se acorta el tiempo de diseño, ya que las simulaciones se pueden ser mas realistas y apegadas a la operación real del diseño en el caso de que las señales de estrada simulen el sistema en operaciones reales.

Sobre la plataforma host corre una librería RDTX opera en conjunto con el CCS. Las herramientas de análisis y despliegue, pueden comunicarse con el RDTX por medio de un API COM, la cual obtiene y envía datos al la tarjeta de desarrollo por medio del puerto de comunicación. Los diseñadores pueden utilizar paquetes estándar de software, tales como Lab View de National Instrument, además de una posible aplicación de herramientas de graficación en tiempo real tales como Quinn-Curtis o Microsoft Excel.

El RDTX también puede grabar datos en tiempo real y representar estos datos en tiempo no real. De esta forma en un programa como Matlab se les puede dar un sentido a los datos e interpretar los resultados obtenidos, además de tener una representación grafica de la misma.

Proyecto

4 Proyecto

4.1 Descripción

De una manera muy general se describió la metodología que se siguió para llevar acabo la construcción de un sistema manejado por voz. En este capitulo se abordara mas detalladamente los componentes, metodología, algoritmos y demás que se implementaron, en el desarrollo del proyecto y en su respectiva programación. Como ya se menciono en la información presentada anteriormente, este proyecto fue realzado con algoritmos matemáticos y un método de reconocimiento de voz denominado VQ (cuantización vectorial) y por métodos de codificación de voz para el reconocimiento de este como el LPC y LPC-Cepsttral. Que de una manera mas explicita se vera como de manera "física" fue implementada en nuestra tarjeta de desarrollo del procesador de señales de la compañía Texas Instrument.

El proyecto en un principio se desarrollo en lenguaje Matlab, esto fue con el cometido de llevar acabo un entrenamiento, así obtener una base de datos la cual se pudiera bajar al DSP, ya que de esta manera se facilitaría la programación y se tendría mas control de la base y del entrenamiento.

Las señales de voz fueron obtenidas por medio de un algoritmo de inicio-fin (**Rabiner** y **Sambur**), el cual se puede consultar en esta tesis. Con este algoritmo se recorto la señal de manera automática, además de que se le aplicó un filtrado y de manera consecutiva se realizo el proceso de entrenamiento en Matlab obteniendo un numero de centroides los cuales son los vectores representativos de todas nuestras palabras grabadas. Las palabras entrenadas para cada conjunto de centroides fueron cuatro con veinte repeticiones cada una.

El desarrollo fue algo complejo pero se explicara de manera explicita y entendible en los apartados contenidos en este capitulo, tratando de no dejar dudas para sus posteriores modificaciones o actualizaciones, ya que este campo en demasiado extenso y apenas empieza a nacer por lo cual le queda un gran campo de investigación. Las implementaciones del procesamiento de voz son tan variados que solo quedan limitados a la imaginación del diseñador, además de que las investigaciones continúan desde que a una persona se le ocurrió que con unos tubos conectados entre si, podían simular la vos humana, por lo cual nació el primer sintetizador de voz y la idea de que se podía continuar desarrollando en esa área, con la creciente invasión de los sistemas digitales y las computadoras se vio en la necesidad de desarrollar algoritmos matemáticos, los cuales han demostrado ser muy controlables, manejables y manipulables a comparación de los sistemas analógicos, aun cuando estos últimos tengan sus ventaja también.

Uno de los grandes avances del reconocimiento y síntesis de voz dio un gran paso en sus desarrollos cuando surgen los sistemas programables como son los circuito programables y las computadoras. En este tiempo se desarrollan algoritmos pensados en comprensión y análisis de la voz. Es cuando se observa que la voz tiene un comportamiento semialeatorio, aun y cuando la misma palabra sea pronunciada por la misma persona tiene pequeñas variaciones entre ellas, esa es una de las razones por las cuales en algunos algoritmos de reconocimiento y síntesis se ocupa la autocorrelacion en muchas de las aplicaciones.

En esta escrito se presentara el código en Matlab ya que a mi parecer es un poco mas simple y demostrativo para el entendimiento de los procesos, ya que el cometido de es tratar de pasar el conocimiento y que sirva posteriormente de una referencia. Así que se tratara de hacer una comparación también entre la implementación en tiempo real que se llevo acabo en el lenguaje de programación C y su implementación el lenguaje de Matlab, en el cual se utilizo para el entrenamiento del sistema, para sacar la base de datos y de esa manera poder realizar la comparación de las señales de voz entrantes y las obtenidas previamente.

4.2 Algoritmos

En el siguiente apartado se describirá el sistema de entrenamiento de palabras que fue programado en lenguaje propio de Matlab. Esta etapa fue muy importante para el diseño e implementación del proyecto de reconocimiento ya que es la base del programas de reconocimiento realizado en el DSP. Los programas y funciones presentados a continuación son en conjunto el programa de entrenamiento. Por medio de este programa se obtuvieron los centróides correspondientes a cada conjunto de palabras a entrenar. El sistema de reconocimiento tiene el mismo concepto que el de entrenamiento, las diferencias son que uno se programo para que funcionara en tiempo real en la tarjeta de desarrollo y el sistema de entrenamiento no, ya que este fue programado para leer archivos de audio previamente grabados por medio del circuito programable (DSP).

4.2.1 Filtro Pre-emfasis

El propósito de este filtro es darle una ganancia mayor a las componentes de mas alta frecuencia que a las componentes de menor frecuencia, con el puposito de aplanar el espectro de la señal; como se puede apreciar en la figura IV.1.

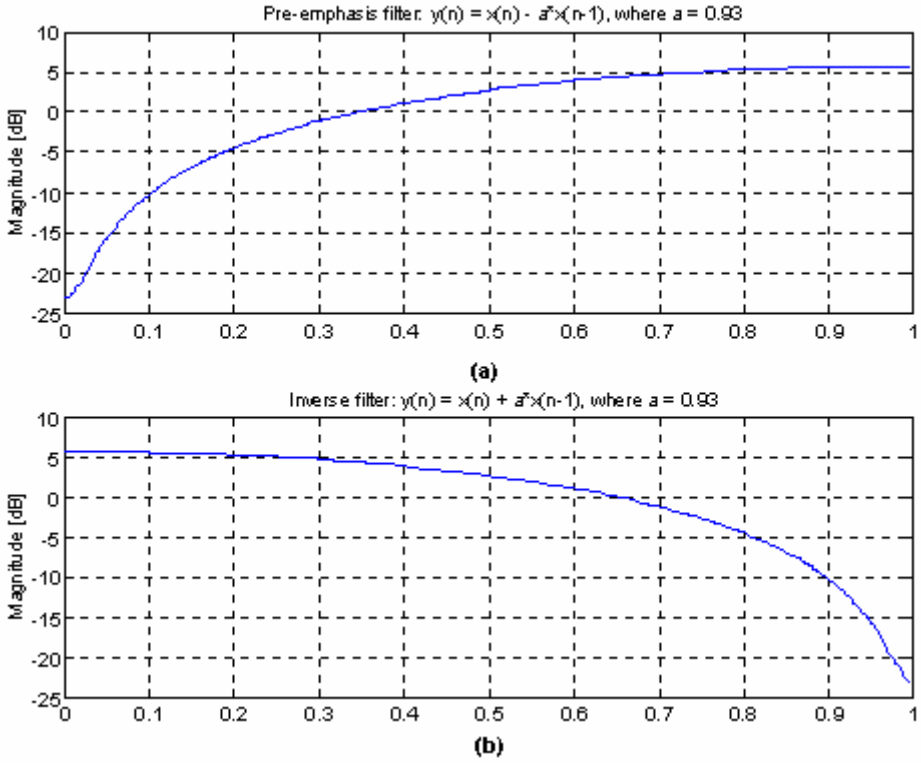

Figura IV.1. (a)Repuesta en frecuencia el filtro pre-énfasis y (b) la inversa del filtro.

Para tener una idea del efecto producido este filtro pre-énfasis se muestran continuación dos graficas. La primera muestra el espectro de frecuencia de la vocal */i/* en la palabra *nine*. En esta grafica se muestra el espectro en frecuencia de esta vocal antes y después respectivamente, de la aplicación de este filtro. Como se puede observar en el espectro, ocurre una suavización en su representación en frecuencia. Esta aplicación de filtrado facilita el calculo de los coeficientes LPC. En la grafica posterior, la cual es una representación de la señal de los coeficientes LPC en el dominio de la frecuencia, se pueden observar picos mas visibles en las frecuencias mas altas del espectro[10]. Claramente, como se puede observar en la grafica, los componentes de mayor frecuencia tienen mayor estimación al momento de aplicar el algoritmo Levinson-Durbin, que fue el método utilizado para el calculo de LPC's.

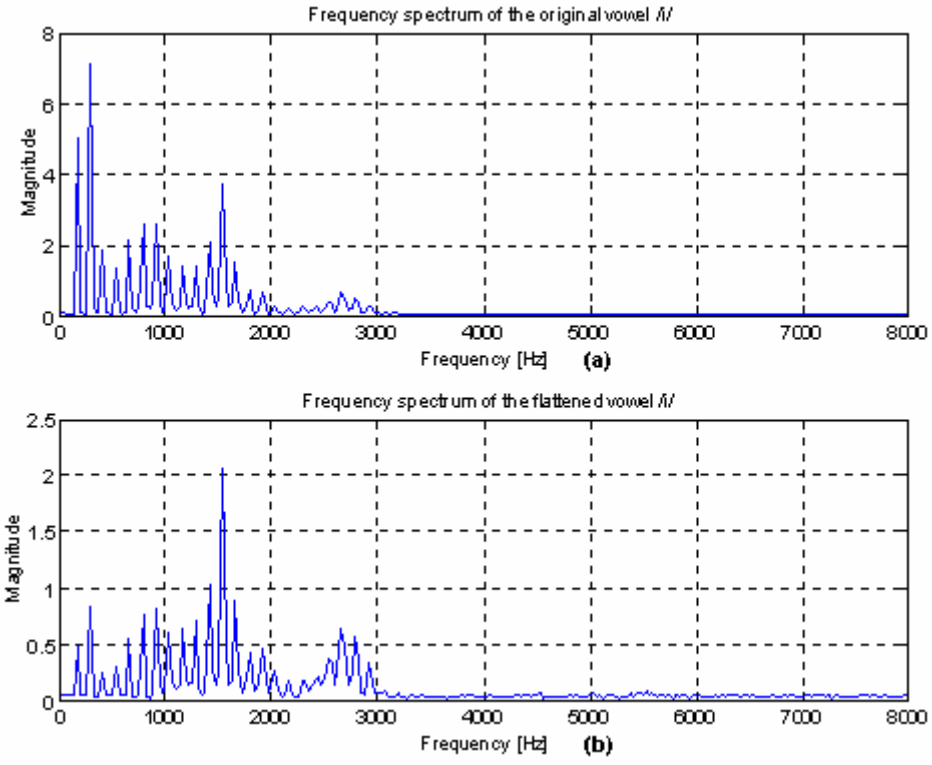

Figura IV.2. (a) Espectro en frecuencia de la vocal /i/ en la palabra nine, (b) Espectro en frecuencia con el filtro preenfasis de la vocal /i/ en la palabra nine

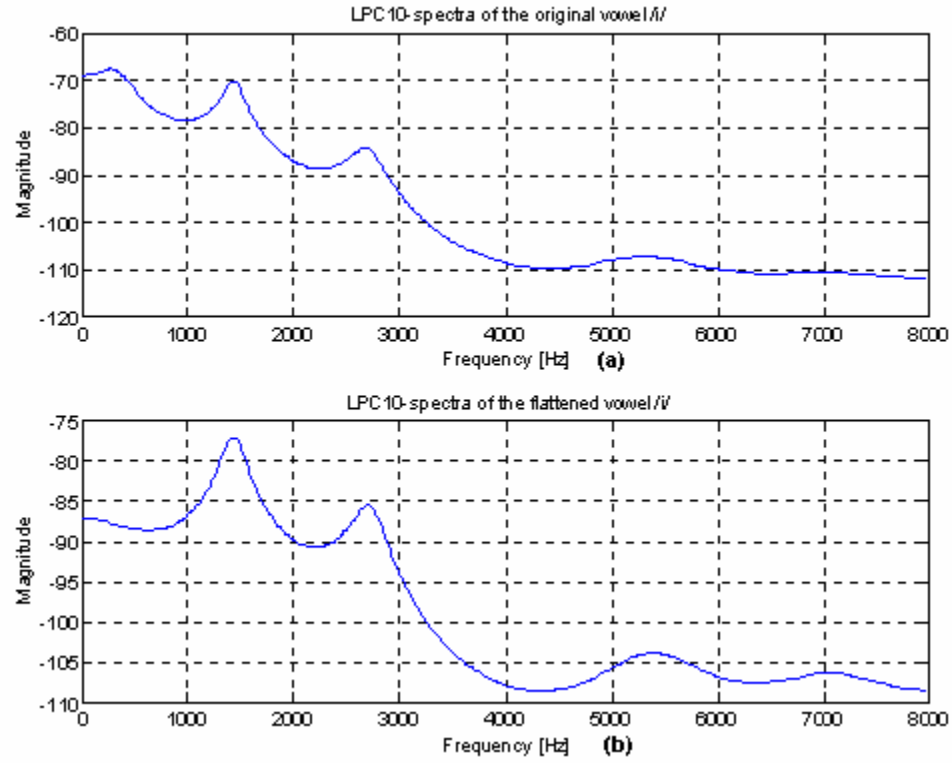

Figura IV.3. (a) Espectro en frecuencia del modelo LPC para la vocal /i/, (b) el en frecuencia del modelo LPC para la vocal /i/ con el filtro prenfasis en la palabra nine.

4.2.2 Autocorrelación

La autocorrelación es indispensable para el algoritmo de cuantización vectorial el cual es una de las entrada para el calculo previo de los coeficientes de predicción lineal, LPC por sus siglas en Ingles, el cometido esta función es conocer la relación que tienen las señales entre si,

```
function [auto] = autocor(x,p)
 %Calcula p coeficientes para la autocorrelación [para modificarlo cambiar todos los (p) por (p+1)]
  mat=zeros(length(x),p);
  n=length(x);

  for k=1:p
     mat((n*(k-1)+1):(n*(k-1))+length(x)-k+1)=x(k:length(x));
  end

  auto= x*mat;
```

Tabla IV.1. Código en MATLAB del algoritmo de autocorrelación.

La función de autocorrelación proporciona una medida de la relación de la señal con una copia desfasada de sí misma. Se va a definir a *p* como el orden de análisis[11].

Para el calculo de esta función, la señal de entrada debe de tener ciertas consideraciones para que el calculo de esta sea adecuado. Por lo cual se deben de tomar las siguientes consideraciones para garantizar el análisis de los coeficientes LPC.

- Consideraciones para el calculo de la autocorrelación
  - Requiere un ventaneo de la señal; compensación entre resolución espectral y resolución de tiempo.
  - Requiere una muestra de entrada de la señal mayor a 20 ms.
  - Además de que el filtro debe de ser estable.

4.2.3 Algoritmo de reconocimiento de voz implementado en el sistema

El sistema implementado en este proyecto de tesis fue el de cuantización vectorial implementando coeficientes de predicción lineal. Con los algoritmos explicados en el capitulo 1 y en este el lector puede darse una idea de cómo o cuales son lo requerimientos y el funcionamiento de manera que el entendimiento del funcionamiento del proyecto sea claro y digerible.

En primera instancia lo que se realizo la fase de entrenamiento de palabras en el cual se ocupo un meto de calculo de *centroides* denominado k-medias el cual por medio de iteraciones agrupa por secciones los LPC mas cercano y como dice la palabra "*centroides"*, es el concepto análogo utilizado en mecánica clásica el cual es un punto de equilibrio en una masa; en nuestro caso el punto de equilibrio se pondría interpretar como un agrupamiento en el cual hay una distancia mínima entre un grupo de vectores y se encuentra agrupados en un conjunto en donde el centro armónico es el *centroide* calculado por este método llamado k-medias.

Cuantización vectorial del LPC

La representación LPC nos proporciona una serie de vectores que reflejan las características del espectro variando en el tiempo de una señal de voz. Para reconocer una señal de voz se puede comparar una serie de vectores a reconocer contra una serie de vectores de entrenamiento y ver

que tanto se “parecen". Sin embargo la comparación de la serie de vectores a reconocer contra los de entrenamiento resulta impráctica ya implica el manejo de demasiados datos: Los vectores de entrenamiento de varias palabras requieren un espacio de almacenamiento muy grande y su comparación sería muy tardada. Por lo tanto se efectúa una cuantización vectorial al conjunto de vectores LPC de entrenamiento. Esta cuantización reduce el espacio de almacenamiento de los vectores de entrenamiento asignando a cada vector un vector aproximado. De esta forma obtenemos un libro de códigos donde un vector del libro de código puede representar varios vectores de entrenamiento. En consecuencia se reduce el espacio de almacenamiento y el número de vectores con el cual tenemos que comparar un vector a reconocer. De esta forma se disminuye el tiempo computación necesaria para en el análisis de la similitud de los vectores espectrales.
Para hacer una cuantización vectorial se necesita de :

1. Un gran conjunto de vectores de análisis espectral.
2. Un medida de similitud, o distancia, entre un par de vectores de análisis espectral.
3. Un procedimiento computacional para encontrar un centroide.
4. Un procedimiento de clasificación para vectores arbitrarios de análisis espectral de voz que clasifique el vector del libro de códigos que sea mas cercano al vector de entrada y que use el índice del libro de códigos como la representación espectral resultante.

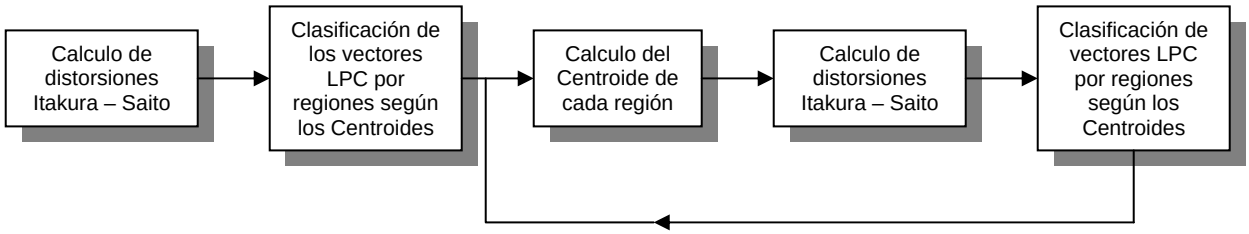

Figura IV.4. Diagrama de bloques del algoritmo de cuantización vectorial

Calculo de la distorsión Itakura - Saito

Este distancia es no lineal comparándola con la común mente ocupada en la distancia entre dos puntos que es la distancia euclidiana. Esta medida de distorsión Itakura – Saito, es la proporción de la energía residual producida por la señal de voz original cuando es utilizada la codificación LPC para el procesado de la señal[12]. La medida de distancia Itakura-Saito es muy sensible a la variación de localización de formantes en nuestra señal y es menos sensible a las variaciones de los valles en el espectro de frecuencia.

El funcionamiento del método de reconocimiento se basa principalmente en la comparación de dos espectros en tiempo corto. Para llevar acabo el reconocimiento de la palabra se debe de tener una medida de distorsión en nuestro sistema. Existen muchas variantes de esta medida, pero todas estas variantes utilizan como una de las entradas el vector de LPC’s de la palabra y los centroides de una base de datos previamente calculada.

En el sistema de entrenamiento que fue realizado en Matlab, en el cual se llevo acabo el calculo de centroides, para ser cargados en la tarjeta de desarrollo, fue programada como una función que calcula la medida de distorsión Itakura – Saito, la cual recibe como argumentos, el vector de auto correlación de la señal de entrada, el vector de autocorrelación del segmento de la señal de entrada. De esta manera teniendo estas entradas como variables de entrada de nuestra función programada y lista para ser invocada, se segmenta la señal de manera lineal, recordando que nuestras señales de entrada son archivos de audio que en Matlab pueden ser leídos de manera muy sencilla con el comando “wavread”. Este tipo de distorsión es no lineal como se puede observar, tanto en el programa como en la ecuación presentada en el capitulo 2, se puede decir que es logarimica.

```
function [DISTANCIA] = ditakura(XLPC,RX,YLPC)
%Calcula la distancia entre los vectores
%RX debe ser de longitud (p+1)

p=length(XLPC);
xaumentado=ones(1,p+1);
xaumentado(1,1)=-1;
xaumentado(2:p+1)=XLPC(:);

yaumentado=ones(1,p+1);
yaumentado(1,1)=-1;
yaumentado(2:p+1)=YLPC(:);

xRxt=xaumentado*toeplitz(RX)*xaumentado';
yRxt=yaumentado*toeplitz(RX)*yaumentado';

if xRxt ~= 0 & yRxt ~= 0
   DISTANCIA=abs(log10(yRxt/xRxt));
end
```

Tabla IV.2. Código en MATLAB del algoritmo Itakura-Saito

Función de codificación LPC

El calculo del vector de codificación LPC se llevo acabo con una entrada de señal de voz previamente grabada por medio de la tarjeta de desarrollo y recuperada de la memoria de esta por medio del programa Code Composer Studio, estas fueron almacenadas en archivos de audio, por medio de la instrucción de Matlab, “wavwrite”. La forma general de esta función se muestra a continuación:

wavwrite(y,Fs,N,filename)
en donde:
y: Variable o vector a grabar en el archivo de audio
Fs: Frecuencia a la que se ha de grabar el archivo de audio .wav
N: Número de Bits, 8-, 16-, 24-, y 32-bit
Filename: nombre del archivo

El archivo se grabará en la carpeta de trabajo en donde se encuentra el usuario o el programador. El numero de resolución 8-, 16-, y 24-bit son del tipo modulación por pulso o PCM y a 32-bit son escritos en el tipo punto flotante normalizado. Para el grabado de estos archivos se escogió la codificación de punto flotante, esto es debido a que el grabado se realizo en este formato, así que para conservar toda la información posible de nuestra señal se decidió conservar el mismo formato de codificación para que la toma de decisión.

Uno de los métodos mas utilizados en el cálculo de los coeficientes de LPC y que tiene muy buenos resultados es el algoritmo de Levinson-Durbin. Así como se programo en la parte de entrenamiento fue ocupado en la fase de reconocimiento, a esto me refiero a que el programa que se presenta a continuación en esta sección, se programo de manera análoga en el DSP, solo que en otro lenguaje, lo cual no varia demasiado, solo un poco de sintaxis lo cual hizo mas fácil su migración y puede hacer mas su entendimiento en el caso de no manejar alguno de los lenguajes de programación.

```
function [A] = LD(r, p)

    E = r(1);
      for i=1:p
            sum = 0;
            for k=1:i-1
                  sum = sum + ant(k)*r(i-k+1);
        end
            K(i) = (r(i+1) - sum)/E;
            coef(i)  = K(i);
            for k=1:i-1
                  coef(k)= ant(k) - K(i)*ant(i-k);
        end
            E = (1-K(i)^2)*E;

        ant = coef;
    end

    A = coef;

return
```

Tabla IV.3. Código en MATLAB del algoritmo LPC-Rabiner

4.2.4 Calculo de centroides por medio del algoritmo matemático K-medias

El cálculo de centroides es una herramienta primordial para la aplicación. Este paso es para obtener una base de datos que pueda ser comparada con una señal de entrada a nuestro sistema. De manera que cuando se va a llevar acabo el reconocimiento, por medio de la distancia de distorsión Itakura-Saito, se lleva acabo una comparación y se obtiene por áreas la distancia mas cercana a nuestro conjunto de vectores. En el momento de realizar todos estos cálculos, nuestra señal de entrada, la cual va a ser comparada, fue previamente segmentada de manera lineal en 4 y 8 partes.

Como ya se ha mencionado el modelo acústico que se ocupo para el reconocimiento en el algoritmo de reconocimiento, fue el llamado VQ o cuantización vectorial. Este algoritmo se utiliza en la comprensión de señales, codificación de imágenes y del habla, además de que se puede transmitir mayor información que en la cuantización escalar. Este método se ha utilizado en los modelos acústicos digitales desde principios de los 80's. Una de las características principales de VQ, es que esta basada en algoritmos de agrupamientos como es el de K-medias.

El método denominado como K-medias es algoritmo iterativo, que su cometido principal es el reunir datos en conjuntos, cada conjunto que exista debe de contener al menos un elemento en su universo. Este algoritmo iterativo converge en un óptimo local. A esto me refiero que el punto de equilibrio de nuestro conjunto es un punto equidistante de todos los elementos que contiene. Así que cuando se tiene un conjunto de vectores que entran al algoritmo de K-medias, como la palabra lo dice se debe dar un valor a la variable K, y de aquí se parte a crear K conjuntos y obtener el punto equidistante del mismo. De esta manera se obtienen los centroides y de esta manera so obtiene también la base de datos que necesitamos para nuestro proyecto.

En general, se desconoce el número de grupos. La selección de K para un buen funcionamiento depende del criterio de agrupamiento, espacio, computación, requisitos de distorsión, o en la métrica del reconocimiento. Al decir métrica de reconocimiento nos referimos a la distancia de Itakura-Saito la cual es una distancia que no es lineal. Esta métrica que se ocupa en el algoritmo de K-medias es indispensable para el agrupamiento que se lleva acabo, además de que es una

parte importantísima para encontrar las distancias equidistantes entre nuestro conjunto de muestras o en este caso, distancia equidistante entre nuestros vectores de voz.

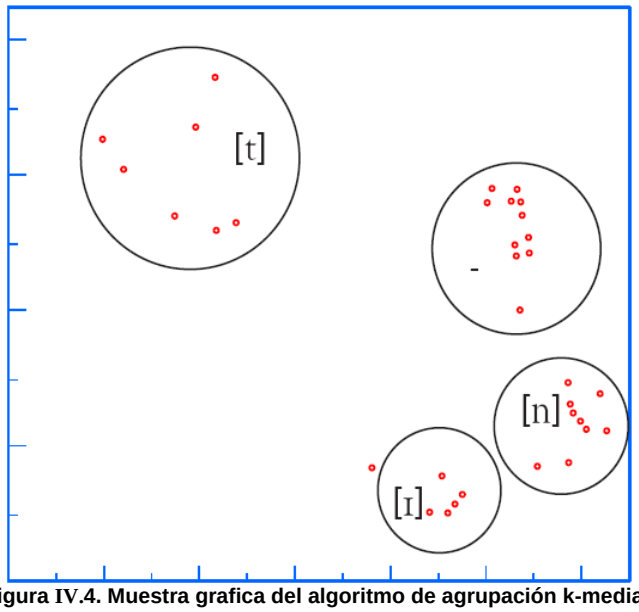

Figura IV.4. Muestra grafica del algoritmo de agrupación k-medias

Como muestra la figura IV.4, los conjuntos formados por el algoritmo pueden contener *n* elementos en su universo. K medias converge en un óptimo local de nuestro conjunto de elementos contenidos, pero este óptimo local no esta garantizado. El cálculo de los elementos óptimos no esta garantizado ya que este método es estadístico, de manera que definido a calcular un conjunto K de centroides, estos no pueden ser los mismos cuando el programa los calcula en tiempos diferentes, esto quiere decir que si el programa se ejecuta una vez y obtiene un conjunto de centroides a la segunda vez que se ejecuta se obtendrán un segundo conjunto de centroides posiblemente muy distinto al anterior ya que nuestro método es estadístico.

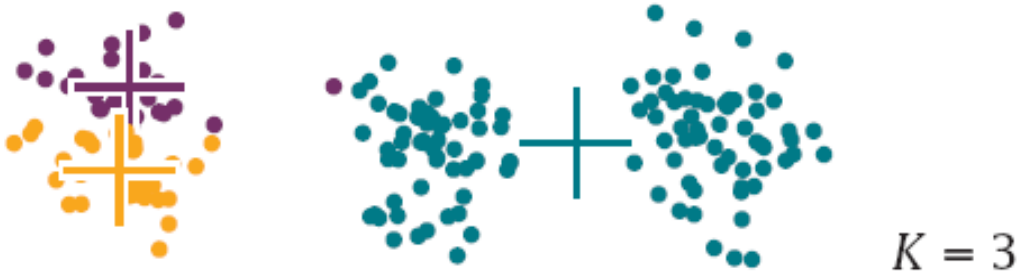

Figura IV.5. Las elecciones iniciales de nuestros centroides pueden influir en nuestro resultado final

Un k medias inicial puede ser seleccionado de manera aleatoria, pero dependiendo de si este conjunto se seleccionó correctamente de manera inicial se pueden tener mejores resultados en el reconocimiento. En nuestro programa los vectores equidistantes seleccionados de manera inicial no se escogieron de manera arbitraria, estos fueron seleccionados de nuestro conjunto de vectores de las palabras de voz.

Como se puede observar en la figura IV.5, al escoger un conjunto de centroides de una manera no óptima, nos puede llevar a problemas como los que se observan en esta figura, como es que los centroides se encuentren a una distancia tan próxima. En la practica nos esto nos puede llevar a confusiones de reconocimiento de proximidad, ya que como se encuentran muy cercanos estos puntos y otro es el punto equidistante de dos conjuntos que podrían estar por separado, al momento del reconocimiento nuestro sistema llega a una conjunción, de manera que si se asignan de manera inicial puntos que puedan optimizar el método se llega a un reconocimiento mejor.

Una métrica de distorsión tiene por lo menos las siguientes propiedades:

1. $0 \leq d(\mathbf{x}, \mathbf{y}) \leq \alpha$
2. $d(\mathbf{x}, \mathbf{y}) = \alpha$ solo si $\mathbf{x} = \mathbf{y}$
3. $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$
4. $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{y}, \mathbf{z})$
5. $d(\boldsymbol{x} + \boldsymbol{z}, \boldsymbol{y} + \boldsymbol{z}) = d(\boldsymbol{x}, \boldsymbol{y})$

En la práctica, es posible que la métrica de distancia no obedezca a alguna de estas propiedades, pero son una medida de disimilitud.

4.2.5 Programa de reconocimiento en Matlab

Para ver el funcionamiento de el reconocedor y su eficacia previamente antes de ser probado en tiempo real, se realizo un programa prueba en el cual se vio el funcionamiento de este. En el programa se ocupa lo programado previamente. Como este programa se realizó con las palabras de entrenamiento se obtuvo un reconocimiento de un 100%, pero este porcentaje de reconocimiento no es el real que se obtuvo ya en la aplicación y en la practica, se tienen muchos factores que en tiempo real afectan el reconocimiento ya aplicado en tiempo real. Uno de los factores que influyen es el descenso de nuestro como ya se a mencionado repetidas ocasiones en este escrito es el ruido, además de algunas pequeñas variaciones en la pronunciación entre repetición y repetición de la palabra, por este motivo los algoritmos utilizados. Estos algoritmos como es el de autocorrelación, minimizan los errores de reconocimiento y el de filtrado, además de pertenecer a un algoritmo de reconocimiento estructurado, con el cual se obtienen resultados muy buenos además de ser confiable.

En el programa que se muestra continuación se observa en el algoritmo que se obtiene un conjunto de distancias entre nuestra base de datos, la cual es un conjunto de vectores en una matriz, y nuestro vector a reconocer, el cual en forma matemática es un vector, pero en la realidad y en términos mas tangibles es nuestra señal de voz muestreada o como se diría en forma mas común es una señal digitalizada.

En la muestra de nuestro programa se tienen como entrada de comparación las 20 muestras que se ocuparon para el entrenamiento, en el programa ocupado en el reconocimiento en tiempo real se tuvieron más elementos a ser considerados, ya que como se implemento en tiempo real el procesamiento debe ser al momento de la captura. Esto es que en el momento de llevar acabo la captura hay tiempo muerto que debe de ser aprovechado por nuestro sistema, así que en la captura, análisis de inicio-fin de señal, autocorrelación, análisis LPC y todo el proceso de nuestro algoritmo de LPC-VQ es realizado mientras nuestra señal de voz es capturada, de esta manera cuando el proceso de captura es terminado por el algoritmo de inicio-fin de señal, el sistema a realizado ya algunos cálculos que son de importancia, además de que el tiempo es aprovechado pro medio de las interrupciones de Software y de hardware.

En Matlab los archivos de audio *.wav* son fácilmente leíbles como se muestra en la instrucción *wavread*. Para la facilidad y manejo de datos se ocupo la concatenación de cadenas, con la instrucción *strcat*, esto con el cometido de agregar una numeración a nuestra palabra (abrepuer1.wav, abrepuer2.wav...), que es como fueron nombradas nuestras palabras en el momento de su captura. Al final del proceso de calculo de métricas se las cuales son sumadas en cada segmento de comparación, que en el caso del ejemplo es de 8 y fue segmentada de forma lineal, se obtiene la mínima distancia a nuestro centroide, así que esta palabra fue la reconocida por el programa.

```
dmin=zeros(1,20);

strA = 'abrepuer';

for n=1:20
   abrePuer=wavread(strcat(strA, sprintf('%d', n),'.wav'));
   tam(n)=fix(length(abrePuer)/8);


   for i=1:8
      pala=abrePuer(1+((i-1)*tam(n)):i*tam(n)); %separamos la palabra en secciones de 8
      aux1=floor(length(pala)/128);

      for j=1:aux1
         AA(:,j)=pala(1+128*(j-1):128*j);       % División de la señal O en ventanas
         AR(:,j)=autocor(AA(:,j)',p+1)';        % Calculo de LPC y autocorrealación
         ALPC(:,j)=ALD(AR(:,j),p)';
      end

      for v=1:aux1
         for u=1:k
            distancia(u)=ditakura(ALPC(:,v),AR(:,v),centroides(i,:,u));    %la distancia itakura se
aplica de manera
         end
                                    %ditakura(LPC,Autocorrelacion,centroides)
         [Y,I] = min(distancia);                    %Verificar cual es la distancia mínima

         dmin(n)=Y + dmin(n);
      end

   end
   dmin(n) = dmin(n)/ aux1;
end

[Y,I] = min(dmin);                      %Verificar cual es la distancia mínima
hold on;
plot(dmin);
```

Tabla IV.4. Código en MATLAB de un reconocedor simple.

En el caso de el calculo de LPC-Cesptral se agrego una subrutina extra para robustecer el método. El código previamente mostrado no fue alterado solo fue anexado un extra para ser complementado el método de codificación de esta manera, después de tener la codificación LPC se procede al calculo de los LPC-Cepstral el cual es un metodo mas robusto de reconocimiento de voz también muy conocido en el campo del reconocimiento de voz. El programa siguiente implementado en Matlab ejemplifica el método de manera mas explicita.

Para la extracción de patrones de voz realizado en la implementación práctica, se considera el cálculo de los coeficientes LPC-C por ventana de señal. La idea es, que para cada segmento de voz, se extraigan los primeros 12 coeficientes LPC [14]. Luego, se desarrolla el cálculo cepstral para los coeficientes LPC anteriormente obtenidos. Así, la mezcla de estas dos técnicas parametriza de mejor manera la señal vocal, puesto que se consideran las características de la

excitación propiamente tal (LPC) y las del tracto vocal (CS) correspondientes con el modelo en cuestión.

```
%
%          Generador de LPC cepstral
%          Juan Manuel Ramirez Sosa
%
function [LPC] = LPCC(R,p)
alpha = LD(R',p);
%_____________________________________________________________
%              LPC cepstral parametros de conversion
%_____________________________________________________________
c(1)=alpha(1);
for m=2:p
   temp = 0;
   for k=1:m-1
      temp = temp + (k/m)*c(k)*alpha(m-k);
   end
   c(m)=alpha(m) + temp;
end


q=fix(3*p/2); %% Segun RABINER

for m=p+1:q
   c(m)=0;
   for k=m-p:m-1
     c(m)=c(m)+(k/m)*c(k)*alpha(m-k);
  end
end

LPC=c(:)';
```

Tabla IV.5. Código en MATLAB de la codificación LPC-Cepstral

4.3 Circuito de preamplificación

Nuestra tarjeta de desarrollo DSK-C6711 tiene como es de esperarse ya que su aplicación es de análisis y procesamiento de audio una entrada y una salida de audio, aprovechadas en esta aplicación de desarrollo, además de que cuenta con un convertidor analógico digital a la entrada de audio y un convertidor digital analógico a la salida. Los convertidotes tanto el de entrada como el de salida son externos a nuestro procesador esto con el propósito de ahorrar tiempo de procesamiento interno y programación. Estos dos circuitos son programables al inicio del programa y los datos son adquiridos por el puerto de comunicación serie por medio de una interrupción de software.

Como ya se a mencionado se conecto un micrófono para la captura de audio de nuestro circuito y como es sabido, la salida del micrófono no tiene mucha amplitud y además que nuestra tarjeta de desarrollo no tiene un sistema de preamplificación en su etapa de captura de datos se vio en la necesidad de diseñar un amplificador de audio de baja potencia.

El circuito integrado utilizado para este cometido fue el LM386, este es un circuito integrado es un amplificador para uso de bajo voltaje de consumo. La ganancia interna del amplificado es de 20,

pero con un capacitor y una resistencia externa entre los pines 1 y 8, la ganancia puede tener un incremento hasta una ganancia de 200. El circuito de la siguiente figura muestra la configuración de nuestro circuito, la configuración presentada muestra gráficamente lo mencionado, entre los pines 1 y 8 se presenta un capacitor de 10 μF, así que la entrada es aumentada en 200 máxima, esta ganancia también es controlada con el potenciómetro conectado en el pin 3.

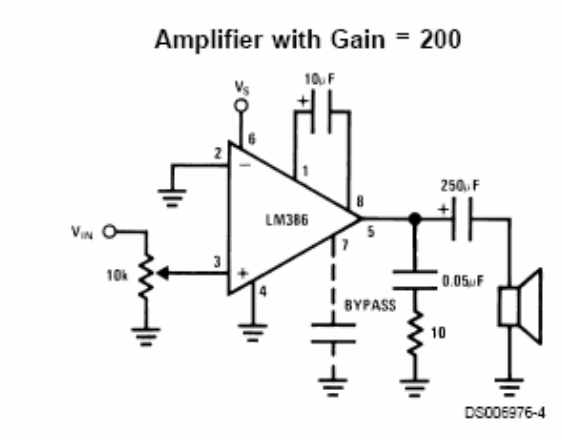

Figura IV.6. Circuito de preamplificación de audio

Chapa electrónica.

Para poder abrir la chapa de forma automática se implemento una chapa magnética o eléctrica, como las que se muestran en las ilustraciones siguientes. Este tipo de contrachapas son muy utilizadas en los sistemas de seguridad, en donde se restringe el acceso. Son muy comúnmente ocupadas en despachos, departamentos, casas, empresas y dependiendo el nivel de seguridad.

Deseado el tipo de chapa o contra chapa que se utiliza. Este tipo de sistemas de acceso de seguridad están catalogados

La contrachapa consta de un sistema magnético el cual por medio de una corriente eléctrica mueve un embolo y permite que la puerta pueda ser abierta. El circuito de esta es controlado en nuestro caso por medio de nuestro sistema de reconocimiento. Se permite el acceso dependiendo de la toma de decisión del circuito programado con el algoritmo previamente explicado.

Contrachapa para puertas de madera y de tambor. Alimentación de 12 VCA o 24 VCA.

Contrachapa cromada, ideal para puertas de madera, metálica, portones, con desactivación manual de emergencia, alimentación 12 VCA o 24 VCA.

Figura IV.7. Chapas electrónicas

La bobina que acciona la chapa se debe de alimentar con 12 o 24 VCA. La alimentación de 12 o 24 VCA no afecta mucho en el funcionamiento de apertura de la puesta solo en el consumo de energía de la contra chapa, que varia de 600 mA a 12 VCA hasta 1.2 A a 24 VCA. Asi que se opto por el uso de un transformador de 12 VCA. En este punto de debe de hacer notar que nuestro circuito general y de control, así como el de reconocimiento es de baja tensión. Es por esto que se diseño un circuito de acoplamiento de la salida de nuestra tarjeta de desarrollo y el circuito de apertura de la puerta.

Para proteger la tarjeta de desarrollo se implemento un acoplamiento óptico con el circuito integrado MOC311. este es un CI el cual por dentro están acoplados óptimamente un diodo y un transistor infrarrojos de manera que no tienen contacto físico, en el remoto caso de que la etapa de potencia falle y alguna corriente de alterna pueda dañar nuestro circuito de control

Esta interfaz se debió a algunas limitaciones en nuestra tarjeta de desarrollo, la salida de confirmación de reconocimiento o rechazo de palabra se llevo a cabo por la salida de audio dando un máximo de voltaje de salida en los casos de reconocer abre, abre puerta y un bajo de voltaje de salida en los casos de cierra, cierra puerta, palabra desconocida o repetir la palabra, esta ultima opción se da cuando en el caso de reconocer la palabra se rebasar un umbral de cercanía con la palabra más próxima, se considera que el reconocimiento de la palabra es dudoso. La confirmación es una manera de asegurar el reconocimiento y garantizar que la palabra reconocida es en realidad la palabra pronunciada.

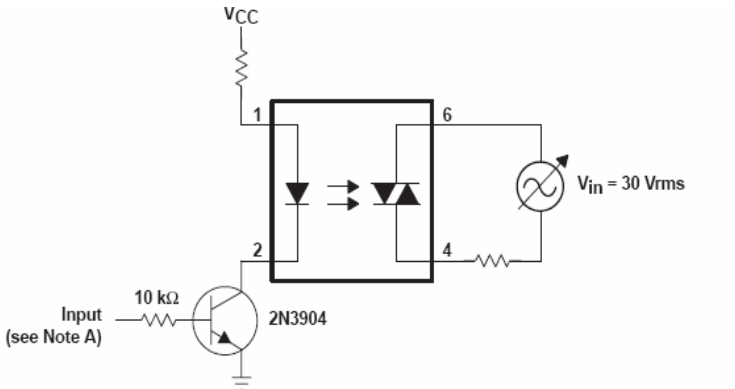

Figura IV.8. Salida para la interfaz de conexión entre la tarjeta TMS320C6711 y el sistema de potencia de acción de la chapa.

El sistema de potencia de acción de nuestra chapa se muestra en la figura el cual fue colocado en la salida de salida de nuestra tarjeta. El transistor es un sistema de seguridad para la protección del circuito de salida de audio, ya que éste no alimenta suficiente corriente, y la corriente demandada por nuestro circuito MOC311 la cual es utilizada para encender el diodo infrarrojo interno del circuito integrado puede dañar la salida de audio.

Resultados y Conclusiones

5. Resultados y Conclusiones

Uno de los puntos mas importantes de este trabajo se investigación, fue la comparación del método LPC y el método LPC-Cepstral. Los resultados obtenidos fueron muy satisfactorios, además de que se cumplieron los objetivos propuestos de una manera exitosa. A continuación se presentaran los datos que se obtuvieron, estadísticas, así como una interpretación de estas.

5.1 Resultados LPC

Como ya se ha explicado anteriormente, este es uno de los métodos mas utilizados en el reconocimiento de palabras, siendo uno de los métodos más simples en la rama del procesamiento de voz, pero no por esa razón pierda su importancia. Este método de procesamiento de voz, en este caso reconocimiento, se complementa con otros métodos como es el VQ.

En el caso del desarrollo del proyecto de reconocimiento en tiempo real, el cual fue desarrollado en la tarjeta de desarrollo DSK 6711, se tuvo que implementar un sistema de umbrales de reconocimiento. Esto con el propósito de garantizar de una manera más confiable el funcionamiento del reconocedor y de esta manera descartar las palabras desconocidas o evitar un reconocimiento erróneo de las palabras.

En el proceso de desarrollo se siguieron algunos pasos para que el funcionamiento del sistema fuera optimo. En el proceso de implementación se llevaron acabo varios procesos de diseño, la parte de entrenamiento, a esto me refiero al calculo de centroides, como ya se ha mencionado anteriormente se llevo acabo en una computadora con la ayuda de MATLAB, para garantizar el funcionamiento del reconocedor en este método se realizó un reconocedor en el mismo lenguaje en el que fue realizado el de entrenamiento. El propósito de esta parte, fue realizado con el cometido de garantizar y ver el comportamiento de los datos que fueron obtenidos.

5.1.1 Reconocedor LPC Matlab

Los resultados del comportamiento de nuestro algoritmo de reconocimiento en Matlab, fue satisfactoriamente probado, garantizando un buen comportamiento en nuestro sistema de desarrollo en tiempo real. En este punto se debe enfatizar que el comportamiento de nuestro programa, aunque es el mismo algoritmo tanto en tiempo real como en el que fue programado en Matlab, su conducta es similar más no igual. Esto se debe aun conjunto de factores los cuales afectan el resultado. Uno de estos factores es que en el desarrollo y pruebas en computadora nuestras señales de voz son archivos de audio previamente grabadas, además de que algunos de estos archivos fueron implementados para el entrenamiento, de manera que, el reconocimiento aumenta en esta prueba.

Las pruebas se realizaron con el diccionario utilizado en nuestro sistema. Se realizó el reconocimiento entre las todas las palabras, se comparo *abre* vs *abre, abre puerta, cierra y cierra puerta* de manera respectiva, y de manera homologa *abre puerta, cierra y cierra puerta,* fueron compradas con el mismo vocabulario que se implemento para el reconocimiento.

En la realización de este proyecto, uno de los objetivos a alcanzar y siendo este el principal en esta tesis fue hacer una comparación entre dos de los métodos mas ocupados en tratamiento digital de la voz. Esta comparación es con el propósito de demostrar que el método LPC comparado con el LPC-Cepstral. Los resultados que se presentan, como se mostrara posteriormente, que el comportamiento del método Cepstral es mas estable que LPC, además que nos permite hacer una diferenciación entre el hablante que pronuncia el comando.

En primera instancia de el entrenamiento se calcularon el conjunto de centroides, de esta manera en Matlab se ejecuto un programa previamente realizado, el cual de manera análoga que en la tarjeta de desarrollo de Texas Instrument. Pero como se ha mencionado de esta manera se ve el

comportamiento posible que podría tener nuestro sistema. Aunque esto no garantiza su igual comportamiento. A continuación se muestran gráficas obtenidas de reconocimiento.

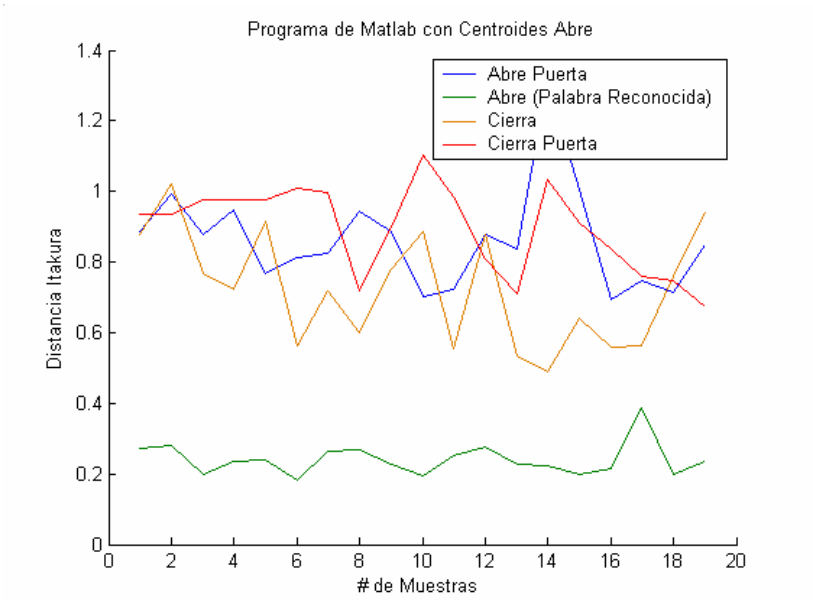

Figura V.1. Gráfica de reconocimiento en Matlab de la palabra *abre*

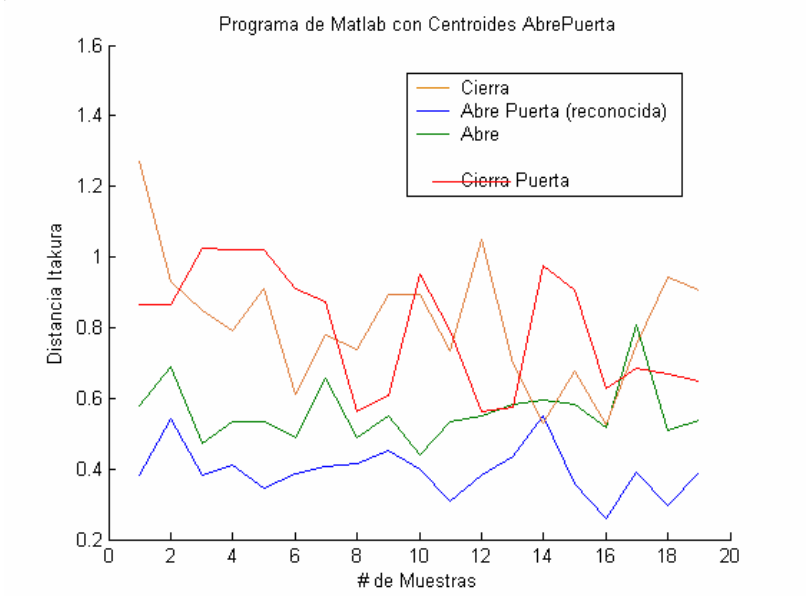

Figura V.2. Gráfica de reconocimiento en Matlab de la palabra *abre puerta*

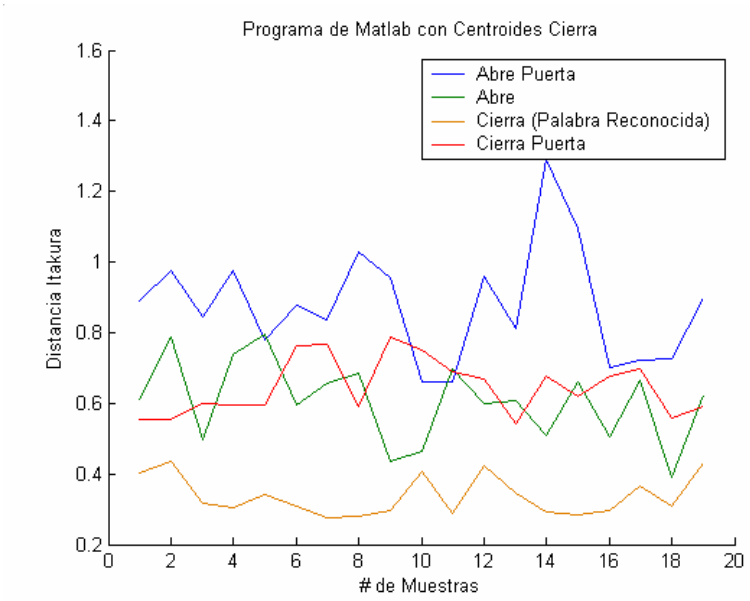

Figura V.3. Gráfica de reconocimiento en Matlab de la palabra *cierra*

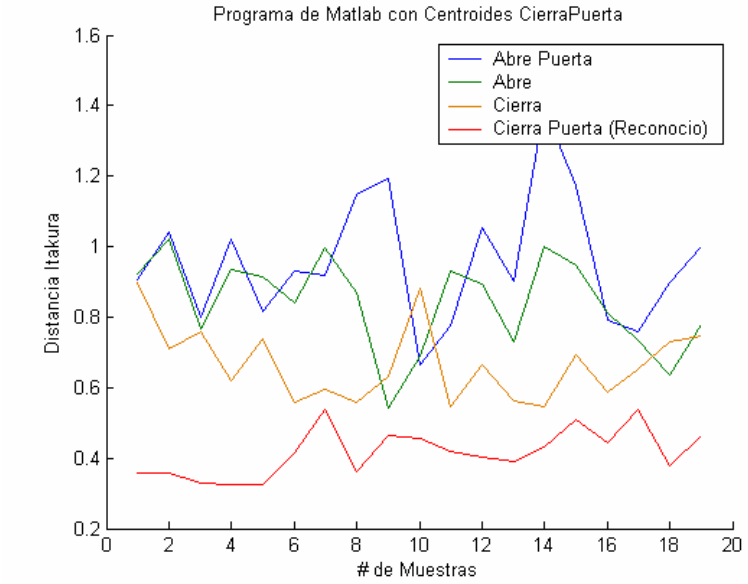

Figura V.4. Gráfica de reconocimiento en Matlab de la palabra *cierra puerta*

En las figuras anteriores se puede observar el comportamiento del reconocimiento, en cada una de estas, se aprecia que la palabra reconocida se encuentra en la parte baja, por lo que se puede decir que el reconocimiento fue correcto. Observando estas mismas gráficas de reconocimiento las palabras no reconocidas se encuentran en la parte superior ya que la distancia de reconocimiento aumenta.

Como ya se ha explicado anteriormente el reconocimiento llevado acabo por medio del algoritmo de cuantización vectorial, cuenta con un método de cálculo de distancias, de esta manera la distancia menor es la reconocida. Al momento de llevar acabo el reconocimiento ya en tiempo real se calcula la distancia de la palabra a reconocer y nuestros centorides, que han sido previamente calculador por el método de K-medias y que están almacenados en nuestra tarjeta de desarrollo, son comparados con la señal de entrada de audio que es capturada en el sistema, es almacenada

y codificada con el algoritmo de LPC, el cual fue el primer método implementado, para posteriormente comparar los resultados con un método denominado LPC-Cepstral.

LPC en comparación a LPC-Ceptral no tiene la característica de discriminación de hablante, esto se debe a que la información arrojada en el entrenamiento en insuficiente para este propósito. LPC-Ceptral en cambio nos da información extra, además que ha demostrado ser un método que arroja más información, siendo por este motivo un método más robusto y confiable.

5.1.2 Implementación del reconocedor LPC sobre la tarjeta de desarrollo

Como paso consecutivo, se probó el algoritmo de VQ en tiempo real sobre la tarjeta de desarrollo DSK6711, los resultados obtenidos en este proceso cambiaron con respecto al programa realizado en el sistema Matlab. El reconocimiento obtenido se presenta en la tabla V.1. Como se puede observar en esta matriz de confusión el reconocimiento obtenido fue muy bueno, si se aprecia esta tabla, se puede decir que solo hubo un error, este se le puede atribuir a al confusión de la palabra *cierra puerta* con la palabra *cierra*.

En el programa de reconocimiento en tiempo real se desarrollo un sistema de umbrales, esto se debió a una mejora en el reconocimiento. En el momento de bajar nuestros centroides a la tarjeta de desarrollo, se hizo una serie de repeticiones con el propósito de calcular un promedio de la distancia menor, que en este caso seria la palabra reconocida correctamente y la distancia más próxima. El cometido de guardar la distancia mas próxima fue con el propósito de poder obtener un promedio de esta distancia con respecto a la palabra reconocida, de esta forma se calculo un umbral de error. De manera que, aunque hubiera una distancia menor a todas se debe de calcular otra distancia entre la distancia mas próxima y esta debe de estar entre un rango obtenido con el promedio de las distancias obtenidas en un tipo entrenamiento para obtener estos umbrales. Por lo anterior si la palabra reconocida excede de un rango promedio de distancias se imprime en el Log *palabra desconocida* y de manera análoga si la distancia de la palabra reconocida entra en este promedio y sin embargo la distancia más próxima es aceptada menor al promedio de distancias próximas se imprime en el Log *repetir palabra*. Para que una palabra se a reconocida la menor distancia y la distancia mas próxima deben de estar en el rango promedio previamente calculado.

Con lo anterior se garantiza un mejor funcionamiento y control del sistema, ya que la confusión del reconocimiento queda restringido a dos parámetros más en el programa y se evita que aunque la palabra sea reconocida debe de cumplir con un parámetro estadístico, que en este caso son dos promedios experimentales que son: promedio de palabra reconocida y promedio de la distancia de la palabra más próxima.

	Palabra Correcta	Palabra Desconocida	Repetir Palabra	Confusión de Palabra
Abre	19	0	1	0
Abre Puerta	17	1	2	0
Cierra	17	3	0	0
Cierra Puerta	17	0	2	1(Cierra)

Tabla V.1. Tabla de confusión para LPC

En las figuras que se muestran a continuación se muestra ya de manera gráfica el resultado del reconocimiento, en porcentajes, del reconocimiento del método de codificación LPC. Los resultados obtenidos en este proceso de la investigación, fueron exitosos. Ya que el reconocimiento mas bajo fue de un 80% y el más alto se presento en un 95%, como se observa tanto en la tabla de confusión como en su respectiva representación gráfica.

Estadísticas de reconocimiento para el método de codificación LPC

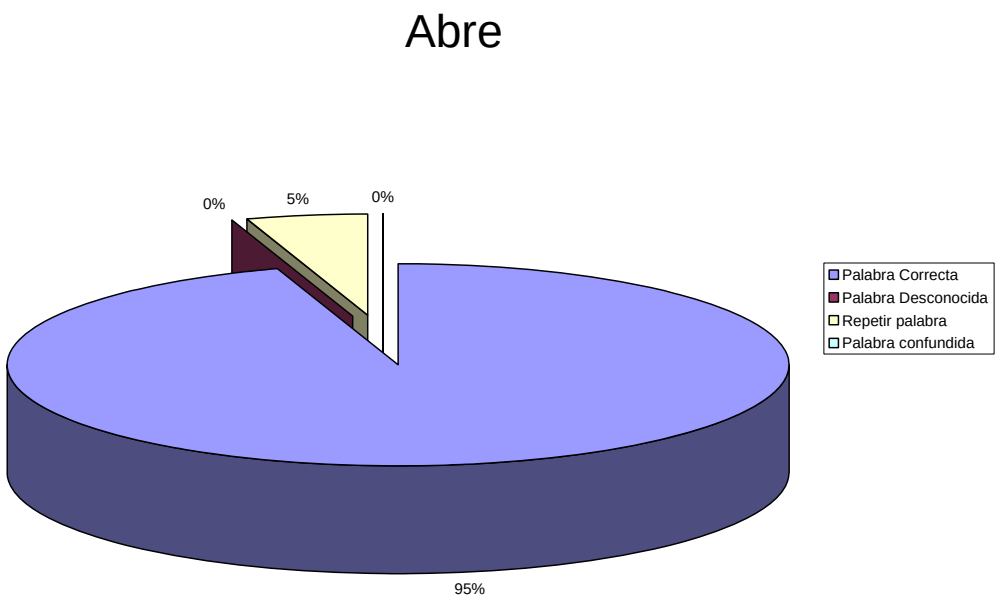

Figura V.5. Gráfica de estadísticas de reconocimiento para la palabra *abre* obtenidas del método LPC

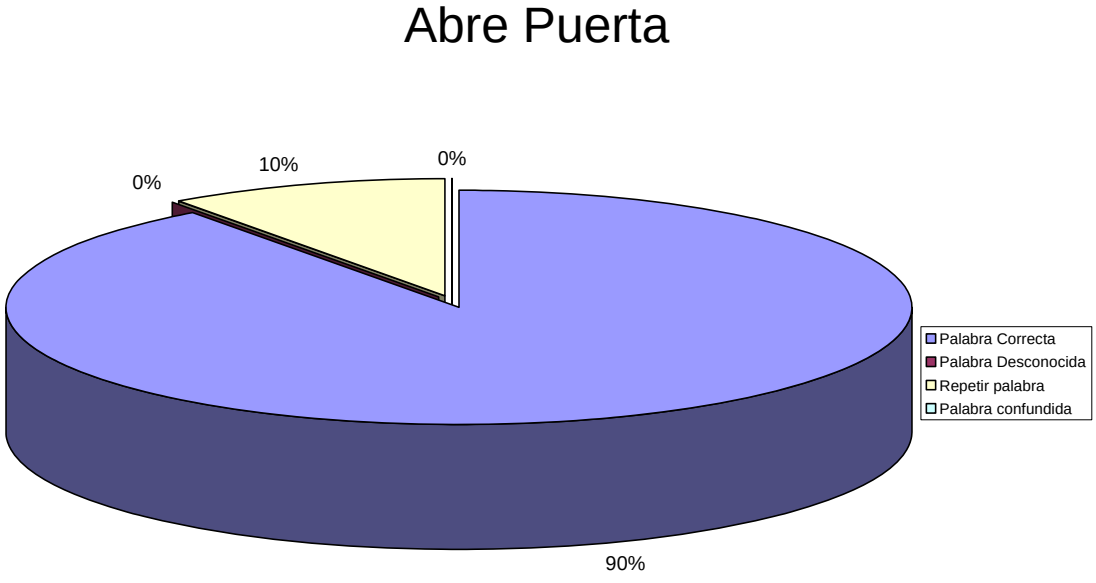

Figura V.6. Gráfica de estadísticas de reconocimiento para la palabra *abre puerta* obtenidas del método LPC

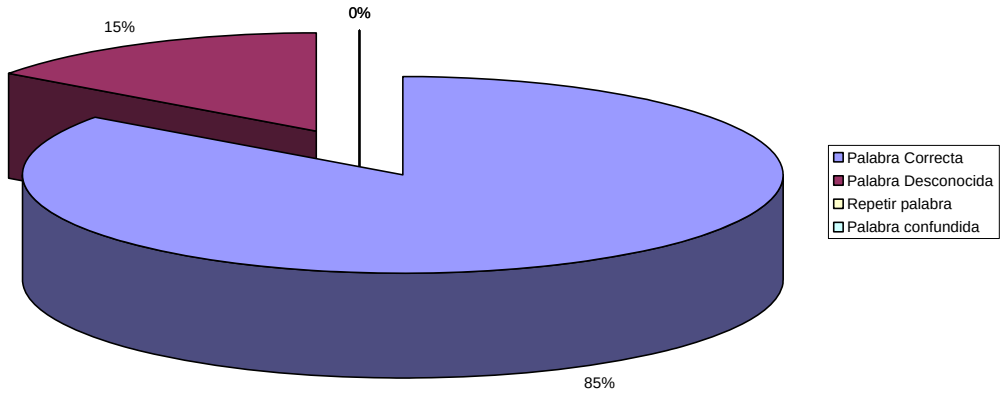

Figura V.7. Gráfica de estadísticas de reconocimiento para la palabra *cierra* obtenidas del método LPC

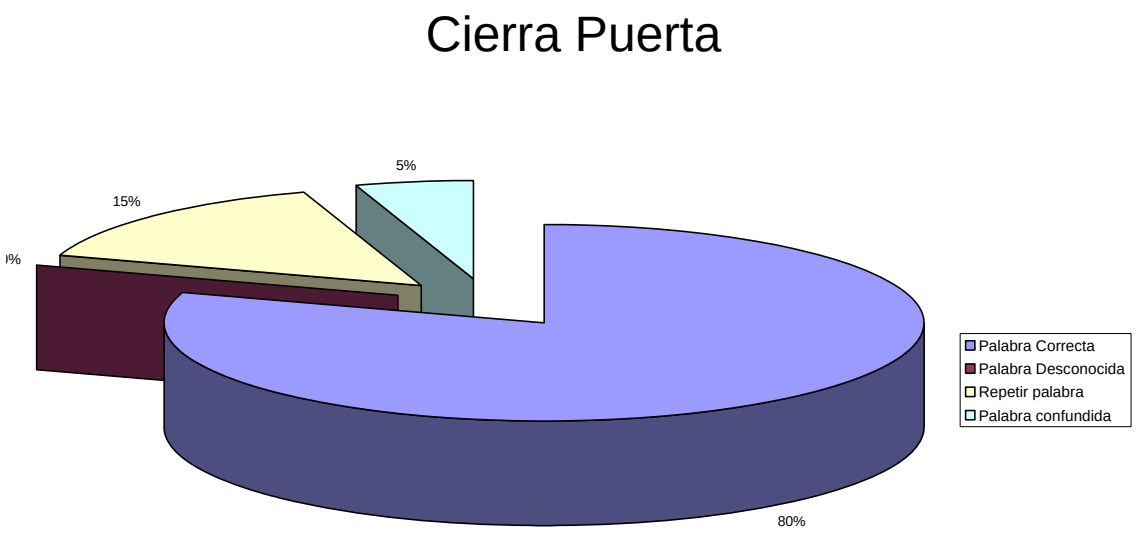

Figura V.8. Gráfica de estadísticas de reconocimiento para la palabra *cierra puerta* obtenidas del método LPC

Num	*Abre*		Abre Puerta		*Cierra Puerta*		*Cierra*	
	Min dist	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*
1	0,5686274	0,6703461	0,6189649	0,6560177	0,631013	0,650912	0,607923	0,717868
2	0,5995308	0,6643184	0,626098	0,6911318	0,686175	0,708026	0,646949	0,897889
3	0,6048662	0,6686248	0,5708082	0,6295714	0,645636	0,694147	0,698882	0,873518
4	0,5873153	0,7007861	0,5803786	0,624073	0,627856	0,666071	0,677066	0,92934
5	0,6125982	0,6823372	0,5822641	0,6440491	0,611198	0,677524	0,687745	0,782162
6	0,6009745	0,7059222	0,5827141	0,6202415	0,618477	0,651854	0,655763	0,82942
7	0,5831336	0,6808682	0,6016037	0,6637017	0,617299	0,674369	0,610056	0,82381
8	0,5688581	0,6461403	0,6051533	0,6153334	0,618413	0,656978	0,691814	0,852499
9	0,5979177	0,6666514	0,5690427	0,6155378	0,612291	0,63937	0,609548	0,740995
10	0,5547066	0,6340821	0,6166631	0,6547149	0,610792	0,666177	0,563407	0,763541
11	0,6168191	0,6770827	0,6431735	0,6840162	0,596193	0,634602	0,672821	0,843168
12	0,5591749	0,6068005	0,5716669	0,6339983	0,649539	0,674412	0,562456	0,762104
13	0,5328606	0,6263942	0,6005392	0,6149229	0,661109	0,692694	0,531173	0,634292
14	0,5408229	0,6248103	0,6004265	0,6267453	0,65004	0,716213	0,607678	0,767354
15	0,6132259	0,6857882	0,5472846	0,5939308	0,643545	0,693091	0,583566	0,767157
16	0,5409141	0,6670928	0,6061357	0,6687972	0,669545	0,727847	0,661411	0,754288
17	0,6074215	0,6884781	0,5947198	0,6773797	0,649012	0,696273	0,670762	0,839648
18	0,5474949	0,6268079	0,6017293	0,6517798	0,602044	0,64824	0,58812	0,698415
19	0,5484366	0,6144641	0,6088808	0,6506915	0,631942	0,691986	0,592646	0,833328
20	0,5913119	0,6801034	0,6039097	0,6570496	0,672593	0,719106	0,549224	0,717371

Tabla V.2. Tabla de distancia mínima y próxima para el algoritmo LPC

	Abre		Abre Puerta		*Cierra Puerta*		*Cierra*	
	Min dist	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*
Promedio	0,578850	0,660894	0,596607	0,643684	0,635235	0,678994	0,623450	0,791408
Desviación estándar	0,027970	0,029374	0,029374	0,026182	0,025051	0,027318	0,051852	0,072954

Tabla V.3. Tabla de estadísticas para LPC de las distancias mínima y próxima

En las tablas anteriores se muestran las estadísticas que se realizaron para el cálculo de umbrales. Como se indicó se realizaron 20 repeticiones, en las cuales se obtuvo un promedio y una desviación estándar. Estos datos se grabaron en el programa que se desarrollo para el sistema de reconocimiento en tiempo real.

Teniendo ya nuestros centroides y el programa de reconocimiento en nuestra tarjeta se procedió a hacer estadísticas de reconocimiento, en las figuras V.7-V.8 se presentan los resultados de manera gráfica, obtenidos de manera experimentar en el programa realizado en el sistema de desarrollo Code Composer Studio. Este programa contiene los elementos presentados en capítulos anteriores, con la variante de interrupciones de hardware y software, los cuales fueron implementados en la tarjeta de desarrollo con el objetivo de hacer el sistema de reconocimiento optimo y mejorar el tiempo de reconocimiento con estas interrupciones.

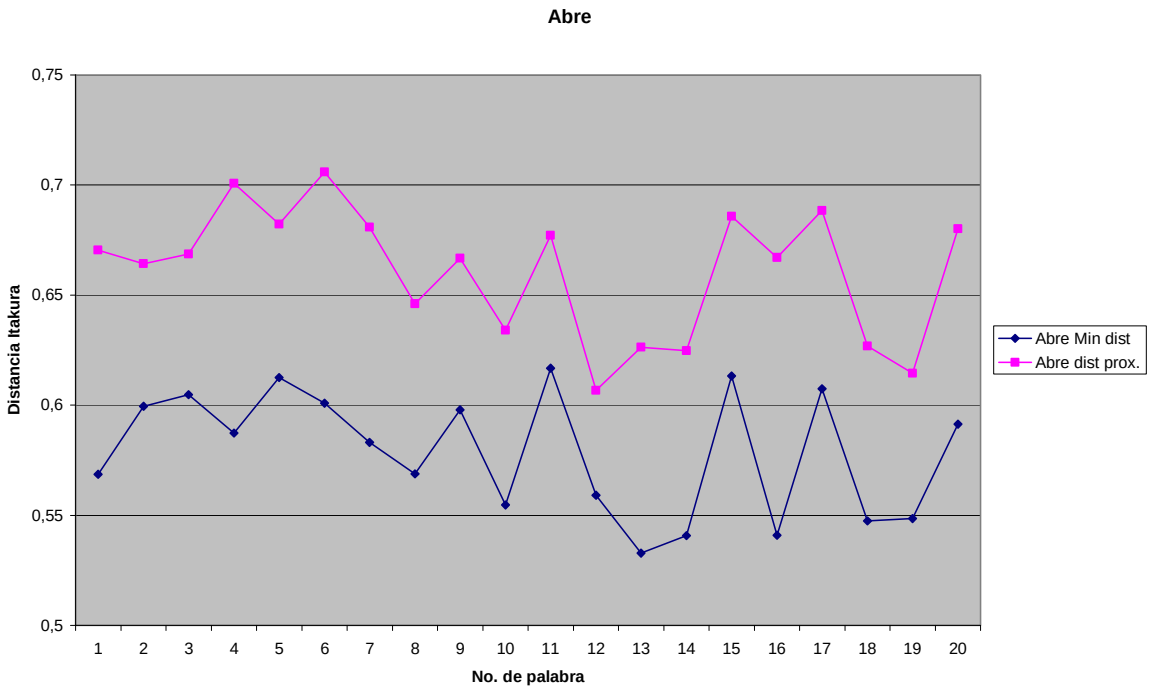

Figura V.9. Representación gráfica para la palabra Abre y su distancia próxima

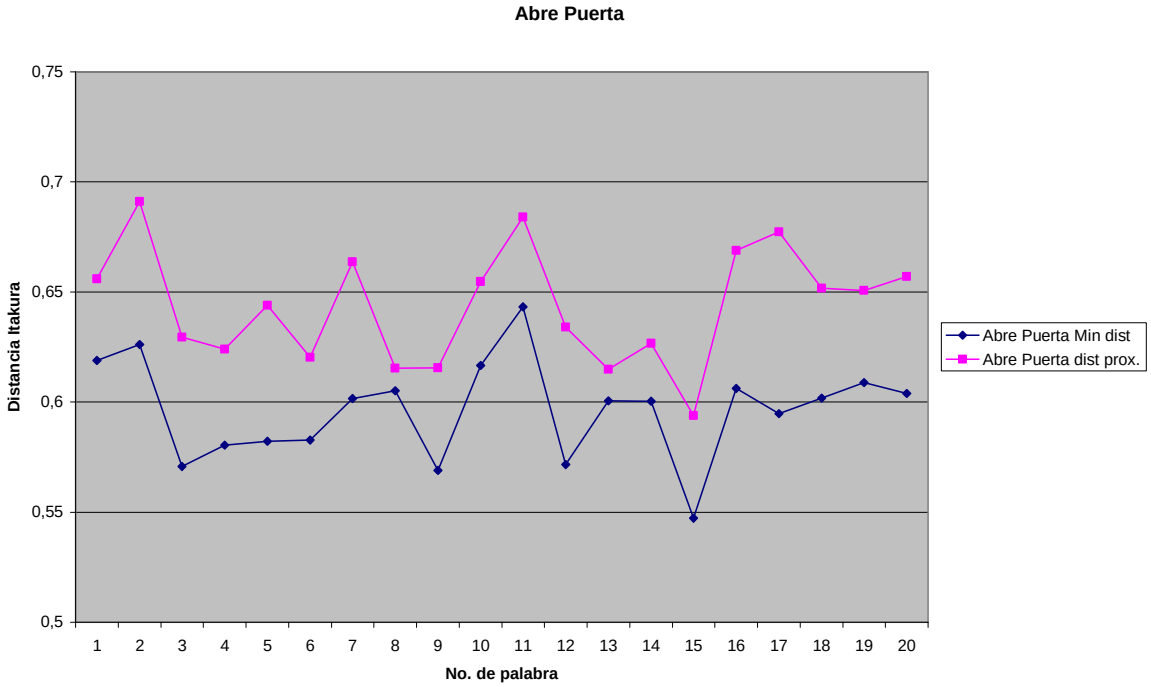

Figura V.10. Representación gráfica para la palabra Abre Puerta y su distancia próxima

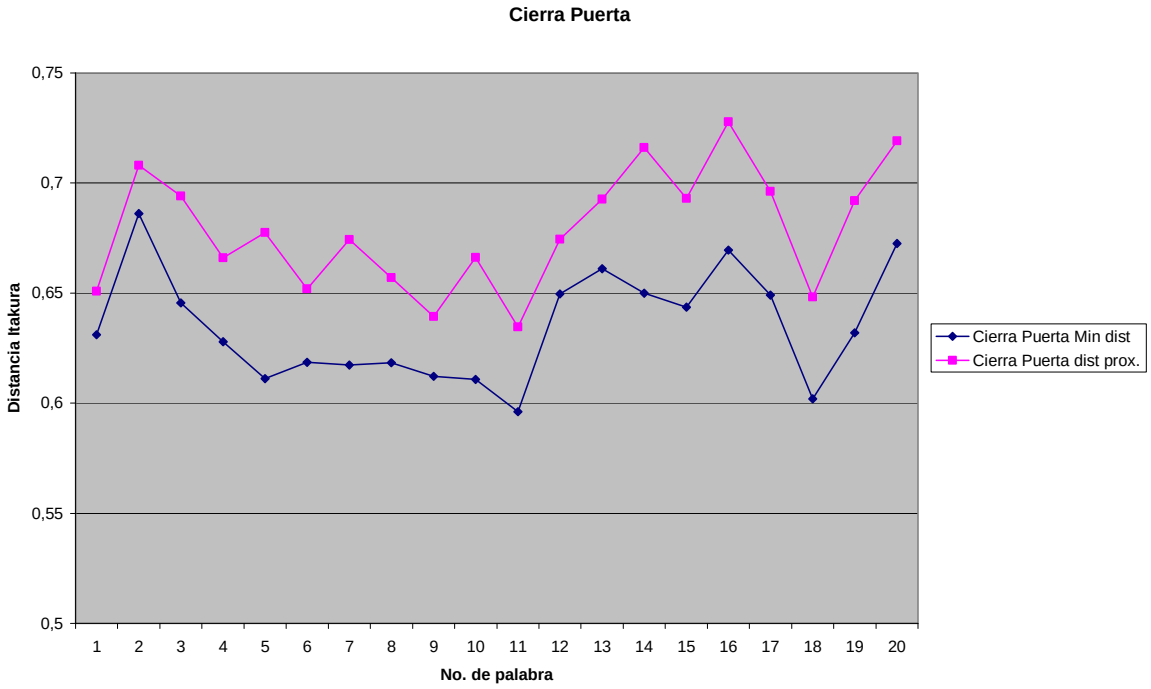

Figura V.11. Representación gráfica para la palabra Cierra PUerta y su distancia próxima

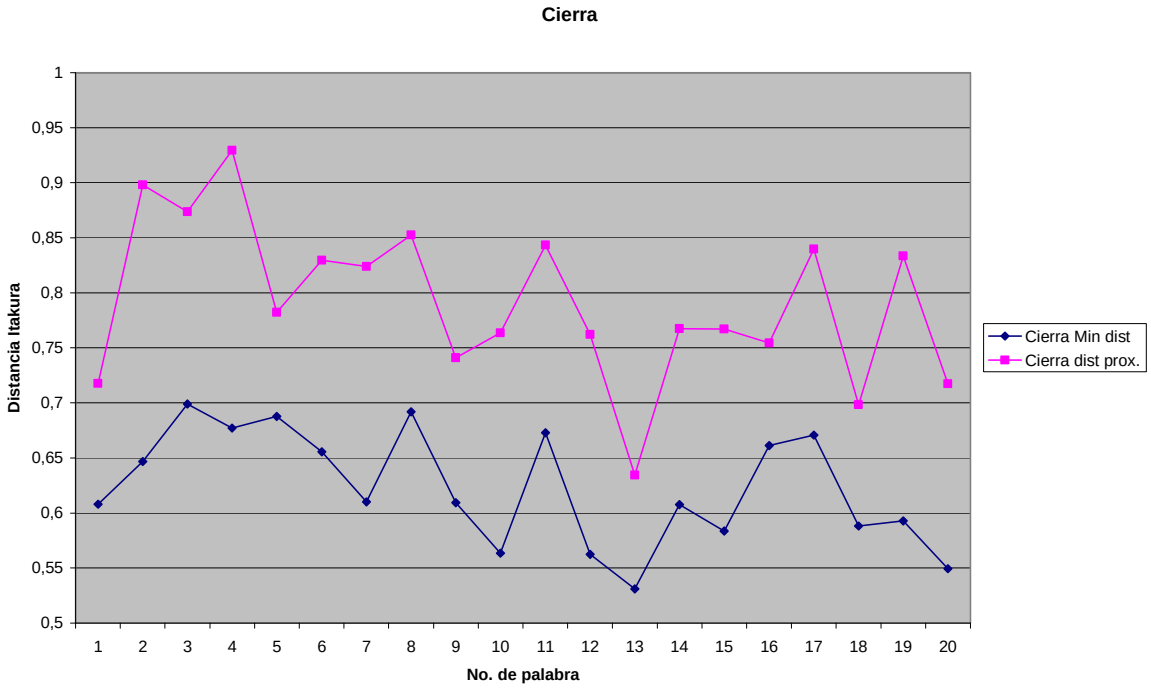

Figura V.12. Representación gráfica para la palabra Cierra y su distancia próxima

En las figuras de la V.9 a la V.12 se muestran las gráficas de reconocimiento, extraídas de los experimentos realizados ya en tiempo real sobre la tarjeta de desarrollo. En estos resultados que se muestran ya gráficamente en las imágenes, se puede apreciar que existe una distancia entre la palabra reconocida y la distancia de la palabra más próxima, como es de esperarse, la distancia más próxima en las palabras reconocidas fueron:

- La distancia más próxima a la palabra ***cierra puerta*** fue ***cierra***.
- La distancia más próxima a la palabra ***abre*** fue ***abre puerta***.
- La distancia más próxima a la palabra ***cierra puerta*** fue ***cierra***.
- La distancia más próxima a la palabra ***cierra*** en este punto se debe de mencionar que en esta palabra hubo distancias muy variadas. Una de las distancias mínimas mas comunes presentadas fue ***cierra*** puerta pero en ocasiones se presentaron aunque en menos proporción que la primera la primera las ***palabras abre*** y ***abre puerta.***

5.2 Reconocedor LPC-Ceptral

Una de las características a comprobar en el algoritmo de reconocimiento Cepstral, fue la discriminación que hace sobre el hablante o locutor. A esto me refiero, que a diferencia del algoritmo de LPC, no diferencia entre un hablante y otro. De manera que programando este algoritmo se intento probar esa teoría, obteniendo muy buenos resultados. A continuación se explicará de manera más explícita, la metodología que se realizo así como los resultados obtenidos. En esta parte se compararon los dos métodos, siendo Cepstral mayor robusto que LPC, los resultados presentados a continuación fueron analizados minuciosamente y fueron comparados con los primeros para obtener las diferencias esenciales entre ambos métodos.

El reconocimiento con respecto al primer método presentó una mejora en sus estadísticas. En esta parte se debe de mencionar que se presento una estabilidad en el reconocimiento en palabras largas, en este caso son las palabras *cierra puerta* y *abre puerta*, ya que el reconocimiento aumentó en estas palabras de nuestro diccionario.

	Palabra Correcta	Palabra Desconocida	Repetir Palabra	Confusión de Palabra
Abre	21	2	2	0
Abre Puerta	24	1	0	0
Cierra	21	3	2	0
Cierra Puerta	23	0	1	1(Cierra)

Tabla V.4. Tabla de confusión para LPC-Cepstral

En la tabla V.4 se observa la tabla la tabla de confusión. Aquí se observa en comparación con la primera tabla solo hubo una confusión de palabra lo cual indica que nuestro sistema es confiable ya que la confusión de esta palabra fue entre las palabras *cierra* y *cierra puerta*. Esto debió a la similaridad que presentan las dos palabras. Posteriormente en la tabla de distancias mínima y próxima se observará esto.

5.2.1 Reconocedor LPC-Ceptral Matlab

De manera análoga al algoritmo anterior, este se programó antes que llevarlo acabo en tiempo real, en Matlab. Este algoritmo se presentó en capítulos anteriores, por lo cual se puede consultar su código y tanto el primer algoritmo de reconocimiento como el segundo pueden ser comparados por medio de su código fuente. Así como hacer mejoras y optimizaciones al código.

5.2.2 Implementación del Reconocedor LPC-Cepstral sobre la tarjeta de desarrollo

El reconocimiento realizado en el programa que se llevó a cabo en Matlab se muestra en las tablas V.4 y V.5. En esta parte cabe mencionar que el umbral entre la distancia mínima o reconocida y al

distancia próxima. Esto indica una estabilidad en el algoritmo ya que la distancia entre la palabra reconocida y la próxima aumenta, evitando la confusión de la palabra.

Como se mostrara en la siguiente tabla de resultados, de distancia mínima y distancia próxima, del algoritmo LPC-Cepstral, se puede apreciar que las distancias para el cálculo de umbrales creció de una forma considerable, tomando en cuenta las distancias obtenidas en el procedimiento anterior (LPC).

Num	***Abre***		**Abre Puerta**		***Cierra Puerta***		***Cierra***	
	Min dist	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*
1	1,001985	1,003826	1,002561	1,003916	1,002172	1,003781	1,002377	1,003509
2	1,002011	1,002777	1,002245	1,003403	1,002108	1,004814	1,00275	1,004318
3	1,001537	1,00268	1,003025	1,004725	1,003017	1,004869	1,002522	1,00309
4	1,001381	1,002167	1,002447	1,00328	1,002622	1,005	1,003186	1,003756
5	1,00126	1,002211	1,002818	1,004226	1,002198	1,004256	1,003021	1,003671
6	1,002026	1,002868	1,002609	1,003224	1,002013	1,004528	1,002579	1,003702
7	1,001377	1,002045	1,002129	1,003461	1,00241	1,004542	1,002771	1,003314
8	1,001906	1,002689	1,002098	1,003543	1,002345	1,004396	1,002965	1,004273
9	1,001245	1,00253	1,002631	1,004106	1,002811	1,005733	1,002509	1,00329
10	1,001646	1,002275	1,00204	1,004156	1,003284	1,004786	1,003115	1,003606
11	1,001602	1,002925	1,002963	1,004431	1,003314	1,005814	1,002259	1,003241
12	1,001616	1,002272	1,00219	1,003702	1,003116	1,005314	1,00296	1,003696
13	1,001856	1,003195	1,002932	1,00369	1,003164	1,004355	1,002562	1,003875
14	1,001837	1,002269	1,001505	1,003444	1,002687	1,004952	1,002894	1,0039
15	1,001999	1,002202	1,002313	1,004215	1,002228	1,004712	1,002436	1,002964
16	1,00203	1,003172	1,002369	1,003126	1,002629	1,00458	1,002312	1,003336
17	1,001867	1,003699	1,002466	1,004959	1,002003	1,004438	1,002261	1,003257
18	1,001974	1,003387	1,001847	1,00342	1,002459	1,004186	1,002692	1,003354
19	1,002026	1,002549	1,002268	1,00315	1,003063	1,00534	1,002391	1,003974
20	1,001293	1,003178	1,002524	1,00358	1,002849	1,004262	1,002878	1,003833

Tabla 4. Tabla de distancia mínima y próxima para el algoritmo LPC-Cepstral

	Abre		**Abre Puerta**		***Cierra Puerta***		***Cierra***	
	Min dist	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*	*Min dist*	*dist prox.*
Promedio	1,00219	1,00362	1,002423	1,003788	1,002672	1,003598	1,002625	1,004733
Desviación estándar	0,00066	0,000858	0,000425	0,00053	0,000292	0,00037	0,000435	0,000519

Tabla V.5. Tabla de estadísticas para LPC- Cepstral de las distancias mínima y próxima

Observando detenidamente aunque las distancias entre palabras son menores que el procedimiento de programación anterior, éstas permanecen más estables, además de que la distancia entre las 2 palabras es más constante. En las gráficas consecuentes, se muestra de manera más explícita lo dicho en esta parte.

En el proceso de estadísticas del método Cepstral, se llegó a la conclusión, tomando en cuenta los resultados obtenidos, que es un algoritmo de reconocimiento más confiable y robusto. En el caso del reconocimiento del *hablante* el proceso a seguir fue el siguiente:

- El entrenamiento se llevo acabo con un solo *hablante*.
- En el programa realizado en Matlab se realizo un reconocedor para verificar el funcionamiento de los centroides calculados.
- Se realizaron pruebas en la computadora del funcionamiento obteniendo muy buenos resultados del proceso.

- En la tarjeta de desarrollo se llevaron pruebas con el algoritmo Central de reconocimiento con el locutor entrenado.
- Se calcularon los umbrales de reconocimiento para el sistema, ya implementado en la tarjeta de desarrollo. En las imágenes 13, 14, 15 y 16 se pueden apreciar las gráficas obtenidas.
- Ya obtenidos los umbrales se procedió a implementarlos con el objeto de mejorar el reconocimiento.

Para llevar un estudio más detallado y obtener resultados y estadísticas más confiables, se optó por llevar a cabo pruebas de reconocimiento en el programa de Matlab, por lo cual se procedió a llevar acabo la captura de palabras de otros *hablantes*. Estos archivos de audio no fueron entrenados como el *hablante* principal.

Se grabaron un conjunto de palabras y se ocuparon 3 *hablantes* diferentes. Esto con el propósito de probar la discriminación que tenía o la dependencia que nuestro sistema presenta a los diferentes hablantes que no han sido entrenados previamente. Con esto se demostraría que el algoritmo de LPC-Cepstral es un método confiable para el reconocimiento de *hablante*.

De los 3 hablantes se obtuvieron 4 conjuntos de archivos, los archivos fueron procesados en la computadora por algoritmo de reconocimiento LPC-Cepstral. Los hablantes que se escogieron para la prueba, fueron escogidos de manera que todos fueran del mismo género y la edad en que se encontrarán fuera muy cercana a la del hablante entrenado. Esto se realizo con el propósito de reproducir de una manera similar las condiciones del hablante, aunque sabemos que entre uno y otro existen variaciones considerables, que el método podría tomar en cuenta para llevar acabo el correcto reconocimiento o no de la palabra. Ya que los hablantes no fueron entrenados el sistema, como se debe de pensar, no debe de reconocer los comandos emitidos por el individuo, pero esto no ocurre en el caso del método LPC, ya que como se sabe es independiente del locutor que emite la señal.

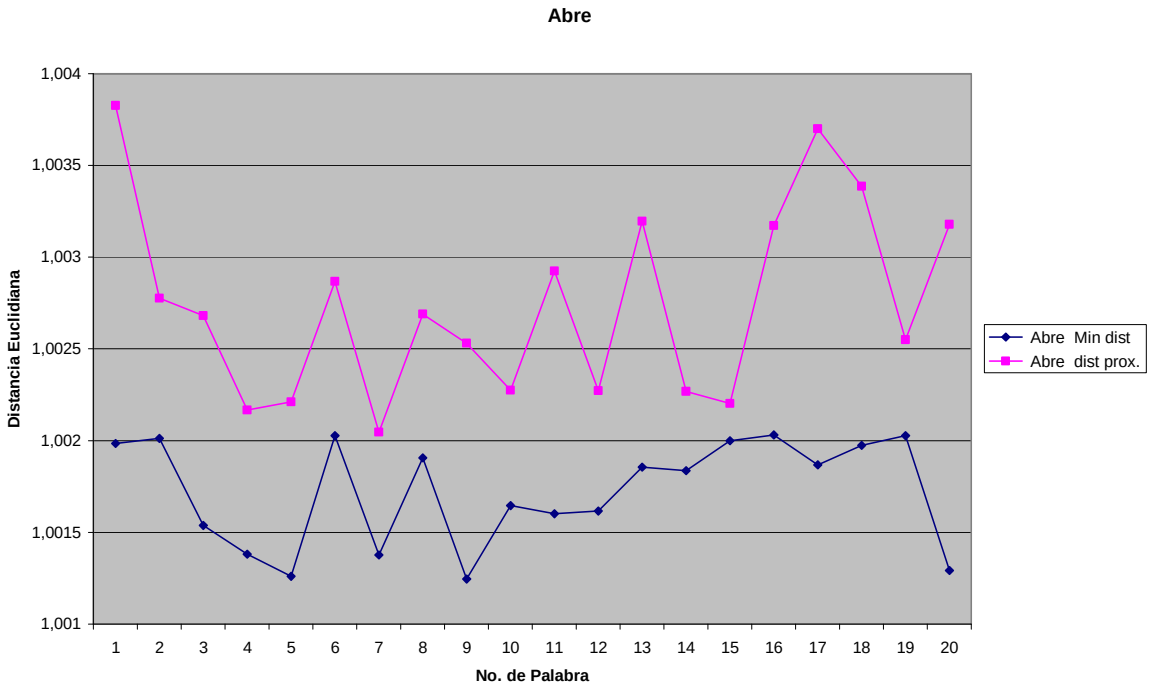

Figura V.13. Representación gráfica para la palabra Abre y su distancia próxima para el método de reconocimiento LPC-Cepstral

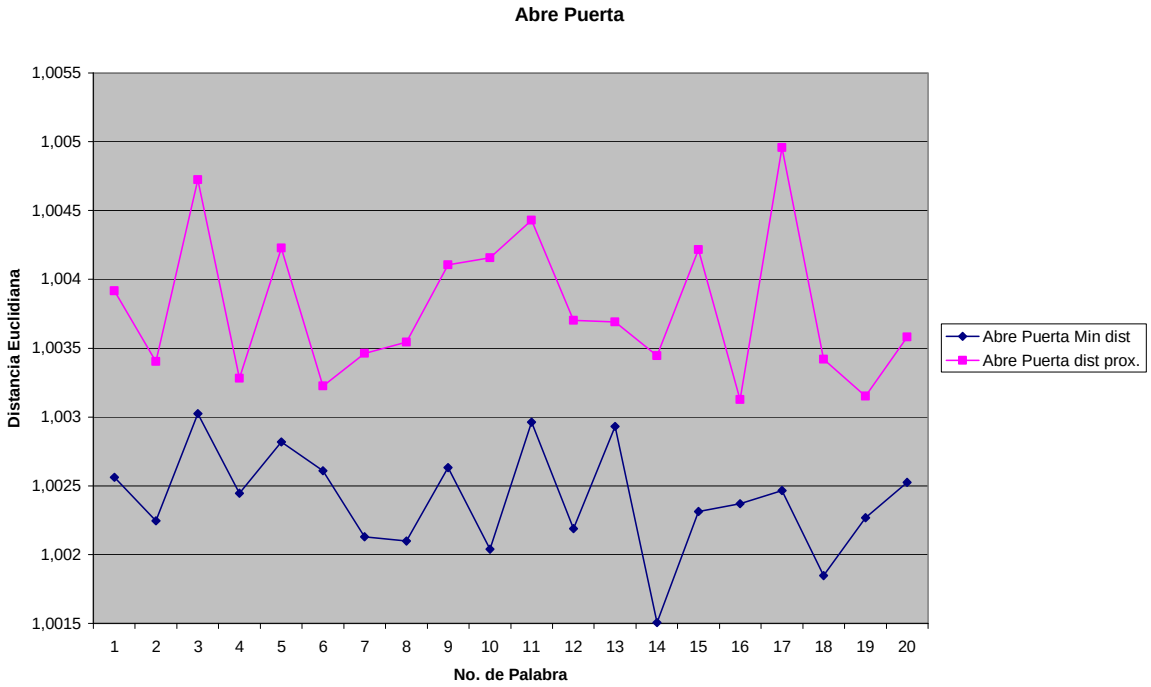

Figura V.14. Representación gráfica para la palabra Abre Puerta y su distancia próxima para el método de reconocimiento LPC-Cepstral

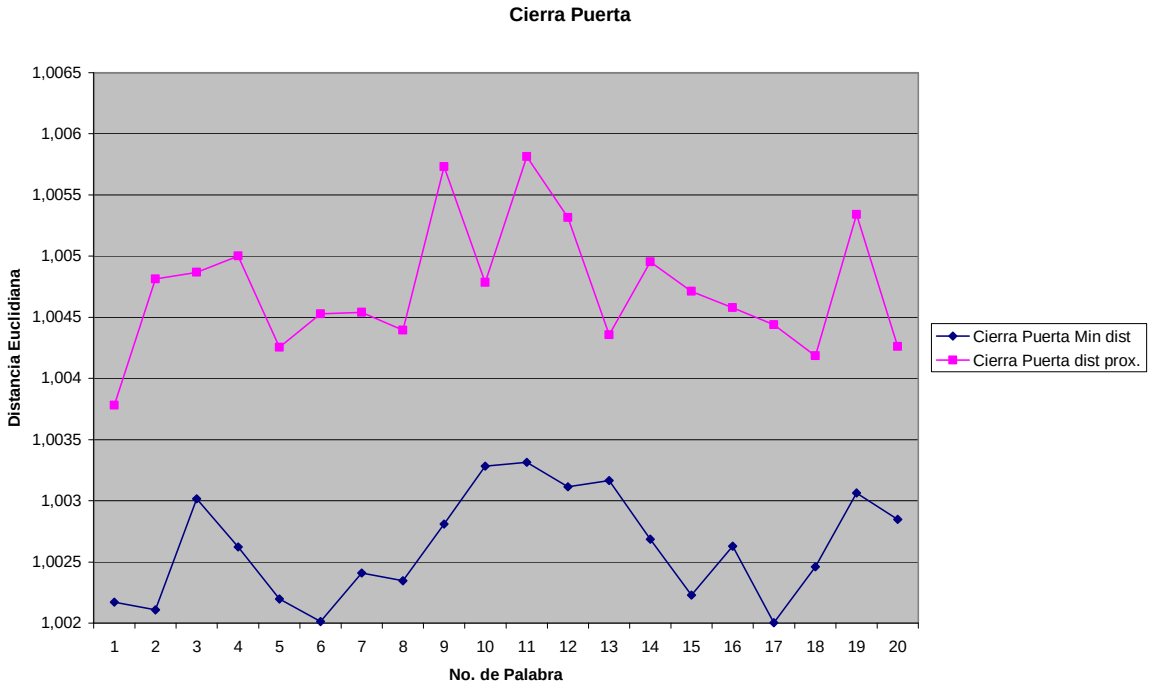

Figura V.15. Representación gráfica para la palabra Cierra Puerta y su distancia próxima para el método de reconocimiento LPC-Cepstral

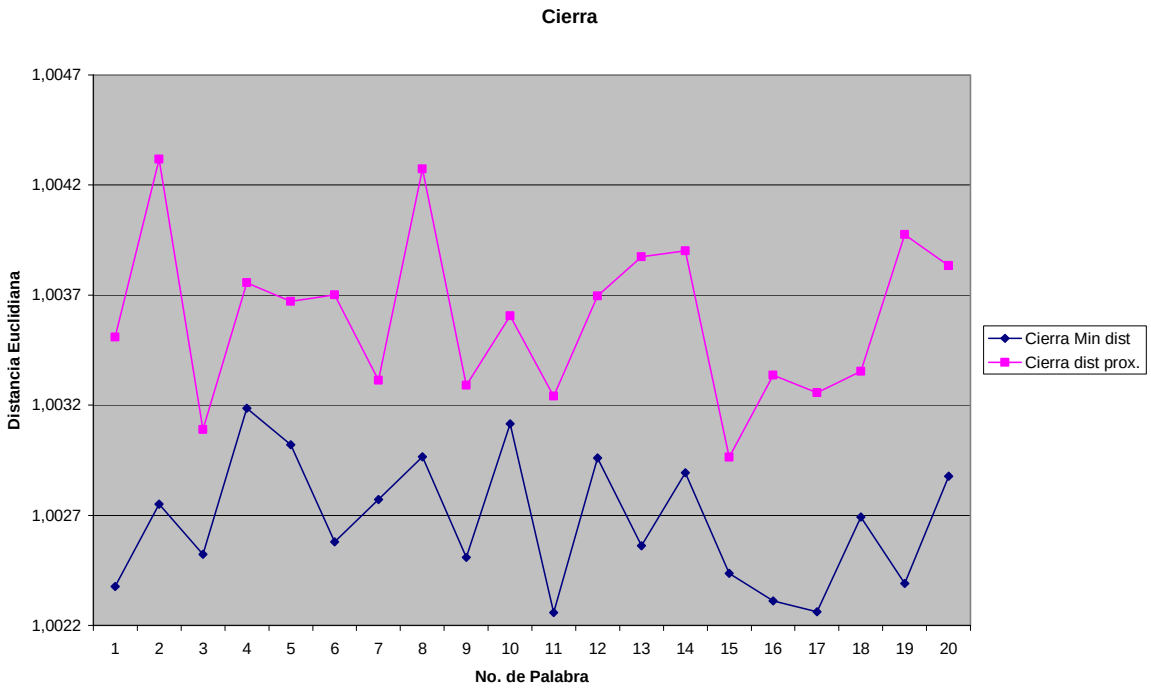

Figura V.16. Representación gráfica para la palabra Cierra y su distancia próxima para el método de reconocimiento LPC-Cepstral

En esta parte de la realización del proyecto se llevaron a cabo los mismos pasos a seguir que en primer caso. Obtenidos lo umbrales de reconocimiento se procedió a obtener ya de manera estadística, resultados de reconocimiento ya de en tiempo real en la tarjeta desarrollo. Los resultados de estas estadísticas fueron muy satisfactorio, pero cabe mencionar que el reconocimiento fue mas restringido, ya que como se mostrara más adelante y como se intentó en esta tesis, comprobar que el segundo método implementado, además de ser mas robusto y confiable, hace discriminación de reconocimiento en los hablantes no entrenados. A esto se puede decir que este sistema con un adecuado sistema de captura de sonido y aislamiento de ruido aumentaría el reconocimiento.

Al referirme a que el sistema puede tener mejoras para aumentar el reconocimiento, señalo que debe de implementarse un sistema un poco más complejo de captura de sonido para aumentar, tanto la etapa de entrenamiento, así como la etapa de reconocimiento. En la etapa de captura se puede implementar un amplificador de micrófono inmune al ruido, tanto ambiental así como al ruido de la alimentación de línea. El micrófono se puede sustituir por un micrófono unidireccional, esto con el propósito de que sea mas limpia la señal además de hacer que el usuario tenga una distancia más corta en el momento del reconocimiento, este tipo de micrófonos también tienen la peculiaridad de que el ruido ambiental es minimizado.

En las gráficas que se muestran a continuación se observa el comportamiento del reconocimiento ya de manera real. Las estadísticas representan el reconocimiento obtenido, como se puede observar, este reconocimiento esta por arriba del 80 % en todos los casos. Por lo anterior se afirma que el sistema obtenido por medio de la programación en tiempo real además de la fase del entrenamiento que se realizó es confiable.

De la figura V.17 a la V.20 se muestra el reconocimiento obtenido con el método LPC-Cepstral. De manera gráfica se puede observar que el reconocimiento obtenido ya, en la tarjeta de desarrollo es confiable. Se pueden tener mejoras en el sistema como algunas mencionadas anteriormente y mejorar estos resultados.

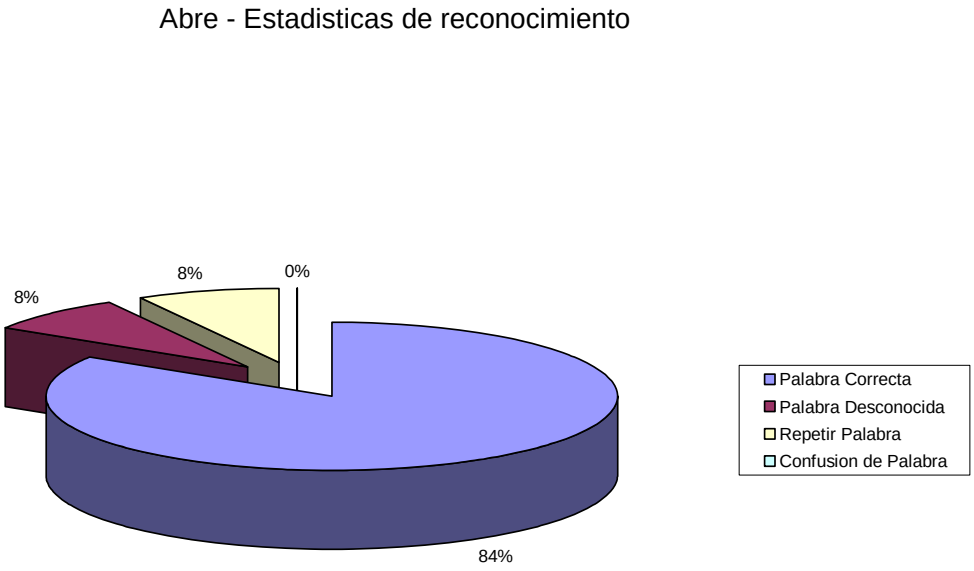

Figura V.17. Gráfica de estadísticas de reconocimiento para la palabra *abre* obtenidas del método LPC-Cepstral

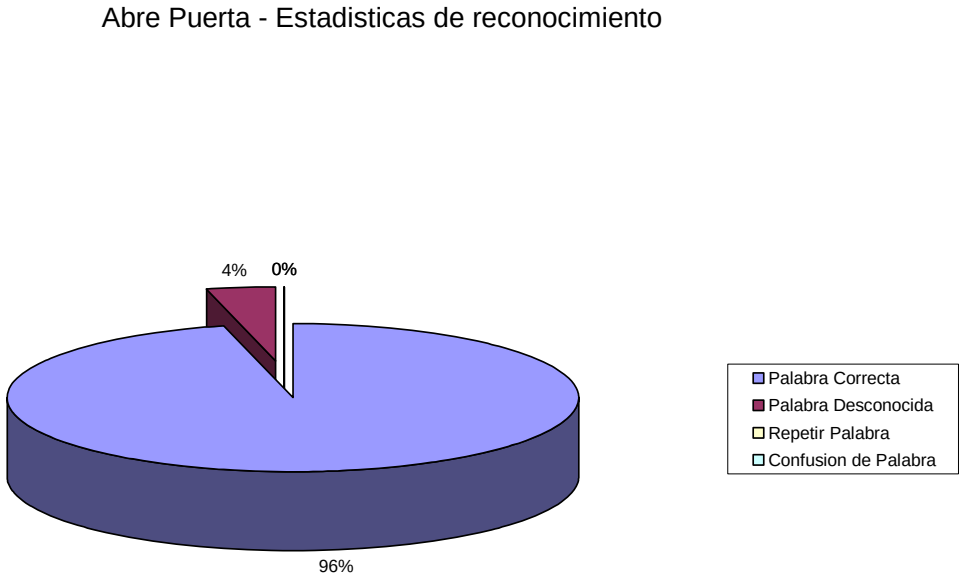

Figura V.18. Gráfica de estadísticas de reconocimiento para la palabra *abre puerta* obtenidas del método LPC-Cepstral

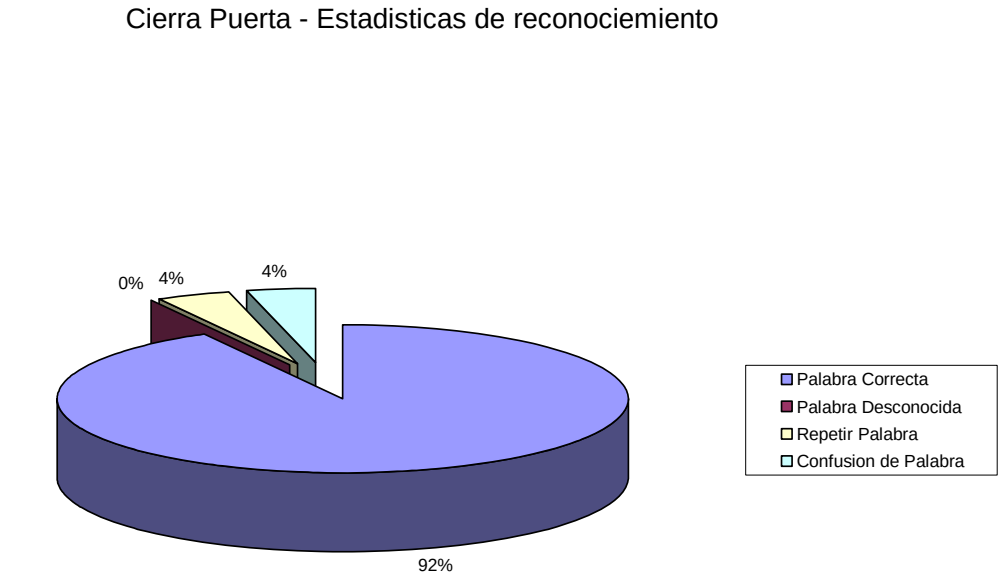

Figura V.19. Gráfica de estadísticas de reconocimiento para la palabra *cierra puerta* obtenidas del método LPC-Cepstral

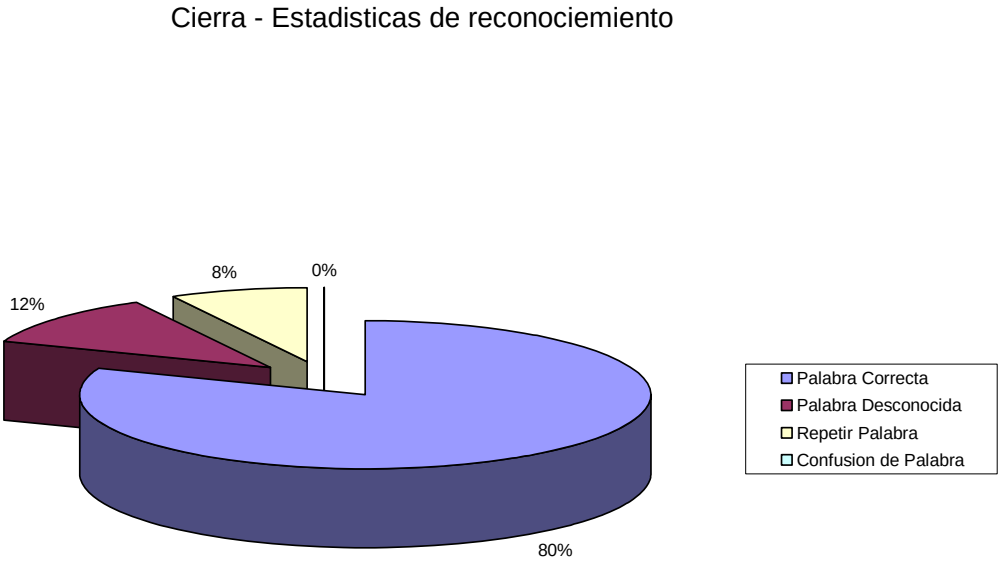

Figura V.20. Gráfica de estadísticas de reconocimiento para la palabra *cierra* obtenidas del método LPC-Cepstral

5.2.3 Resultados LPC-Cepstral con un hablante no entrenado

Como se ha mencionado ya con anterioridad, con el segundo método de esta tesis se trato de demostrar que el algoritmo LPC-Cepstral hace diseminación de hablantes. El reconocimiento con hablantes no entrenados se llevó a cabo en nuestro sistema de reconocimiento, arrojando muy buenos resultados en su desarrollo. Se ocuparon tres hablantes no entrenados en esta sección de obtención de resultados.

Los hablantes fueron seleccionados del mismo género y con una edad aproximada, con el propósito de obtener las condiciones más próximas del usuario entrenado. De esta manera ya en la implementación en tiempo real en nuestra tarjeta de desarrollo, se procedió a hacer el reconocimiento de nuestros hablantes.

Se obtuvo que dos de nuestros candidatos a las pruebas de reconocimiento fueron totalmente excluidos en el reconocimiento y el tercer hablante fue reconocido de manera errónea en muy pocas ocasiones y en los demás casos fue rechazado por el sistema. Cabe recalcar que estas pruebas se llevaron acabo en la tarjeta de desarrollo de manera que el proyecto realizado llevó acabo todo este proceso de reconocimiento en tiempo real.

Estos resultados indican que nuestro algoritmo de reconocimiento hace una discriminación con hablantes no entrenados. Por consiguiente, si un hablante quiere ser reconocido en el sistema debe de ser previamente entrenado.

Con lo previamente observado en los resultados se debe mencionar que el resultado obtenido se debe de deber a la robustez del algoritmo de reconocimiento LPC-Cepstra. Los hablantes no entrenados objetos de nuestro estudio en esta ultima parte de nuestro análisis, cabe mencionar, que fueron sometidos al entrenamiento en las mismas condiciones de reconocimiento, a esto debo referirme que el ambiente de ruido y fue el mismo que en los caso del hablante entrenado.

La implementación fue muy exitosa, ya que los resultados obtenidos en la tarjeta de desarrollo, cumplieron con las expectativas esperadas. En el transcurso de las pruebas de reconocimiento entre los hablantes, se esperaba que los hablantes ajenos al entrenamiento fueran descartados por el sistema, aunque hubo en uno de ellos un reconocimiento, siendo este casi nulo o erróneo.

Tomando en cuenta esta ultima parte, se puede decir que el sistema es confiable, realizando posibles mejoras que se pueden implementar tanto en programación así como en aditamentos electrónicos, de audio y un sin fin de modificaciones realizables de optimización, lo cual incrementaría el rendimiento del sistema, los cuales son viables para una mejora sustancial del proyecto. El algoritmo de reconocimiento fue implementado con las especificaciones y recomendaciones según Rabiner.

	Palabra Correcta	Palabra Desconocida	Repetir Palabra	Confusion de Palabra	
Abre	0	23	0	2	Cierra Puerta
Abre Puerta	2	23	0	0	
Cierra	3	0	0	0	
Cierra Puerta	0	19	1	6	4 Abre Puerta / 2 Cierra Puerta

Tabla V.6. Matriz de confusión para una persona ajena al entrenamiento

En la matriz de confusión se presentan reconocimientos correctos, aún cuando nuestro individuo a reconocer no fue entrenado el sistema. Se debe mencionar que en el caso de este reconocimiento fueron tres personas de prueba para el sistema. Dos de ellas fueron totalmente descartadas, esto quiere decir que en ningún momento dos personas de las tres a prueba, fueron reconocidas en ninguna de sus repeticiones. Solo una de estas tres personas fue reconocida aunque erróneamente en la mayora de los caso, hubo un reconocimiento demaciado bajo para el

reconocimiento acertado este fue de un 12 % en el caso de la palabra *cierra* y un 4% en el caso de *cierra puerta*.

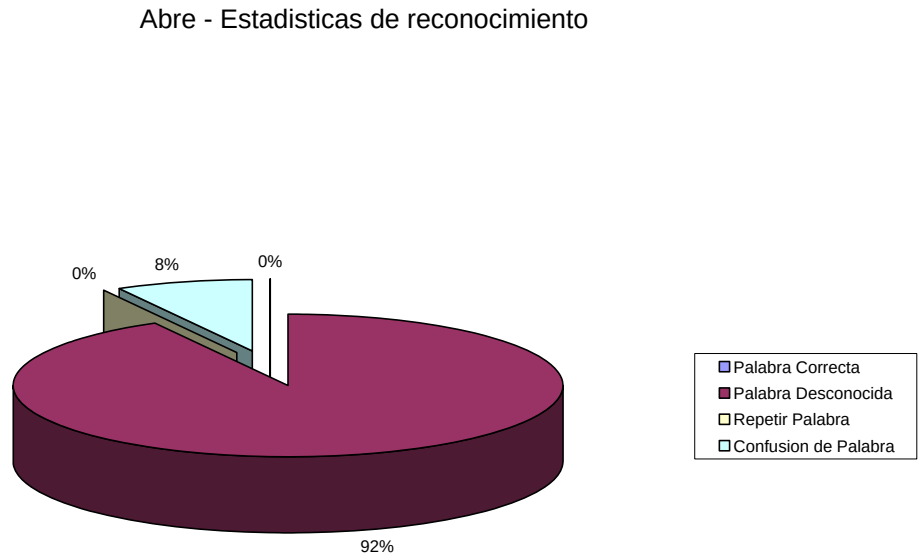

Figura V.21. Gráfica de estadísticas de reconocimiento para la palabra *abre* obtenidas del método LPC-Cepstral, para una persona ajena al entrenamiento

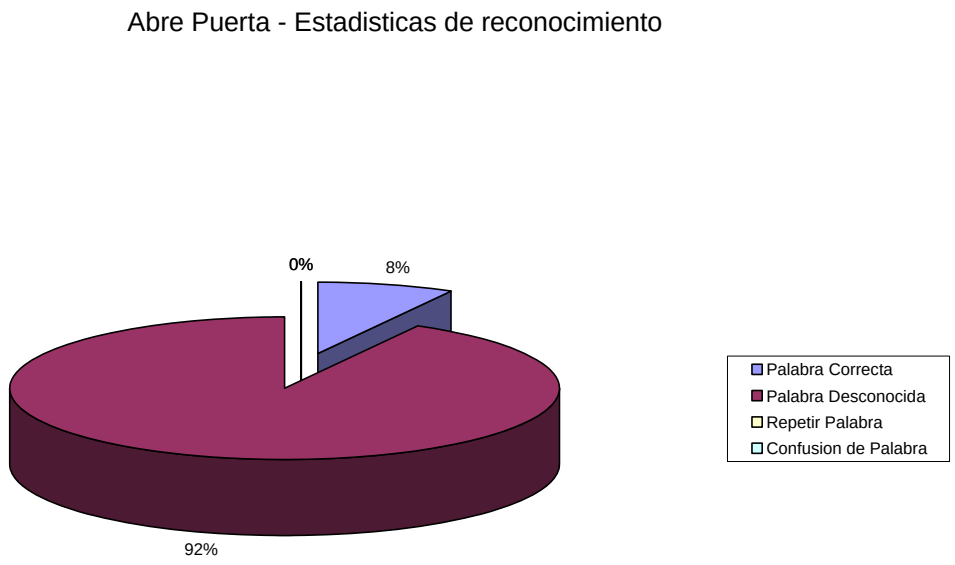

Figura V.22. Gráfica de estadísticas de reconocimiento para la palabra *abre puerta* obtenidas del método LPC-Cepstral, para una persona ajena al entrenamiento

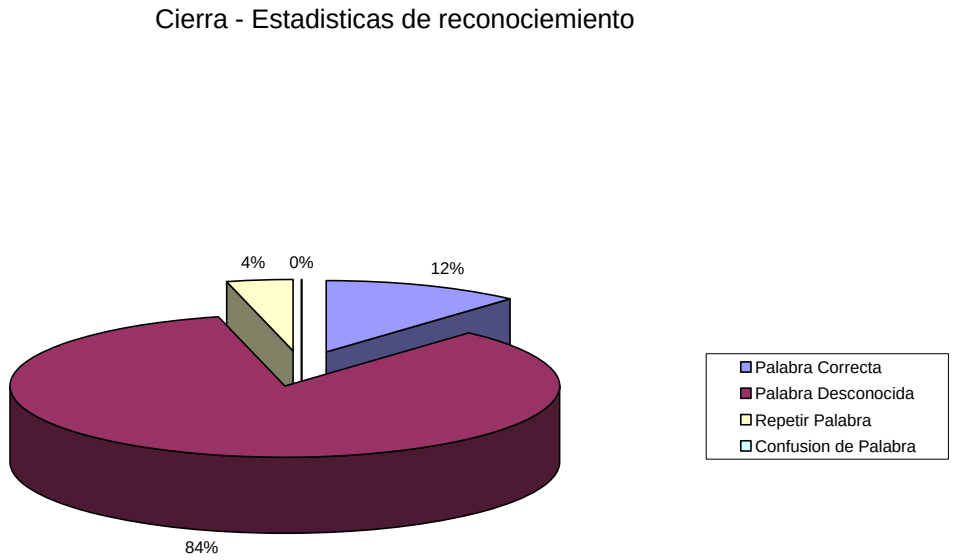

Figura V.23. Gráfica de estadísticas de reconocimiento para la palabra *cierra* obtenidas del método LPC-Cepstral, para una persona ajena al entrenamiento

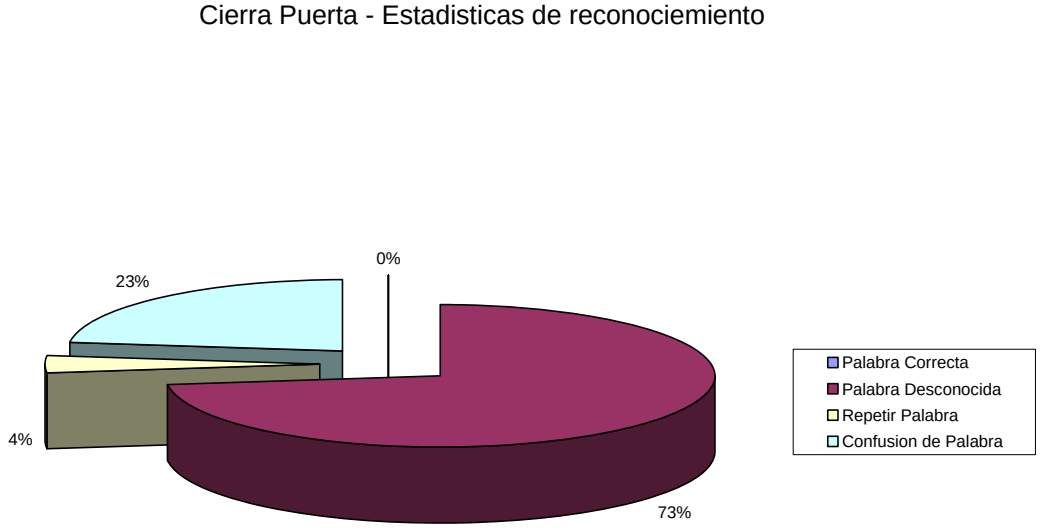

Figura V.24. Gráfica de estadísticas de reconocimiento para la palabra *cierra puerta* obtenidas del método LPC-Cepstral, para una persona ajena al entrenamiento

Los resultados obtenidos, fueron demasiado satisfactorios ya que se cumplieron las expectativas planteadas en esta tesis. Se demostró con una serie de experimentos que el método Cepstral es más robusto que el método LPC. Además de que este método hace una selección relativamente buena de los individuos no entrenados del sistema.

La estadísticas obtenidas muestran la efectividad de nuestro método, de manera que se pueden decir ambos métodos aunque son la base de algoritmos más complejos son confiables y se pueden implementar ambos de manera confiable dependiendo del uso o la implementación que se requiera, es el algoritmo a implementar.

BIBLIOGRAFÍA

[1] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals*. Prentice Hall, 1978.

[2] A. Herrera, R. Ibarra y V.R. Algazi, *An Isolated Speech Recognition Method Using KLT*. Proceedings of the IASTED International Conference on Signal and Image Processing, pp.26-29, 1999.

[3] O. Nieto, V. López y A. Herrera*. Mexican Spanish Speech Commands Recognition Using the TMS320C6711 DSP*. Proceedings of GSPx-2003 International Signal Processing Conference, CD, 2003.

[4] Signal modeling techniques in speech recognition. *Proceedings of the IEEE* [en línea]. Vol. 81 núm. 9 [Consulta:
28/06/2006]. Disponible a: <http://ieeexplore.ieee.org/servlet/opac?punumber=5>.

[5] A review of large vocabulary continuous-speech recognition. *IEEE signal processing magazine* [en línea].
september, 1996 [Consulta: 28/06/2007]. Disponible a:
<http://ieeexplore.ieee.org/servlet/opac?punumber=79>.

[6] , *OShaughnessy, D.* Speech Communications; Human and Machine, *Second Edition. IEEE Press. 2000*

[7] *Deller, J; Proakis, J.G.; Hansen, J.H.L.,* Discrete-Time Processing of Speech Signals, *Macmillan Publishing Company. New York, 1993.*

[8] Xuedong, Huang. Acero, Alez. Hon-Wen. SPOKEN LANGUAJE PROCESSING: A GUIDE TO THEORY, ALGORITHM AND SYSTEM DEVELOPMENT. Prentice Hall

[9] Jurafsky Daniel & Martin James. *Speech and Language Processing – An introduction to Natural Language Processing, Conputational Linguistic and Speech Recognition*, Prentice- Hall Inc. New Jersey 2000.

[10] [Consulta:28/06/2006]. Disponible a http://www.dspexperts.com/dsp/projects/lpc/

[11] J. Proakis, Ch.D.G. Manolakis; “Tratamiento Digital de Señales”, Prentice - Hall, 1998.

[12] D.G. Jamieson1, L. Deng2, M. Price1 , Vijay Parsa1 and J. Till3, INTERACTION OF SPEECH DISORDERS WITH SPEECH CODERS: EFFECTS ON SPEECH INTELLIGIBILITY http://www.asel.udel.edu/icslp/cdrom/vol2/516/a516.pdf

[13] Huang, Acero y Hon, *Spoken Language Processing*,Prentice-Hall, 2001.

[14] Duda, Harty Stork, *Pattern Classification*, John Wiley & Sons, 2001.

[15] A. Gershoy R. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Press, 1992.

[16] R. Gray, Vector Quantization, *IEEE ASSP Magazine*, 1(2), 1984.

[17] B. Juang, D. Wang, A. Gray, Distortion Performance of Vector Quantization for LPC Voice Coding, *IEEE Trans ASSP*, 30(2), 1982.

[18] J. Makhoul, S. Roucos, H. Gish, Vector Quantization in Speech Coding, *Proc. IEEE*, 73(11), 1985.

[19] L. Rabiner y B. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993.

[20] www.national.com

[21] MOC3009 THRU MOC3012, OPTOCOUPLERS/OPTOISOLATORS. User Manual, SOES024A – AUGUST 1985 – REVISED APRIL 1998

[22] Linear Predictive Coding and Cepstrum coefficients for mining time variant information from software repositories. Giuliano Antoniol, RCOST- University Of Sannio Via Traiano 1
82100, Benevento (BN), ITALY