UNIVERSIDAD PRIVADA DE TACNA

ESCUELA DE POSTGRADO

MAESTRÍA EN INGENIERÍA ELECTRÓNICA CON MENCIÓN EN TELECOMUNICACIONES

DESARROLLO DE UN CODIFICADOR PARAMÉTRICO DE VOZ (VOCODER) UTILIZANDO EL MODELO DE PREDICCIÓN LINEAL DEL ESTANDAR FS-1015

TESIS

Presentada por:

Br. Heraclio Henry Gómez del Carpio

ORCID: 0000-0002-0060-6761

Asesor:

Mag. Anibal Juan Espinoza Aranciaga

ORCID: 0000-0001-6489-5112

Para Obtener el Grado Académico de:

MAESTRO EN INGENIERÍA ELECTRÓNICA CON MENCIÓN EN TELECOMUNICACIONES

TACNA – PERÚ

2021

UNIVERSIDAD PRIVADA DE TACNA

ESCUELA DE POSTGRADO

MAESTRÍA EN INGENIERÍA ELECTRÓNICA CON MENCIÓN EN TELECOMUNICACIONES

Tesis

"DESARROLLO DE UN CODIFICADOR PARAMÉTRICO DE VOZ (VOCODER) UTILIZANDO EL MODELO DE PREDICCIÓN LINEAL DEL ESTANDAR FS-1015"

Presentada por:

Br. Heraclio Henry Gómez del Carpio

Tesis sustentada y aprobada el 17 de diciembre del 2021; ante el siguiente jurado examinador:

PRESIDENTE: **Mag. Tito Leoncio CÓRDOVA MIRANDA**

SECRETARIO: **Mag. Hugo Javier RIVERA HERRERA**

VOCAL: **Mag. María Hilda BERMEJO RÍOS**

ASESOR: **Mag. Anibal Juan ESPINOZA ARANCIAGA**

DECLARACIÓN JURADA DE ORIGINALIDAD

Yo **Heraclio Henry GÓMEZ DEL CARPIO**, en calidad de Egresado de la Maestría en Ingeniería Electrónica de la Escuela de Postgrado de la Universidad Privada de Tacna, identificado con DNI 00790033.

Soy autor de la tesis titulada: **DESARROLLO DE UN CODIFICADOR PARAMÉTRICO DE VOZ (VOCODER) UTILIZANDO EL MODELO DE PREDICCIÓN LINEAL DEL ESTANDAR FS-1015.**

DECLARO BAJO JURAMENTO

Ser el único autor del texto entregado para obtener el grado académico de **Maestro en Ingeniería Electrónica con mención en Telecomunicaciones**, y que tal texto no ha sido entregado ni total ni parcialmente para obtención de un grado académico en ninguna otra universidad o instituto, ni ha sido publicado anteriormente para cualquier otro fin.

Así mismo, declaro no haber trasgredido ninguna norma universitaria con respecto al plagio ni a las leyes establecidas que protegen la propiedad intelectual.

Declaro, que después de la revisión de la tesis con el software Turnitin se declara **25%** de similitud, además que el archivo entregado en formato PDF corresponde exactamente al texto digital que presento junto al mismo.

Por último, declaro que para la recopilación de datos se ha solicitado la autorización respectiva a la empresa u organización, evidenciándose que la información presentada es real y soy conocedor (a) de las sanciones penales en caso de infringir las leyes del plagio y de falsa declaración, y que firmo la presente con pleno uso de mis facultades y asumiendo todas las responsabilidades de ella derivada.

Por lo expuesto, mediante la presente asumo frente a LA UNIVERSIDAD cualquier responsabilidad que pudiera derivarse por la autoría, originalidad y veracidad del contenido de la tesis, así como por los derechos sobre la obra o invención presentada. En consecuencia, me hago responsable frente a LA UNIVERSIDAD y

a terceros, de cualquier daño que pudiera ocasionar, por el incumplimiento de lo declarado o que pudiera encontrar como causa del trabajo presentado, asumiendo todas las cargas pecuniarias que pudieran derivarse de ello en favor de terceros con motivo de acciones, reclamaciones o conflictos derivados del incumplimiento de lo declarado o las que encontrasen causa en el contenido de la tesis, libro o invento.

De identificarse fraude, piratería, plagio, falsificación o que el trabajo de investigación haya sido publicado anteriormente; asumo las consecuencias y sanciones que de mi acción se deriven, sometiéndome a la normatividad vigente de la Universidad Privada de Tacna.

Tacna, 17 de diciembre del 2021

Heraclio Henry Gómez del Carpio
DNI: 00790033

Dedicatoria

Dedico está tesis a mis hijos Henry, Herbert y Evelyn, que siempre han sido mi fuente de inspiración y de motivación.

Agradecimientos

Quiere agradece profundamente a todos las personas que intervinieron para que este proyecto se haga realidad.

Índice de contenido

Índice de tablas

Índice de figuras

Resumen

El principal objetivo que se propone en este trabajo de investigación, es el desarrollo de un códec de voz de baja tasa de bits, utilizando la codificación predictiva lineal (LPC) según el estándar FS 1015. Se propuso todos los bloques requeridos según el mencionado estándar, tanto en el codificador como el decodificador.

Para determinar si un bloque es considerado como vocal o no vocal, se entrenó un clasificador estadístico lineal basado en máquinas de soporte vectorial.

Todos los archivos se voz tomaron de la base de datos abierta Open SLR. La evaluación de la calidad de la voz sintetizada se realizó con la métrica de calidad objetiva ODG (Objective Difference Grade), que se basa en la recomendación ITU BS.1387-1.

Finalmente, se obtuvo el resultado esperado: un codificador de voz basado en el estándar FS 1015 con baja tasa de bits, pero también con bajo nivel de calidad.

Palabras clave: Predicción lineal, codificador de voz y tasa de bits

Abstract

The main objective proposed in this research work is the development of a low bit rate voice codec, using linear predictive coding (LPC) according to the FS 1015 standard. All the required blocks were proposed according to the aforementioned standard, both in the encoder and the decoder.

To determine whether a block is considered as vowel or non-vowel, a linear statistical classifier based on vector support machines was trained.

All voice files were taken from the Open SLR database. The evaluation of the quality of the synthesized voice was carried out with the objective quality metric ODG (Objective Difference Grade), which is based on the ITU BS.1387-1 recommendation.

Finally, the expected result was obtained: a speech coder based on the FS 1015 standard with a low bit rate, but also, a low quality level.

Keywords: Linear prediction, vocoder and bit rate.

Introducción

Actualmente con la aparición de nuevas tecnologías de la información, el desarrollo de nuevos dispositivos electrónicos para aplicaciones específicas y el uso de herramientas de simulación, se ha tratado de modelar y emular las condiciones de un tracto vocal, que reproducen los sonidos que componen la voz humana. Así mismo, por medio de dispositivos cada vez más sofisticados, con la finalidad de verificar e identificar la voz de las personas, se han desarrollado algoritmos computacionales que permitan realizar grandes cantidades de operaciones en muy poco tiempo, lo que da como resultado el procesamiento digital de señales en tiempo real.

El presente trabajo propone el desarrollo de un códec de voz LPC (Linear Predictive Coding), de baja tasa de bits según el estándar FS 1015. En primer lugar, se proponen todos los bloques requeridos según el mencionado estándar, tanto en el codificador como el decodificador. El vocoder LPC estándar proporciona una función de análisis en el extremo transmisor y una función de síntesis en el extremo receptor.

Por el lado del codificador, significa la implementación de: segmentación de bloques de 240 muestras, filtro de pre-énfasis, detector de voz/no voz, algoritmo de predicción lineal, cálculo de error de predicción, estimación de pitch y de potencia, así como de las tablas de búsqueda para codificar y cuantizar los coeficientes de reflexión de la predicción lineal, el pitch y la potencia. Por el lado del decodificador, se ha implementado: tablas de búsqueda para la decodificación de los coeficientes de reflexión de predicción lineal, pitch y potencia, así como el proceso de síntesis de predicción lineal con excitación de tren de impulsos (caso vocal) o ruido blanco (caso no vocal).

Para determinar si un bloque es considerado como vocal o no vocal, se entrenó un clasificador estadístico lineal basado en máquinas de soporte vectorial. Para el entrenamiento estadístico, se analizó 890 bloques de voz extraídos de 7 archivos de voz de hombre y 7 archivos de voz de mujer, y clasificados manualmente. Asimismo, se extrajo una base de datos de 20 archivos de voz (10 de hombre y 10 de mujer) para las pruebas sucesivas. Todos los archivos de voz tomaron de la base de datos abierta Open SLR.

Finalmente, la evaluación de la calidad de la voz sintetizada se realizó con la métrica de calidad objetiva ODG (Objective Difference Grade), que se basa en la recomendación ITU BS.1387-1. La principal ventaja de utilizar esta métrica es que, a diferencia de métricas subjetivas como MOS (Mean Opinion Score), el ODG se obtiene a partir de un algoritmo automático que implementa un modelo psico-acústico basado en la percepción auditiva humana, por lo que no se requiere de participantes para obtener el resultado de calidad. Como consecuencia de ello, se obtuvo el resultado esperado: un codificador de voz basado en FS 1015, con baja tasa de bits, pero también con bajo nivel de calidad.

Esta investigación se ha desarrollado en cinco capítulos. En el capítulo I, se desarrolla el planteamiento del problema y la justificación; así mismo, se plantean los objetivos específicos y el objetivo general al cuál apunta el presente trabajo.

En el capítulo II, se realiza una recopilación de información proveniente de trabajos de investigación, tesis y artículos científicos, para conocer a profundidad los fundamentos de las dos variables con las que se operan mi proyecto: modelo de predicción lineal y el codificador paramétrico de voz, En este capítulo, se establece el fundamento teórico para el diseño de los instrumentos.

En el capítulo III, se presenta el marco metodológico de la investigación: Se plantean la hipótesis general y las hipótesis específicas, así como la operacionabilidad de las variables.

En el capítulo IV, se realiza un diagnóstico situacional de la problemática. Se analiza los factores críticos y se evalúa como resolver la dificultad encontradas

para el desarrollo del trabajo de investigación.

En el capítulo V, se hace la descripción del codificador paramétrico de voz y d su estructura funcional.

En el capítulo VI, se analiza los resultados de la simulación del vocoder propuesto y de su funcionalidad, así como de los cambios relevantes de la aplicación.

En el capítulo VII, se describen las conclusiones y las recomendaciones resultantes de la investigación, que permitan verificar que se cumple con lo propuesto en el presente trabajo de tesis, así como las mejoras que se pueden efectuar.

Capítulo I: El problema

1.1 Planteamiento del problema

Los sistemas de comunicación electrónica actuales requieren transmitir todo tipo información (audio, video y datos) de alta calidad, para ello se han desarrollado técnicas de procesamiento de señales digitales más avanzadas y sofisticadas que utilizan un canal digital.

El proceso de codificación digital de la voz ha tomado mucha importancia en el establecimiento de una comunicación hablada.

Este proceso consiste en un conjunto de transformaciones de la señal de voz a transmitir con el fin de mejorar la eficiencia sin pérdida de la calidad de la información.

Uno de los principales problemas del procesamiento digital de la voz es la disposición de ancho de banda del canal digital requerido.

Las diferentes técnicas de codificación de voz han permitido reducir el uso de la tasa de transmisión (bit rate) del canal digital, mediante el uso de algoritmos de compresión de voz de 40, 32, 24, 16 kbps.

La codificación de audio digital ha sido a lo largo de los últimos 20 años un campo de aplicación del procesado de señales. Basta con enumerar una serie de campos de aplicación:

- Almacenamiento en discos ópticos y dispositivos portátiles.
- Audio asociado para vídeo digital.

- Transmisión de audio mediante redes digitales.

A partir del año 2000, la transmisión por Internet ha obligado a reducir drásticamente el uso de la tasa de transmisión (bit rate) del canal digital e incrementar el tiempo de almacenamiento de la señal en dispositivos móviles de bajo costo. Para lograr este fin se ha apuntado la necesidad de cambiar la tecnología de codificación, pasando de utilizar las transformadas tiempo-frecuencia para codificar la forma de onda de la señal, a desarrollar modelos de señal que extraigan parámetros de la señal de audio que son posteriormente codificados.

Una solución al problema es la utilización del modelo de predicción lineal (LPC) que permite estimar los parámetros más importantes de los codificadores de voz (vocoders).

Este algoritmo basado en el estándar FS-1015 de comprensión digital, permitirá reducir en un alto grado la utilización de la tasa de transmisión (bits rates) del canal digital, obteniéndose tasas de hasta 2.4 kbps.

1.2 Formulación del problema

1.2.1 Interrogante principal.

¿En qué medida el empleo del modelo de predicción lineal (LPC) basado en el estándar FS-1015, permitirá el desarrollo de un codificador paramétrico de voz (vocoder) a bajas tasas de transmisión de bits?

1.2.2 Interrogantes secundarias.

¿Cómo incide en el modelamiento de un codificador paramétrico de voz, los principales parámetros de un modelo de predicción lineal basado en el estándar FS-1015?

¿Cómo la simulación en la plataforma de desarrollo MATLAB del proceso de compresión de voz utilizando el estándar FS-1015, permite el desarrollo de un codificador de voz (vocoder)?

¿Cómo los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, determina los resultados del procesamiento elegido?

1.3 Justificación de la investigación

El presente trabajo de tesis se desarrolla con la finalidad de impulsar la investigación científica y tecnológica en la utilización de las técnicas de comprensión digital en el procesamiento y codificación de señales de voz.

Los motivos que me llevó a investigar sobre el tema a tratar en la propuesta del presente trabajo de tesis magistral fueron:

En primer lugar, todo avance tecnológico se fundamenta en varias etapas: investigación básica, investigación aplicada, desarrollo y producción. En esta tesis magistral se pretende desarrollar un trabajo que combina las principales características de la investigación básica y de la investigación aplicada.

En segundo lugar, mediante el empleo de las interfaces gráficas de la plataforma de desarrollo MATLAB, simular el desarrollo de un codificador de voz (vocoder) que permita analizar el proceso de compresión y reconstrucción de una señal de voz a partir de los parámetros de predicción lineal del estándar FS-1015.

En tercer lugar, presentar una metodología que constituya una herramienta para el estudio del procesamiento digital de voz y la codificación de predicción lineal.

1.4 Objetivos de la investigación

1.4.1 Objetivo general.

Desarrollar un codificador paramétrico de voz (vocoder) a bajas tasas de transmisión de bits, empleando el modelo de predicción lineal (LP) basado en el estándar FS-1015.

1.4.2 Objetivos específicos.

Identificar los principales parámetros del modelo de predicción lineal aplicados a las señales de voz, que faciliten el modelamiento de un codificador de voz (vocoder).

Simular en la plataforma de desarrollo MATLAB el proceso de compresión de voz utilizando el estándar FS-1015, que permita el desarrollo de un codificador de voz (vocoder).

Analizar los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, que determine los resultados del procesamiento elegido.

1.5 Alcances de la investigación

En el desarrollo del presente trabajo, debo precisar los alcances de la investigación:

Se propusondrá en esta investigación, el desarrollo de un vocoder siguiendo el modelo de predicción lineal (LP) del estándar FS-1015.

Se utilizará el software de aplicación matemática MATLAB para la simulación del vocoder propuesto.

Lo que se pretende con esta investigación, es una metodología para el desarrollo de codificador de voz utilizando otros estándares que los regulan.

Para evaluar la calidad del audio procesado (señal sintetizada), se utilizó la métrica denominada grado de diferencia objetiva (ODG), regulada por el estándar ITU-R BS.1387, en lugar de la métrica MOS (Mean Opinion Score), que evalúa la calidad de modo subjetiva al realizar pruebas comparativas con otro codificador.

Si bien es cierto que se propuso desarrollar un vocoder de baja calidad, ello no significó que la voz sintetizada no se pueda entender o sea ininteligible.

Capítulo II: Marco teórico

2.1 Antecedentes de la investigación

2.1.1 Internacionales

Gallego, Correa, Blanco y Álvarez (2017) en el artículo "La predicción lineal aplicada al reconocimiento distribuido del habla en redes IP", presentaron las siguientes conclusiones:

> La predicción lineal se muestra como un método efectivo para realizar el reconocimiento de palabras aisladas.
> En este reconocimiento el proceso de cuantificación de parámetros solo implica un aumento considerable del coste computacional. Las pérdidas de información que puedan ocurrir en esquemas de reconcomiendo distribuido, no afectan la efectividad del reconocimiento al emplear la predicción lineal, por lo que esta técnica se presenta como un método robusto para el reconocimiento automático del habla. (Gallego, Correa, Blanco y Álvarez, 2017, p. 146)

Domínguez, García, Gallego, Correa y Rodríguez (2011) en el artículo

científico “Codificación de voz mediante coeficientes de predicción lineal (LPC) sobre Microblaze”, llegaron a las siguientes conclusiones:

> Se diseñó e implementó el codec para voz IP LPC10 soportado sobre un microcontrolador Microblaze empotrado en una tarjeta Spartan 3E, teniendo en cuenta todas las recomendaciones que ofrece el algoritmo Lineal Prediction Code, para las transmisiones de voz a 2.4kbp.
> Se comprobó además el correcto funcionamiento del mismo evaluando su calidad de servicio y comparándolo con un procesamiento similar en Matlab, permitiendo la evaluación de cada uno de los bloques funcionales que conforman el algoritmo de codificación, mediante la herramienta de simulación virtual que proporciona XPS (Domínguez, García, Gallego, Correa y Rodríguez, 2011, p. 60).

Vera (2006) en su trabajo de Tesis Doctoral “Desarrollo de técnicas de codificación de audio basadas en modelos de señal paramétricos”, llegó a la siguiente conclusión:

> El uso de modelos de señal paramétricos en una aplicación de codificación de audio orientada a realizar streaming por internet permite el desarrollo de un codificador completamente paramétrico.
> Este codificador divide la señal en tonos, transitorios y ruido. Con un modelo tonal con guiado y parada perceptuales, un modelo de transitorios con un diccionario mixto formado por funciones wavelet packets y exponenciales complejas, y un modelo de ruido que obtiene la envolvente en frecuencia mediante warped-LPC y la envolvente temporal mediante LPC, es posible diseñar un codificador paramétrico de audio con un régimen binario medio de 16 Kbit/s y una buena calidad subjetiva de señal (Vera, 2006, p. 245).

Ortiz (2006) en su trabajo de Tesis de Titulación "Determinación de coeficientes LPC en tiempo real utilizando un DSP de TI, para señales de voz", llegó a la siguiente conclusión:

> Cuando se realiza la simulación en MATLAB, debemos tener presente el número de coeficientes con los que vamos a estar trabajando, ya que, el sistema asigna un valor de uno al inicio del primer coeficiente u origen de los vectores (LPC's). Lo anterior es más evidente al observar las gráficas que se obtuvieron en la simulación (capitulo 4). Debido al origen de los vectores (LPC's), debemos de colocar el valor deseado de coeficientes más uno en el programa. De esta forma seguro tendremos el número de coeficientes deseados y podremos recuperar mejor la señal. Por otro lado, el grado de definición de la gráfica no siempre nos permitirá observar la existencia de los coeficientes buscados, motivo por el cual, siempre debemos respaldar la gráfica con información de los valores de los mismos. También es importante tener en cuenta la velocidad de procesamiento de la PC en donde se emplea el MATLAB, ya que el tiempo de respuesta varia con respecto al tamaño de la señal (Ortiz, 2006, p. 37).

Orellana (2002) en la Tesis de Titulación "Análisis y simulación de compresión de voz utilizando códigos de predicción lineal"**,** llegó a establecer las siguientes conclusiones:

> En la implementación del modelo de compresión de voz utilizando códigos de predicción lineal se ha elegido el método de la autocorrelación para el cálculo de los coeficientes del filtro ya que además de garantizar la estabilidad del filtro, mantiene el error de predicción en su valor mínimo y permite el cálculo de los coeficientes de reflexión con la ayuda del algoritmo de Levinson-Durbin.

> Los programas de simulación constituyen en la actualidad herramientas de gran utilidad en la investigación de procesos, en el caso del tratamiento digital de señales, la implementación de un programa de simulación permite evaluar los procedimientos a llevarse a cabo, las ventajas y desventajas del so de un determinado algoritmo de cálculo, así como también la dificultad en cuanto a los cálculos y a la calidad de la señal resultante (Orellana, 2002, p. 157).

2.1.2 Nacionales

Llanos (2016) en su informe técnico denominado "Análisis por predicción lineal para señales de voz", indica que:

> El objetivo principal de este trabajo es utilizar herramientas matemáticas para modelar el sistema del tracto vocal humano, para lo cual se han utilizado tubos de sección constante y longitud infinita, y tubos de diámetros variables y longitud infinita.
>
> Aplicando señales de presión se obtuvieron ecuaciones matemáticas que van a servir para representar la dinámica del comportamiento tracto vocal y a su vez el modelo de la radiación en los labios de las personas. La función de transferencia sirve para aplicar la técnica de codificación predictiva lineal, la cual es importante para estimar los parámetros básicos de la voz, tales como la frecuencia fundamental o pitch, formantes, espectro y funciones de área del tracto vocal.
>
> Adicionalmente, se realizó la programación en Matlab de la solución de las ecuaciones de la codificación predictiva lineal para el modelo de voz que utiliza las ecuaciones obtenidas de un modelo basado en tubos especialmente diseñados para que puedan tener un comportamiento dinámico, similar a la de la voz humana.

Finalmente, se muestran resultados comparativos entre una muestra de voz de un hombre adulto y su correspondiente luego de ser codificada y sintetizada por la técnica de codificación predictiva lineal.

Espinoza (2012) en su informe técnico denominado "Comprensión de voz - Códec G. 729", indica que:

El presente informe pretende dar a entender el funcionamiento del codificador de voz basado en la recomendación G.729 de la ITU-T, "Codificación de la voz a 8 kbit/s mediante Predicción Lineal por Excitación Algebraica de Código de Estructura Conjugada - CSACELP". Este codificador está diseñado para trabajar con señales digitales y se basa en una variante del modelo de codificación por Predicción Lineal por Excitación de Código (CELP), que a su vez nace de la necesidad de optimizar el modelo de Codificación por Predicción Lineal (LPC) basado en un modelo muy simplificado de producción sintetizada de la voz. Opera con tramas vocales de 10 ms correspondientes a 80 muestras a una velocidad de muestreo de 8000 muestras por segundo. En cada trama de 10 ms se analiza la señal vocal para extraer sólo parámetros como coeficientes del filtro de predicción lineal, ganancias e índices de las tablas de códigos adaptativos y fijos. Estos parámetros e índices se empaquetan y se transmiten. En el decodificador, se reciben estos parámetros e índices para recuperar los parámetros de excitación y del filtro de síntesis.

En este informe revisaremos las definiciones previas de los principales conceptos para proceder con la descripción de las características más. importantes de la estructura codificador G.729 y así alcanzar el objetivo de entender su funcionamiento.

Argandoña (2008), en la tesis de Maestría denominada "Implementación

de un codificador de voz CELP mejorado para canales de banda angosta", manifiesta que:

> El presente trabajo de tesis describe el estudio e implementación en hardware (DSP) de un codificador de voz basado en el estándar CELP FS1016 para canales de banda angosta. En primer lugar, se realiza un breve estudio del marco teórico en el que se basan los codificadores de voz, luego se describen las técnicas de procesamiento de voz más importantes que conforman el presente trabajo, en especial las referidas al estándar FS1016, se describen también los detalles de la implementación en el DSP, y finalmente se presentan las pruebas de calidad respectivas.
>
> En base al estudio y correcto conocimiento de todos los bloques constituyentes del esquema CELP FS1016, se proponen mejoras y optimizaciones tanto en el ámbito algorítmico, como en el ámbito del procesador DSP. Con ello se pretende obtener una mayor calidad de voz decodificada (sintética), sin comprometer seriamente la tasa de compresión del sistema ni el tiempo de ejecución del algoritmo, de tal forma que pueda ser empleado en sistemas de transmisión digital de datos de banda angosta.
>
> En el presente trabajo se usaron varias técnicas que pertenecen al esquema de codificación de voz basado en la Predicción Lineal con Excitación de Código (Code Excited Linear Prediction) como son: Cuantización Escalar de los Coeficientes LPC usando la técnica LSP (Line Spectral Frequency), Interpolación de los LSP, Análisis por Síntesis basado en la minimización del error ponderado perceptualmente (Perceptual Weighting Error), Excitación basada en un Codebook Estocástico (Excitación Vectorial) más un Codebook Adaptivo (Pitch), Post-Filtro con control automático de ganancia, etc.
>
> La implementación en hardware constituye un aporte importante del presente trabajo. El codificador es implementado sobre la tarjeta de desarrollo DSK TMS320C6711 de Texas Instruments, lo cual permitirá su fácil incorporación a sistemas de comunicaciones banda angosta, a través

de interfaces como RS232, RS485, etc.

Finalmente se evalúa la calidad de la voz sintética del codificador tanto sin inserción como con inserción de errores (BER variable) en la señal codificada.

Argandoña (2008) en su tesis de título denominada "Simulación para plataforma Xilinx (FPGA) e implementación hardware (DSP texas instruments tms320C6711) en punto fijo y punto flotante de un codificador de voz LPC-10 para bajas tasas de bits", indica que:

El presente trabajo de tesis describe el estudio, simulación (para plataforma FPGA Xilinx) e implementación en hardware (DSP) en dos versiones: punto fijo de 16 bits y punto flotante, de un codificador de voz basado en el esquema LPC-10 para canales de banda angosta.

En primer lugar, se realiza un estudio del marco teórico en el que se basan los codificadores de voz, luego se describen las técnicas de procesamiento de voz más importantes que conforman el presente trabajo, destacando entre ellas el algoritmo SIFT para cálculo óptimo de la frecuencia fundamental de la voz.

Seguidamente se describen los detalles de la simulación y de la implementación en DSP en las dos versiones (punto fijo y punto flotante).

En base al estudio y correcto conocimiento de todos los bloques constituyentes del esquema LPC-10, se proponen mejoras (en el algoritmo SIFT) y optimizaciones tanto en el ámbito algorítmico, como en el ámbito del procesador DSP. Con ello se pretende obtener una mayor calidad de voz decodificada (sintética), sin comprometer seriamente la tasa de compresión del sistema ni el tiempo de ejecución del algoritmo, de tal forma que pueda ser empleado en sistemas de transmisión digital de datos de banda angosta.

La implementación en hardware constituye un aporte importante del

presente trabajo. El codificador es implementado sobre la tarjeta de desarrollo DSK TMS320C6711 de Texas lnstruments en las dos versiones (punto fijo y punto flotante). Finalmente se evalúa la calidad de la voz sintética del codificador mediante pruebas subjetivas (Test MOS).

Diaz (2005), en su tesis de título denominado "Desarrollo de un Sistema Seguro de Transmisión de Voz mediante Internet", indica que:

> El sistema proporciona en base a diferentes elementos criptográficos, alternativas para la seguridad de las comunicaciones de voz en Internet. Asimismo, con el fin de que la voz digitalizada viaje de manera segura en Internet, se han seleccionado algoritmos simétricos TripleDES, IDEA, Blowfish, AES, CAST5 que sean resistentes a ataques criptográficos, y dos protocolos de comunicación para el intercambio de la clave: uno diseñado en base a la criptografía de clave pública RSA y el otro, en el SSL (Secure Socket Layer).
>
> El desarrollo de este sistema ha tomado en cuenta las consideraciones en cuanto al ancho de banda y a la calidad de servicio en las comunicaciones de voz. Para esto, se ha seleccionado el uso de codificadores de voz como G723 y GSM, los cuales disminuyen el uso del ancho de banda y utilizan detectores de actividad de la voz para incorporar tramas de silencio.
>
> Con esto, se consiguió que al tener menor cantidad de información que viaje por Internet, exista menos cantidad de paquetes perdidos. Asimismo, para la transmisión de voz, se emplea el protocolo de tiempo real (RTP, Real Time Protocol. Del mismo modo, dentro de este sistema se ha considerado el uso del protocolo de control de tiempo real (RTCP, Real Time Control Protocol), pues éste permite monitorear la calidad de servicio percibido por el receptor, y almacenar la información de los usuarios de las distintas sesiones establecidas.
>
> La implementación del sistema propuesto ha sido desarrollada en Java,

> utilizando: el Java Media Framework (JMF) como Interfase de Programación de Aplicaciones multimedia API (Application Programming Interface) desarrollada por Sun.

Pozo (2005), en su informe técnico denominado "Comprensión de la voz aplicando la técnica de predicción lineal de coeficientes", hace las siguientes precisiones:

> Este trabajo trata sobre el método de compresión de voz aplicando la técnica de predicción lineal de coeficientes.
> En la introducción se realiza un análisis de como a través de un estudio de la forma en que se pronuncia la voz, se puede hacer un modelo matemático y de esta forma tratar de comprimir la señal de voz.
> En el primer capítulo es una introducción teórica de los distintos métodos en la que se puede representar el modelo matemático (coeficientes del filtro LPC) de la voz.
> En el segundo capítulo se desarrolla las distintas técnicas para hallar los coeficientes del LPC.
> En tercer capítulo se ha desarrollado en MATLAB, un algoritmo en el cual, a partir de la captura de una frase, siguiendo la técnica del LPC, se demuestra que se puede comprimir la voz, y después recuperarla.
> En las conclusiones se muestra las observaciones que uno ha realizado al correr el algoritmo y las limitaciones que se tiene para poder desarrollar un algoritmo más elaborado.
> En las recomendaciones se da ideas que uno tiene acerca de cómo está la Electrónica actualmente en las Telecomunicaciones.

2.2 Bases teóricas

2.2.1 La voz

Según Hernández (2005) “la voz o el lenguaje hablado es la forma más común y natural de comunicación del ser humano. Su tratamiento, el procesamiento y la transmisión, ha representado una de las tareas más interesantes en el análisis de señales”

Las investigaciones sobre el procesamiento de voz han permitido el desarrollo de innovación tecnológica al servicio de la sociedad y del ser humano.

El soporte fundamental de la voz es el sonido. El sonido se caracteriza por fluctuaciones de presión en un medio compresible.

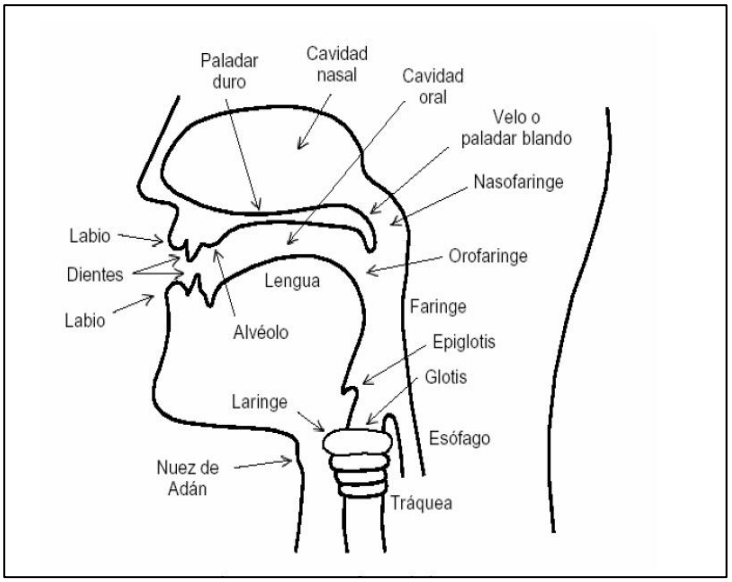
Según Hernández (2005) “dos cosas deben existir a fin de que se produzca una onda sonora: una fuente mecánica de vibración y un medio elástico a través del cual pueda propagar la perturbación”

Para Velásquez (2008), la voz humana es una onda de presión que se genera por medio del aparato fonatorio. Está conformado por los pulmones considerada la fuente de energía, en forma de flujo de aire, la laringe que contiene las cuerdas vocales, la faringe, las cavidades oral y nasal y una serie de elementos articulatorios.

En la Figura 1, se muestra el camino que debe seguir el aire exhalado desde los pulmones hasta el exterior a través de los diferentes conductos.

Figura 1

Aparato fonatorio humano



Fuente. Recuperado de Hernández (2005)

2.2.1.1 Propiedades de las señales de voz

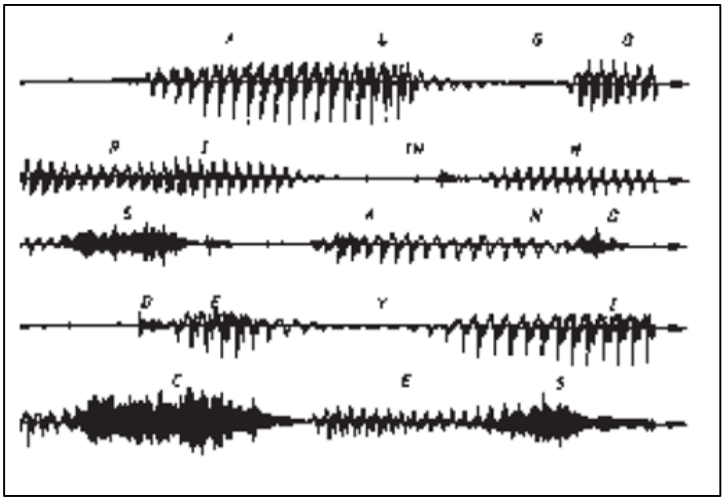
La voz se representa por señales no estacionarias y con variaciones lentas en el dominio del tiempo y se procesan en intervalos muy cortos, entre 5 y 30 ms. Según Proakis y Manolakis (2007) "un segmento de voz se puede representar con un alto grado de precisión como la suma de varias señales sinusoidales de diferentes amplitudes y frecuencias"

$$\sum_{i=1}^{N} A_i(t)\,\mathrm{sen}[2\pi F_i(t)t + \theta_i(t)]$$

Es decir, donde {Ai(t)}, {Fi(t)} y {θi(t)} son los conjuntos de amplitudes, frecuencias y fases (posiblemente variables con el tiempo), respectivamente, de las señales sinusoidales, como se puede ver en la figura 2.

Figura 2

Ejemplo de una señal de voz



Fuente. Recuperado de Proakis y Manolakis (2007)

Según Hernández (2005), cuando se refiera a las características de los

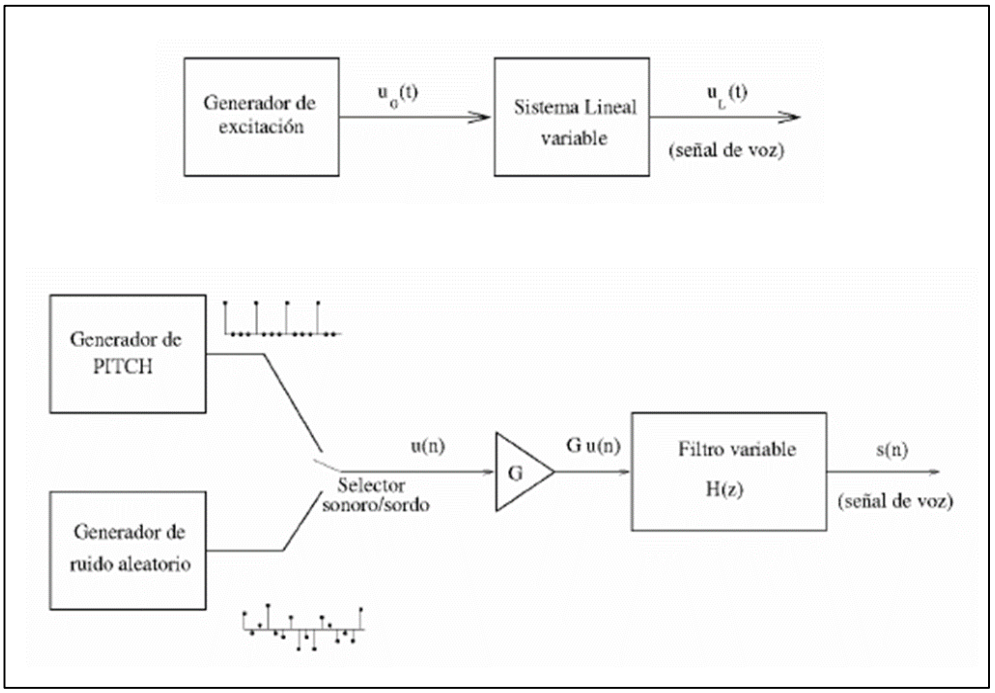
sonidos indica que:

> En el dominio del tiempo los sonidos sonoros tienen una naturaleza cuasi periódica y en el dominio de la frecuencia una estructura armónica fina. Los sonidos no sonoros presentan una estructura típica aleatoria, que se manifiestan en el dominio del tiempo sin periodicidades marcadas y un espectro mucho más compensado en frecuencia (Hernández, 2005, p. 23).

Un sistema de producción de voz se puede modelar de manera sencilla con un modelo de sistema-fuente, que es un sistema lineal alimentado por una fuente de excitación. En una aproximación de segundo orden, la excitación consiste en un tren de pulsos para los sonidos sonoros y ruido blanco para los no sonoros, tal como se muestra en la figura 3.

Figura 3

Diagrama de bloques del modelo digital de producción de voz



Fuente. Recuperado de De la Torre (2001)

2.2.1.2 Clasificación de los sonidos de la voz

Los sonidos emitidos por el aparato fonatorio pueden clasificarse de acuerdo con diversos criterios que tienen en cuenta los diferentes aspectos del fenómeno de emisión. Según Torres (2005, p.8), estos criterios son:

a) Según su carácter vocálico o consonántico.
b) Según su oralidad o nasalidad
c) Según su carácter tonal (sonoro) o no tonal (sordo)
d) Según el lugar de articulación
e) Según el modo de articulación
f) Según la posición de los órganos articulatorios
g) Según la duración

2.2.2 Clasificación de los sonidos de la voz

2.2.2.1 Las señales de voz sonoras

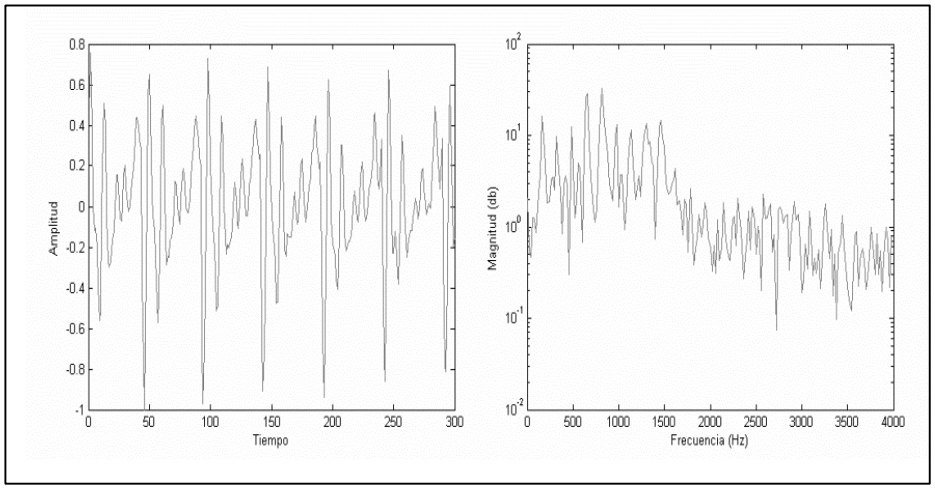
Según Hernández (2005), cada segmento o trama puede ser clasificado como sonoro e indica que:

> **Las señales de voz sonoras** tienen una naturaleza cuasi periódica en el dominio del tiempo y una estructura armónica fina en el dominio de la frecuencia, provocada por la vibración de las cuerdas vocales. Además, su espectro decae hacia altas frecuencias. Su energía es alta debido a que el aire encuentra poca obstrucción al pasar por el tracto vocal. La figura 4 muestra un tramo de señal de voz sonoro en el tiempo y la frecuencia.
> Estos sonidos consisten en una frecuencia fundamental (frecuencia de pitch) y un conjunto de componentes armónicos de la misma, producidos por las cuerdas vocales. El tracto vocal modifica la señal de excitación

provocando frecuencias formantes (ceros) (p.3).

Figura 4

Tramo de señal de voz sonoro en el tiempo y la frecuencia



Fuente. Recuperado de Hernández (2005)

Formantes

Según Velásquez (2008), al referirse de los formantes como las frecuencias de resonancia del espectro, manifiesta que:

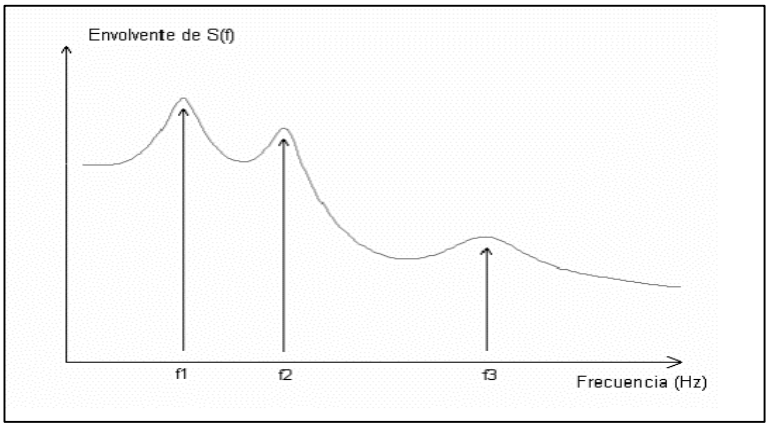
> Son elementos que sirven para distinguir componentes del habla humana, principalmente, las vocales y sonidos sonantes. El formante con la frecuencia más baja se llama F1, el segundo F2, el tercero F3, etc.
>
> Normalmente sólo los dos primeros son necesarios para caracterizar una vocal, aunque la pueden caracterizar más formantes. Generalmente, los formantes posteriores determinan propiedades acústicas como el timbre (p.13).

Cabe indicar que según Aritz Dallo (2012), al referirse de los formantes, manifiesta que (ver figura 5):

> Los sonidos sonoros consisten en una frecuencia fundamental (frecuencia de pitch) y una serie de componentes armónicos de la misma, producidos por las cuerdas vocales. El espectro de la señal de voz varía con el tiempo debido a las variaciones en la forma y en la posición del tracto vocal. Los formantes son las frecuencias de resonancia del espectro, es decir, los picos de la envolvente del espectro de la señal de voz que representan las frecuencias de resonancia del tracto vocal. En señales de voz esas frecuencias dependen del tamaño y de la forma del tracto vocal. Así pues, los formantes caracterizan a un sonido frente a los demás, y son los que nos permiten distinguir a las personas. En un fonema, los más importantes son los 3-4 primeros formantes, que son los que tienen la mayor parte de energía (p. 15).

Figura 5

Formantes de la señal de voz en el espectro



Fuente. Recuperado de Aritz Dallo (2012)

2.2.2.2 Las señales de voz no sonoras

Según Hernández (2005), un segmento o trama puede ser definido como no sonora:

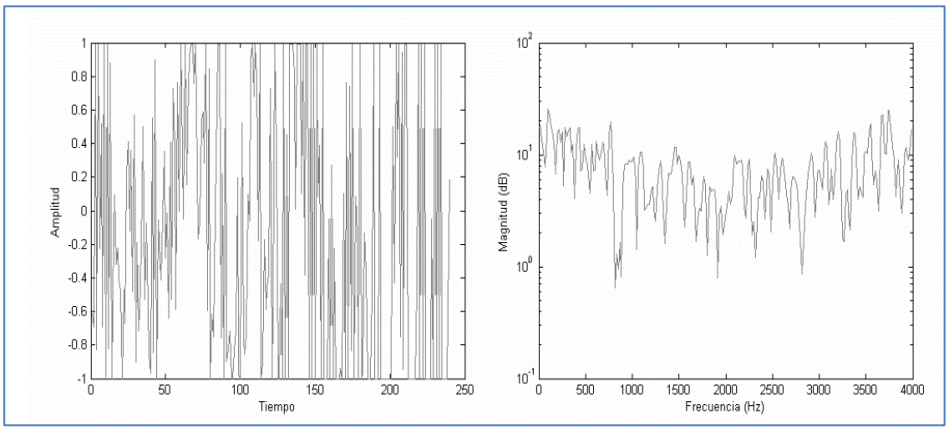
Las señales de voz no sonoras tienen una estructura típica aleatoria, sin periodicidades marcadas en el dominio del tiempo y un espectro mucho más compensado en frecuencia (tiene un espectro de banda ancha). Su energía es mucho menor debido a la presencia de obstrucciones en el tracto vocal.

Un segmento que no es consecuencia de la actividad vocal se denomina como silencio. En telefonía, aproximadamente el 50 % del tiempo de conversación es silencio (p.3).

La figura 6 muestra un tramo de señal de voz sonoro en el tiempo y la frecuencia.

Figura 6

Tramo de señal de voz no sonoro en el tiempo y la frecuencia



Fuente. Recuperado de Hernández (2005)

.

2.2.2.3 Dispersión no selectiva

Este tipo de dispersión se origina cuando todas las longitudes de onda son dispersadas más o menos con la misma intensidad, esto es, el factor de atenuación que sufre el haz de luz transmitido es el mismo independientemente de la frecuencia

óptica que lo caracterice.

Nuevamente, la magnitud de tal atenuación viene directamente relacionada con la visibilidad, para los casos de niebla por advección y niebla por convección; y por la intensidad de lluvia o nieve en el caso en el que se produzcan tales fenómenos meteorológicos.

2.2.3 Caracterización de los sonidos de voz

Uno de los mayores problemas encontrados en los trabajos de investigación sobre el estudio del habla ha sido el de la variabilidad en ésta. Varios investigadores coinciden al manifestar que diferentes aspectos del estado físico y emocional del locutor, que incluyen la apariencia, el sexo, la personalidad, la edad y la inteligencia personalidad, solamente pueden identificarse mediante la voz. Todos estos factores contribuyen a la variabilidad del habla, que son diferentes para cada interlocutor.

2.2.3.1 La intensidad

Según Duque y Morales (2007) cuando hace referencia a este parámetro, indica que:

> **La intensidad:** Es equivalente al volumen. El aire al salir de los pulmones golpea la glotis y produce vibraciones. Cuanto más amplias sean, mayor será la fuerza. Se mide en decibelios (dB) y para tener una referencia, una conversación normal ronda entre los 50 dB. Tiene efectos en el oyente porque transmite emociones. Un volumen de voz alto se asocia a la agresividad, nerviosismo, tensión y lejanía. Al contrario, un volumen bajo puede sugerir depresión, cansancio y proximidad (p. 12).

Según Torres (2006) con referencia a este parámetro, manifiesta lo siguiente:

> **La intensidad** es la fuerza con que se emite un sonido dependiendo del esfuerzo muscular o, en general, de la fuerza que haya desencadenado las ondas; se mide en decibelios. Las intensidades acústicas poseen valores muy bajos. Para la máxima intensidad audible el nivel de intensidad es de 120 dB; para la mínima intensidad audible, el nivel sería 0 dB. Un sonido producido por el murmullo de las hojas es de 10 dB; el de una conversación normal 60 dB, el de una calle con tráfico 80 dB; el de un reactor superior a 100 dB. En las proximidades a 120 dB se siente dolor (p. 24).

2.2.3.2 El tono o frecuencia

. Según Duque y Morales (2007) cuando hace referencia a este parámetro, indica que:

> **El tono:** Está relacionado con la cantidad de vibraciones que posee una onda de sonido. A mayor número más aguda será la voz. Estas vibraciones se producen en el ser humano en la laringe y se miden en Hertzios o Hertz (Hz). Las voces masculinas oscilan entre los 75 Hz y los 200 Hz y las voces femeninas entre los 150 Hz y los 300 Hz. Ver la figura 6 (p. 11).

Según Torres (2006) con referencia a este parámetro, manifiesta lo siguiente:

> **Frecuencia (tono)** el número de vibraciones por segundo, es decir, las veces que se repite en un segundo el ciclo completo de la onda. A mayor número de vibraciones, el sonido será más agudo, (chillón); a menos, grave, (ronco). El oído humano es sensible a los sonidos en un margen de frecuencias entre 20 y 20.000 vibraciones por segundo. Por debajo son infrasonidos y por encima ultrasonidos. El campo de audibilidad o zona de frecuencia de los sonidos audibles varía según los sujetos, si bien se admite generalmente que cubre desde las 16 vibraciones hasta las 20 mil. Las ondas mecánicas de frecuencia inferior a las 16 vibraciones se llaman infrasonidos, y las superiores a 20.000,

ultrasonidos. La frecuencia de los sonidos de una conversación normal se sitúa entre 512 y 1.624 vibraciones por segundo (p. 23).

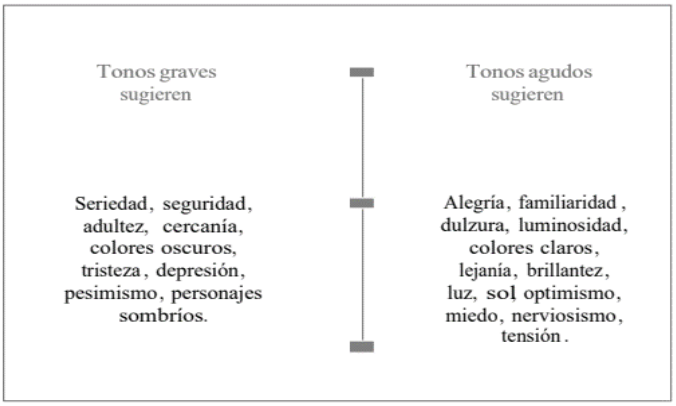
2.2.3.3 El timbre

Según Duque y Morales (2007) cuando hace referencia a este parámetro, indica que:

> **El timbre:** Es lo que permite que distingamos entre dos sonidos de igual intensidad y tono. El aire que sale de los pulmones, recorre y choca con la laringe, labios, dientes y lengua; tiene peculiaridades únicas dependiendo de la morfología de cada persona. Esta característica es el carnet de identidad de cualquier voz. Aporta mucha información real o imaginaria sobre la edad, la apariencia física e incluso una especie de retrato de la personalidad del hablante (p. 11).

Figura 7

Relación tonos-emociones



Fuente. Recuperado de Duque y Morales (2007)

Según Torres (2006) con referencia a este parámetro, manifiesta lo siguiente:

> **Timbre** es la cualidad de un sonido en virtud de la cual lo podemos distinguir de otro de la misma intensidad y frecuencia, pero producido por otra fuente sonora (otro instrumento).
> El timbre depende del hecho de que, normalmente, una onda sonora va acompañada de otras ondas secundarias o armónicas, que dependen del medio ambiente o de la caja de resonancia (p. 24).

2.2.4 Los efectos de las emociones en el habla

Se lograron identificar muchos de los componentes del habla que se utilizan para expresar emociones, dentro de los cuales se consideran los más importantes:

2.2.4.1 El pitch

Según Duque y Morales (2007), "el pitch es la frecuencia fundamental a la que las cuerdas vocales vibran, también llamada frecuencia fundamental" (p. 13).

Según Aritz Dallo (2012), "el pitch ó frecuencia fundamental es un parámetro importante para algunos algoritmos de codificación de voz. Se puede identificar como la periodicidad de los picos de la amplitud en la forma de onda y la estructura fina del espectro" (p. 14). Las frecuencias de pitch de hombres y mujeres normalmente se encuentran en el rango 50-250 Hz (4-20 ms) y 120-500 Hz (2-8,3 ms), respectivamente.

Según Wainschenker et al. (2003), "históricamente se ha definido al pitch como la frecuencia fundamental de espectro de frecuencias del habla y se lo ha asociado al movimiento que realiza la glotis en la generación del sonido". Desafortunadamente "cualquiera sea la forma en la que se lo defina no se ajustará a la realidad, porque la oscilación glotal es una función cuasi periódica" (Klatt, 1987).

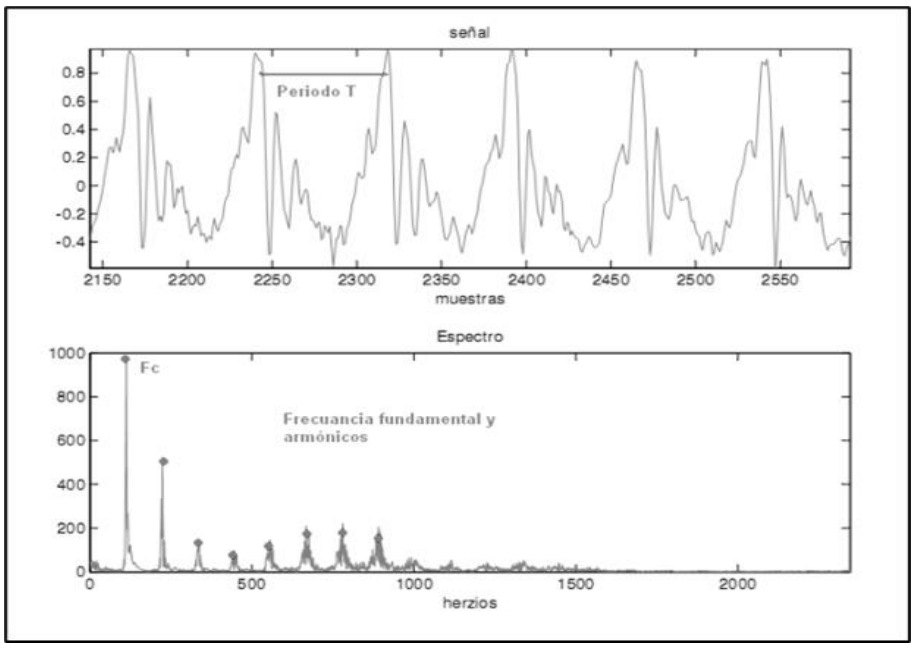
El periodo pitch debe ser estimado en cada trama. Al comparar una trama con muestras pasadas, es posible identificar el periodo en el cual la señal se repite,

resultando en una estimación del periodo pitch actual. Se debe indicar que el procedimiento de estimación tiene efecto solo para tramas sonoras, lo que no sucede con las tramas no sonoras por su naturaleza aleatoria.

El diseño de un algoritmo de estimación del periodo pitch es una tarea compleja por falta de periodicidad perfecta, interferencia con los formantes del tracto vocal, incertidumbre del instante de inicio del segmento sonoro y otros elementos del mundo real como el ruido y el eco.

Figura 8

Señal sonora en tiempo y la magnitud del espectro en escala lineal



Fuente. Recuperado de Aritz Dallo (2012)

Según Duque y Morales (2007), "se considera que las características de la frecuencia fundamental son una de las principales portadoras de la información sobre las emociones". Es por ello que:

- El valor medio del pitch expresa el nivel de excitación del locutor. Una media elevada de F0 indica un mayor grado de excitación.
- El rango del pitch es la distancia entre el valor máximo y mínimo de la

frecuencia fundamental. Refleja también el grado de **exaltación** del locutor. Un rango más extenso que el normal refleja una excitación emocional o psicológica.

- Las fluctuaciones en el pitch descritas como la velocidad de las fluctuaciones entre valores altos y bajos y si son abruptas o suaves son producidas psicológicamente. En general, la curva de tono es discontinua para las emociones consideradas como negativas (miedo, enfado) y es suave para las emociones positivas, por ejemplo: la alegría (p. 13).

2.2.4.2 La calidad de la señal de voz codificada

Según Duque y Morales (2007), la intensidad, las irregularidades en la voz, el cociente entre energías a baja y alta frecuencia, breathiness y la laringerización son algunas de las características que diferencian la calidad de la voz. Indica que:

- **Intensidad:** Está relacionada con la percepción del volumen y se refleja en la amplitud de la forma de onda.
- **Irregularidades vocales:** Abarcan un gran rango de características vocales. El jitter vocal refleja las fluctuaciones de un pulso glotal al siguiente (como se observa en el enfado) o la desaparición de voz en algunas emociones como la pena, en la que el habla se convierte en un simple susurro.
- **El cociente entre energía de alta y baja frecuencia:** Gran cantidad de energía en las frecuencias altas se asocia con agitación (enfado), mientras que baja concentración de energía en las frecuencias altas se relaciona con depresión o calma (pena).
- **Breathiness y laringerización:** reflejan las características del tracto vocal que están más relacionados con la personalización de cada voz. Breathiness describe la generación de ruido respiratorio de forma de que la componente fundamental tiende a ser más fuerte, mientras que las frecuencias altas son reemplazadas por ruido aspiratorio. La laringerización se caracteriza por

> una vibración aperiódica de las cuerdas vocales, con un pulso glotal estrecho y pitch bajo, lo que se traduce en una voz chirriante (pp. 13-14).

La calidad de la señal de voz codificada puede clasificarse, a grandes rasgos, en 4 categorías:

- Calidad BROADCAST para comunicaciones en banda ancha,
- Calidad TOLL o de red para señales analógicas (telefonía),
- Calidad de comunicación (señales degradadas pero naturales e inteligibles)
- Calidad sintética (señales poco naturales pero inteligibles, representadas por los codificadores lineales predictivos (LPC) o vocoders).

2.2.4.3 La duración

Según Duque y Morales (2007), "la duración es la componente de la prosodia descrita por la velocidad del habla y la situación de los acentos, y cuyos efectos son el ritmo y la velocidad" (p. 13).

Así mismo, Duque y Morales (2007) manifiesta que las emociones pueden distinguirse por una serie de parámetros que conciernen a la duración, como son:

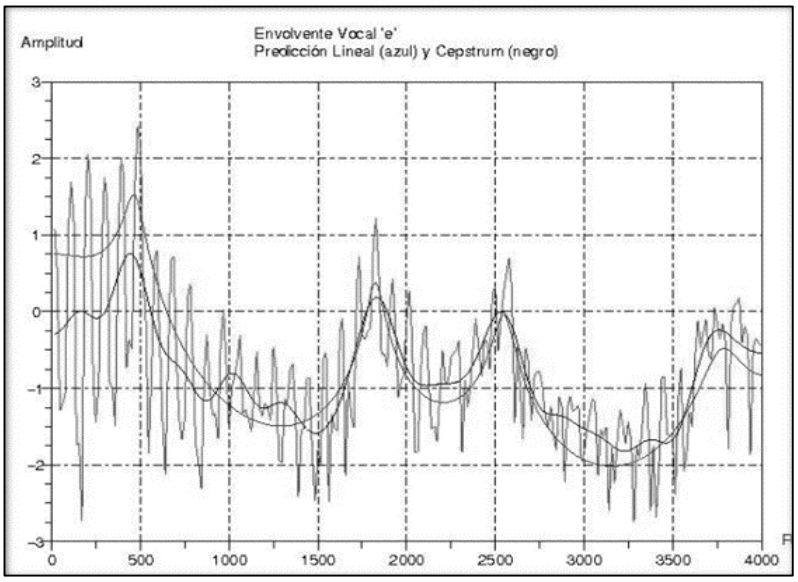
- **Velocidad de locución:** generalmente un locutor en estado de excitación acortará la duración de las sílabas, con lo que la velocidad de locución medida en sílabas por segundo o en palabras por minuto se incrementará.
- **Número de pausas y su duración:** un locutor exaltado tenderá a hablar rápidamente con menos pausas y más cortas, mientras que un locutor deprimido hablará más lentamente, introduciendo pausas más largas.
- **Cociente entre el tiempo de locución y el de pausas** (p. 13).

2.2.5 *Extracción de características de la señal de voz*

En el reconocimiento del habla, la señal de voz preprocesada se ingresa a un nuevo procesamiento para producir una representación de la voz en forma de secuencia de vectores o agrupaciones de valores que se denominan parámetros, que deben representar la información contenida en la envolvente del espectro.

Figura 9

La composición espectral de la muestra sonora



Fuente. Recuperado de Rocamora y López (2005)

Existen distintos métodos de análisis para la extracción de características de la señal de voz. En este caso se analiza los dos de mayor importancia para el análisis de la voz:

- Análisis de predicción lineal (LPC)
- Análisis cepstral

2.2.5.1 Análisis de predicción lineal (LPC)

Se trata de una de las técnicas más potentes de análisis de voz, y uno de los métodos más útiles para codificar voz con buena calidad.

Según Velásquez (2008), "su función es representar la envolvente espectral de una señal digital de voz en una forma comprimida, utilizando la información de un modelo lineal, con lo cual se proporcionan unas aproximaciones a los parámetros de la voz muy precisas".

Además, Velásquez (2008) con referencia sobre la técnica de predicción lineal indica que:

> Se fundamenta en establecer un modelo de filtro de tipo todo polo, para la fuente de sonido. La principal motivación del modelo todo polo viene dada porque permite describir la función de transferencia de un tubo, que sin perdidas está formado por diferentes secciones.
>
> El modelo recibe este nombre porque pretende extrapolar el valor de la siguiente muestra de voz s(n) como la suma ponderada de muestras pasadas s(n-1), s(n-2),…, s(n-K):

$$s(n) \approx -\sum_{k=1}^{p} \alpha_k s(n-k)$$

> Incluyendo un término de excitación *Gu(n)*, la ecuación puede escribirse como una igualdad:

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

> Siendo α_k los denominados coeficientes de predicción lineal (LPC), y G, la ganancia de excitación. Por otro lado, en el dominio Z la ecuación puede escribirse como:

$$S(z) = -\sum_{k=1}^{p} \alpha_k z^{-k} S(z) + GU(z)$$

Lo que conduce a una función de transferencia del tipo todo polo

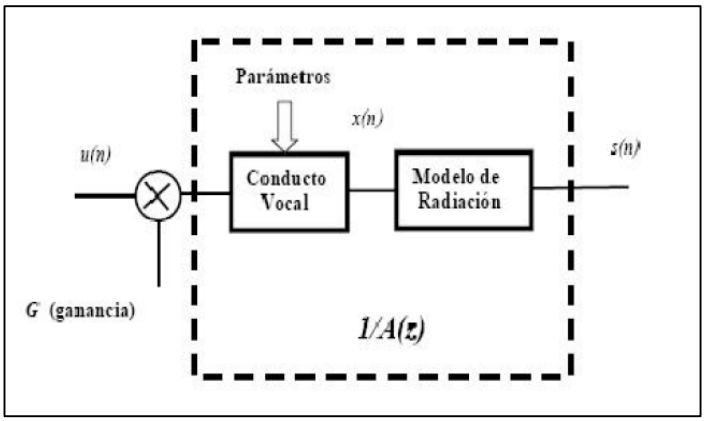
$$H(z) = \frac{S(z)}{GU(z)} = \frac{1}{1 + \sum_{k=1}^{p} \alpha_k z^{-k}} = \frac{1}{A(z)},$$

H(z) representa la función transferencia de un modelo lineal del conducto vocal + radiación. Los parámetros del filtro digital *H(z)* son controlados por la señal de voz que está siendo producida y los coeficientes de este filtro son los LPC (p. 39).

Una interpretación de esta ecuación, que es una versión simplificada, está dada en la figura 10.

Figura 10

Modelo de producción de voz basado en LPC



Fuente. Recuperado de Velásquez (2008),

2.2.5.2 Estimación de los LPC

Según Velásquez (2008, p. 41) con referencia a la estimación de los

coeficientes LPC, expresa que:

Una estima (o predicción) de s(n) basada en p muestras anteriores, puede calcularse como

$$\hat{s}(n) = -\sum_{k=1}^{p} \alpha_k s(n-k)$$

y el error de estimación (predicción) puede entonces definirse como:

$$\varepsilon(n) = s(n) - \hat{s}(n),$$

resultando el error de predicción:

Los LPC se obtienen minimizando un criterio cuadrático en los errores de predicción, para cada cuadro en que es dividido el segmento de voz. Suponiendo que en cada cuadro hay $m+1 >> p$ muestras, y definiendo lo siguiente:

$$\alpha = [\ \alpha_1 \quad \alpha_2 \quad \dots \quad \alpha_p]^T$$

$$\Phi^T(n) = [-s(n-1) \quad -s(n-2) \dots \quad -s(n-p)]$$

la ecuación

$$\hat{s}(n) = -\sum_{k=1}^{p} \alpha_k s(n-k)$$

puede escribirse matricialmente como,

$$\begin{bmatrix} s(n) \\ s(n+1) \\ \vdots \\ s(n+m) \end{bmatrix} = \begin{bmatrix} -s(n-1) & \cdots & -s(n-p) \\ -s(n) & \cdots & -s(n-p+1) \\ \vdots & \vdots & \vdots \\ -s(n+m-1) & \cdots & -s(n+m-1-p) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{bmatrix}$$

Donde el vector α son los coeficientes; también se puede escribir lo anterior como,

$$S_m(n) = \Phi^T(n)\alpha.$$

2.2.6 *Los codificadores de voz*

La compresión de la voz hace referencia a la representación compacta de las señales de voz. La codificación de la voz se refiere a la representación digital de las señales. Siendo el principal objetivo de la codificación de la voz es reducir el número de bits necesarios para su representación para lograr una alta calidad de percepción en la señal de voz reconstruida a un bajo costo, ambos términos, compresión y codificación de la voz, se pueden usar de modo indistinto.

Los códecs de voz están optimizados para señales de voz. Siendo la voz humana más sencilla de modelar que el audio genérico, entonces la voz de alta calidad se puede codificar con métodos específicos con mejor ventaja que el audio genérico.

Dado que en la actualidad existen una variedad de aplicaciones sobre procesamiento de la voz, puede resultar complicado elegir el codificador de voz que mejor se adapte a cada una de ellas. Aunque la mejor solución sería tener un codificador que se pudiera utilizar para cualquier tipo de aplicación, por lo general resulta más económico adaptar el codificador a la aplicación.

2.2.6.1 Propiedades de los codificadores de voz

La calidad de la voz generada a partir de un codificador está en función de la tasa de bit, la complejidad, el retraso y el ancho de banda del mismo; factores a tener muy presentes a la hora de confeccionar cualquier codificador de voz.

Otros factores que se toman en cuenta en la elección de un codificador de voz son su disponibilidad, las condiciones de la licencia y las especificaciones del estándar, ya que algunos sólo se describen como un algoritmo, mientras que otros tienen un código completo.

Tasa de bit

Desde el momento en que los codificadores de voz comenzaron a compartir el canal con otro tipo de información, se hizo necesaria la utilización de la menor tasa de bit posible para no usar una parte excesiva del canal.

Una solución bastante común es usar una tasa de bit fija cuando hay actividad vocal y una tasa baja para el ruido de fondo.

Retraso

Uno de los aspectos de diseño más importantes a la hora de implementar la voz es minimizar el retraso de extremo a extremo. También se le conoce como latencia del códec. Valores máximos de hasta 400 ms pueden ser admisibles si no hay ecos, aunque es preferible que este retraso esté menor de los 200 ms.

- **Retraso algorítmico**: Muchos de los codificadores de voz con una tasa de bit baja procesan las tramas una a una. Los parámetros de la señal son actualizados y transmitidos para cada trama. Esto significa que antes de analizar la señal de voz, es necesario almacenar una serie de información, esto produce el retardo.

- **Retraso debido al procesamiento**: Tiempo que emplea el codificador en analizar la señal de voz y el decodificador en reconstruirla.

- **Retraso de la comunicación**: Tiempo que necesita una trama de información para transmitirse desde el codificador al decodificador.

Complejidad

Desde la perspectiva del diseño una mayor complejidad conlleva un mayor costo y una mayor necesidad de consumo de potencia. Para aplicaciones portátiles, un mayor consumo de potencia implica la reducción del tiempo entre recargas. Sus

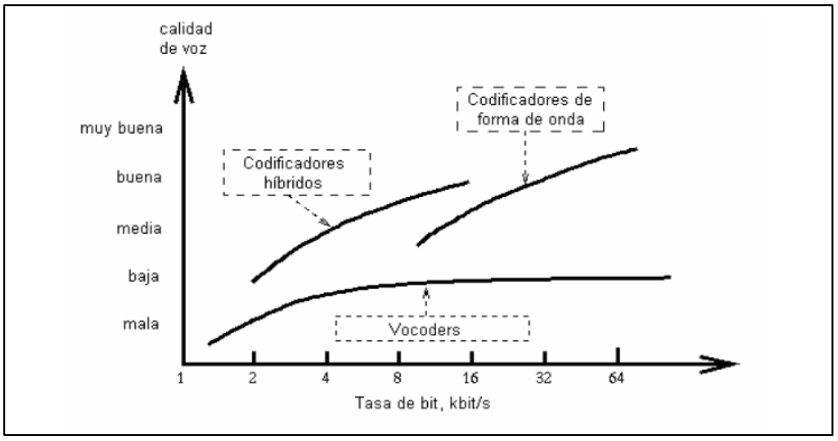
características se pueden medir en función de la velocidad computacional, medida en millones de instrucciones por segundo (MIPS).

Calidad

Un factor subjetivo, pero de mucha importancia es la percepción de cuán bien se escucha la señal codificada en presencia de condiciones ideales. En la figura 5 se observa la relación entre la tasa de bit y la calidad de una señal de voz codificada y sin ruido de fondo para los 3 tipos de codificadores de voz más usados.

Figura 11

Relación tasa de bits-calidad para diferentes codificadores



Fuente. Recuperado de Hernández (2005)

Cada códec proporciona una cierta calidad de voz. La calidad de la voz transmitida es una respuesta subjetiva del oyente. Una medida común de referencia para determinar la calidad del sonido producido por códecs específicos es la Mean Opinión Store (MOS).

Tabla 1

Medida subjetiva de la calidad de la voz (MOS)

MOS	Calidad	Clasificación de la voz
1	Mala	
2	Pobre	
3	Media	Voz artificial Comunicaciones
4	Buena	Red digital mejorada
5	Excelente	Transparente

Fuente. Recuperado de Hernández (2005)

2.2.7 *Tipos de algoritmos de codificación de voz*

Los distintos algoritmos de codificación tratan de eliminar la redundancia de la señal y así poder reducir al mínimo el número de bits usados para codificar cada muestra. Un método de codificación de voz se evalúa según las siguientes características:

- Velocidad de transmisión ("bit rate")
- Calidad de la voz reconstruida.
- Complejidad de la implementación
- Retardo introducido
- Robustez ante la aparición de errores en el canal o interferencias acústicas.

Los codificadores de voz se pueden encuadrar dentro de tres grandes categorías (ver figura 6):

- Codificadores de forma de onda.
- Codificadores de voz (vocoders).
- Codificadores híbridos.

Figura 12

Clasificación de los codificadores más importantes

Tipo	*Algoritmo de codificación*
Codificadores de forma de onda	PCM (Pulse-Code Modulation), APCM (Adaptive PCM) DPCM (Differential PCM), ADPCM (Adaptive DPCM) DM (Delta Modulation), ADM (Adaptive DM) CVSD (Continuously Variable-Slope DM) APC (Adaptive Predictive Coding) SBC (Subband Coding) ATC (Adaptive Transform Coding)
Codificadores híbridos	MPLP (Multipulse-Excited Linear Prediction) RPE (Regular Pulse-Excited linear prediction) RELP (Residual-Excited Linear Prediction) VSELP (Vector-Sum Excited Linear Prediction) CELP (Code-Excited Linear Prediction) ACELP (Algebraic CELP) CS-ACELP (Conjugated Structure ACELP)
Vocoders	Canal, Formante, Fase, Cepstral o Homomórfico LPC (Linear Predictive Coding) MELP (Mixed-Excitation Linear Prediction) STC (Sinusoidal Transform Coding) MBE (Multiband Excitation), MBE mejorada

Fuente. Recuperado de Hernández (2005)

2.2.8 *Codificadores de voz (vocoders)*

Los vocoders (Voice Coders) intentan generar una señal de voz que suene igual que la original, independientemente de si la forma de onda se parece o no. En el emisor se analizan la señal de voz y se extraen los parámetros del modelo y la excitación. Estos parámetros son cuantizados y transmitidos al receptor, donde la señal de voz se reconstruye en base a ellos. Por esta razón, a los vocoders también se les llama codificadores paramétricos.

Los vocoders pueden, por norma general, conseguir una mayor compresión de la voz que los codificadores de forma de onda, sin embargo, se les reconoce por la calidad artificial o innatural de la voz que generan, excepto por las recientes mejoras efectuadas en algunos, como por ejemplo el MELP.

2.2.8.1 Vocoder de predicción lineal (LPC)

El vocoder más utilizado es el de predicción lineal LPC (Linear Predictive Code), que supone que cada muestra puede obtenerse a partir de una combinación lineal de las anteriores, aceptándose un filtro todo polo para modelar el tracto vocal. La expresión para realizar la decodificación sería la siguiente:

$$s_n = \sum_{k=1}^{p} a_k s_{n-k}$$

donde s_n es la muestra actual, s_{n-k} son las muestras precedentes, a_k son los coeficientes del filtro, que se calculan para minimizar el error de la muestra actual y su predicción, y p es el orden del filtro.

Para un LPC de orden 10 la tasa de bit ronda los 2.4 kbit/s. Aunque el método da como resultado una señal de voz que suena artificial, es inteligible. Este método tiene extensos usos en aplicaciones militares, donde una calidad alta de la voz no es tan importante como una tasa baja de bit para permitir una fuerte encriptación de la información.

A continuación, se presenta la estructura de un codificador de voz de predicción lineal:

Encoder

La figura 13, muestra el diagrama de bloques del codificador. La voz de entrada se segmenta primero en cuadros no superpuestos. Se utiliza un filtro de pre-acentuación para ajustar el espectro de la señal de entrada.

El detector de voz clasifica el cuadro actual como sonoro o no sonoro y emite un bit que indica el estado de sonorización.

La señal pre-enfatizada se usa para el análisis de LP, donde se derivan los diez coeficientes LPC.

Estos coeficientes se cuantifican con los índices transmitidos como

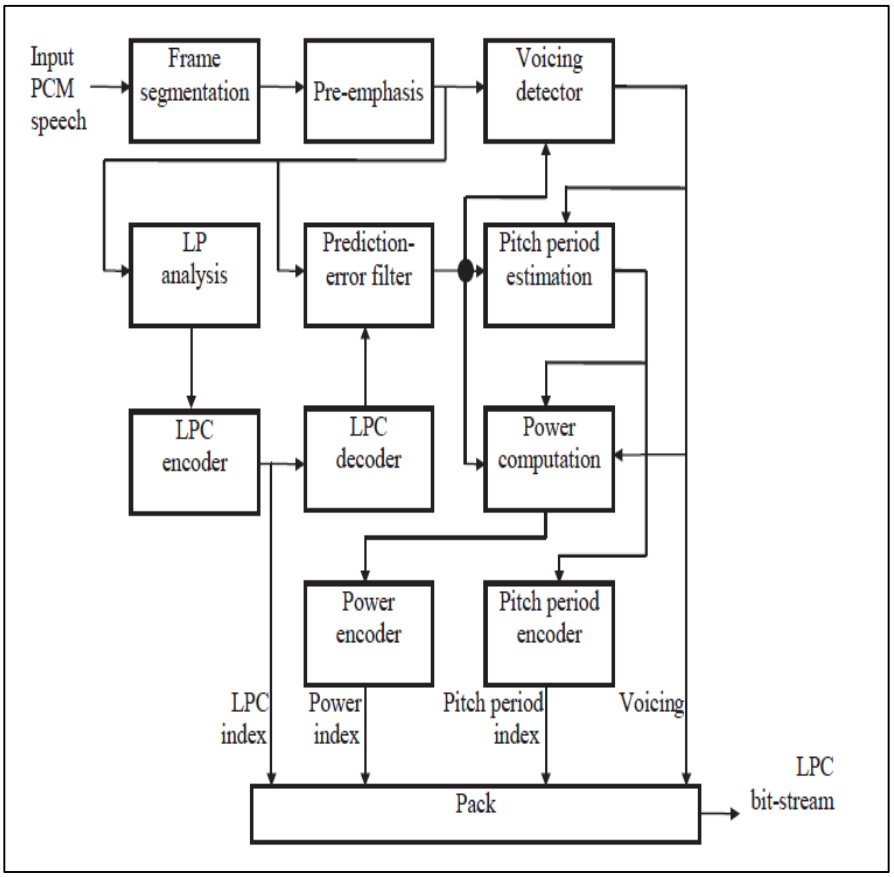
información de la trama. Los coeficientes LPC cuantificados se utilizan para construir el filtro de error de predicción, que filtra la voz pre-acentuada para obtener la señal de error de predicción en su salida (predicción interna).

El período de tono o periodo pitch se estima a partir de la señal de error de predicción si y solo si la trama es sonora.

Al utilizar la señal de error de predicción como entrada al algoritmo de estimación del período de pitch, se puede obtener una estimación más precisa ya que se elimina la estructura formante debido al tracto vocal.

Figura 13

Diagrama de bloques del codificador



Fuente. Recuperado de Chu (2003)

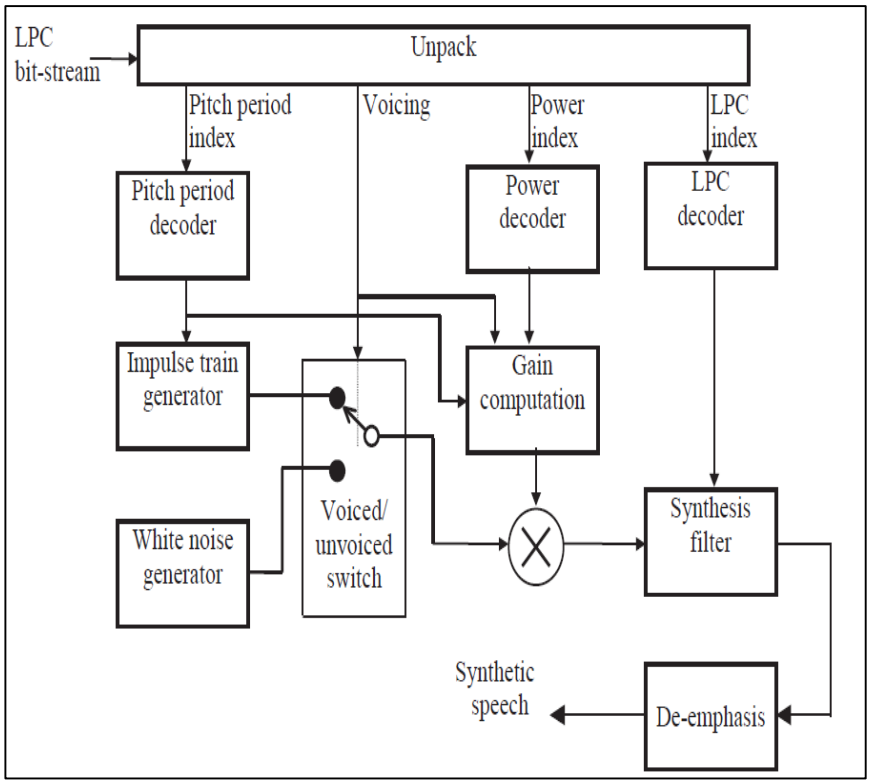
Decoder

La figura 14, muestra el diagrama de bloques del decodificador y es esencialmente el modelo LPC de producción de voz con parámetros controlados por el flujo de bits.

Se supone que la salida del generador del tren de impulsos se compone de una serie de impulsos de amplitud unitaria, mientras que el generador de ruido blanco tiene una salida de variación unitaria.

Figura 14

Diagrama de bloques del decodificador



Fuente. Recuperado de Chu (2003)

El cálculo de ganancia se realiza de la siguiente manera. Para el caso no sonoro, la potencia de la entrada del filtro de síntesis debe ser el mismo que el error de predicción en el lado del codificador. Denotando la ganancia por g, tenemos

$$g= (p)^{1/2}$$

ya que el generador que produce el ruido blanco tiene salida de variación de valor la unidad.

Para el caso no sonoro, la potencia del tren de impulsos que tiene una amplitud de g y un período de T, medido en un intervalo de longitud [N / T] T, debe ser igual a p. De la operación se obtiene

$$g= (T.p)^{1/2}$$

Finalmente, la salida del filtro de síntesis se enfatiza para producir el habla sintética.

2.3 Definición de conceptos

2.3.1 Autocorrelación

La autocorrelación o dependencia secuencial, es una herramienta estadística utilizada frecuentemente en el procesado de señales. La función de autocorrelación se define como la correlación cruzada de la señal consigo misma.

2.3.2 CELP

Técnica de codificación por predicción lineal con excitación, se basa en procedimientos de búsqueda de análisis por síntesis, cuantificación de vectores con pesos (VQ) y predicción lineal (LP).

2.3.3 Método Cepstrum

Es un método para la estimación del pitch en dominio de la frecuencia

mediante el análisis cepstral. Mediante el método se obtiene los coeficientes cepstrales. Para definir un determinado periodo de pitch se encuentra el máximo valor de la secuencia de coeficientes cepstrales. La palabra "cepstrum" se deriva de la palabra inglesa "spectrum" (espectro), para dar una idea del cálculo de la transformada inversa del espectro.

2.3.4 Codificadores paramétricos

Se basan en la obtención de los parámetros de un modelo de producción de señal de voz. Estos codificadores se denominan codificadores de la fuente o vocoders. Intentan producir una señal que suena como original, independientemente de si la forma de onda se parece o no.

2.3.5 ITU (Unión Internacional de Telecomunicaciones)

Organismo internacional de las Naciones Unidas que se encarga de la regulación de los sistemas de telecomunicaciones a nivel mundial.

2.3.6 LPC (Linear Predicitive Coding)

Es la aplicación del análisis de predicción lineal para estimar los parámetros básicos de la voz.

2.3.7 LPC-10

Basado en el estándar FS-1015, este algoritmo de codificación de voz fue desarrollado por el DOD (Department Of Defense), el Departamento de Defensa Estadounidense. Permite la codificación de la señal de la voz a una velocidad de 2400 bps.

2.3.8 MIPS (Millones de Instrucciones Por Segundo)

Es la unidad de medida de la complejidad computacional que debe realizar el sistema para llevar a cabo el proceso de codificación específico.

2.3.9 MOS (Mean Opinion Score)

Representa la calidad subjetiva de un codificador de voz al realizar pruebas comparativas con otro codificador, realizado audiciones de una misma señal codificada. Se evalúa el porcentaje de oyentes que han preferido cada uno de ellos.

Capítulo III: Marco Metodológico

3.1 Hipótesis de la investigación

3.1.1 Hipótesis general

"La parametrización de la señal de voz mediante el uso del modelo de predicción lineal (LP) basados en el estándar FS-1015, permite desarrollar codificadores de voz (vocoders) con una baja tasa de bits".

3.1.2 Hipótesis específicas

"Los principales parámetros del modelo de Predicción Lineal (LP) aplicados a las señales de voz, permitirá el modelamiento de un codificador de voz (vocoder)".

"Los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, permitirán obtener resultados del procesamiento elegido".

"La simulación en la plataforma de desarrollo MATLAB del proceso de compresión de voz basados en el estándar FS-1015, permitirá el desarrollo del codificador de voz".

3.2 Variables

3.2.1 Identificación de la variable independiente

La variable independiente es **el modelo de predicción lineal (LPC)**

3.2.1.1 Indicadores

Dimensión 1: Determinación de los coeficientes de predicción lineal.

Indicadores
Método de la Autocorrelación

Dimensión 2: Determinación de señales sonoras o no sonoras.

Indicadores
Número de cruces por cero
Energía de la señal: log Es

DIMENSIÓN 3: Estimación del pitch.

Indicadores
Método de autocorrelación (ACM)

3.2.1.2 Escala para la medición de la variable

La escala de medición para la variable independiente, está relacionado con la estimación de la duración del pitch y se mide en unidades de tiempo (mseg).

3.2.2 Identificación de la variable dependiente

La variable dependiente es **el codificador de voz (vocoder).**

3.2.2.1 Indicadores

DIMENSIÓN 1: Según sus parámetros

Indicadores
Velocidad de transmisión
Calidad de la voz reconstruida.

DIMENSIÓN 2: Según la técnica de codificación

Indicadores
Retardo producido

3.2.2.2 Escala para la medición de la variable

La escala de medición para la variable dependiente, está relacionado con la tasa de transmisión que desarrolla el codificador de voz y se mide en kbit/seg.

3.3 Tipo y diseño de investigación

Este trabajo de investigación es de tipo investigación aplicada, porque se está utilizando tecnología para el diseño de un codificador paramétrico de voz basado en el estándar FS-1015.
El diseño de investigación es experimental, porque se simula en un laboratorio, que consta con una computadora personal y emplea herramientas informáticas de simulación (MATLAB).

3.4 Nivel de investigación

Se aplica un nivel de investigación descriptiva porque se trata acerca de un aspecto de la realidad (modelo de Predicción Lineal), describiendo su importancia a través de la investigación documental, es decir lo que ha generado el motivo de la investigación (desarrollo de un codificador paramétrico de voz) y trata de explicar y demostrar que el modelo de predicción lineal (LP) basado en el estándar FS-1015 puede aplicarse en el desarrollo de codificadores de voz (vocoder).

3.5 Ámbito y tiempo social de la investigación

Se desarrollará en los laboratorios de la Escuela Profesional de Ingeniería Electrónica de la Universidad Privada de Tacna, ciudad de Tacna, en el periodo enero a julio del 2020.

3.6 Población y muestra

3.6.1 Unidad de estudio

La unidad de estudio comprende los codificadores que utilizan la compresión de la voz.

3.6.2 Población

No corresponde por tratarse de una investigación aplicada.

3.6.3 Muestra

No corresponde por tratarse de una investigación aplicada.

3.7 Técnicas e instrumentos para la recolección de datos

3.7.1 Técnicas de recolección de los datos

Se empleó la técnica de análisis documental para la recolección de datos. Está técnica hace uso de los documentos confiables, tales como libros de consultas, tesis de investigación, artículos científicos, libros de bibliotecas virtuales ya existentes. Así como de fuentes similares de información como las fuentes de datos, tales como diseño y simulación de codificadores en Matlab, encontradas en la World Wide Web de otros diseñadores.

3.7.2 Instrumentos para la recolección de los datos

A continuación, se muestran los instrumentos utilizados en la presente investigación.

Se utilizaron como fuentes secundarias: libros especializados sobre el procesamiento digital de señales (DSP), algoritmos de codificación de habla (speech coding algorithms), técnicas de procesamiento digital de señales usando MATLAB, tratamiento de señales en tiempo discreto; catálogos sobre especificaciones técnicas de códecs digitales; **recomendaciones técnicas** de la Unión Internacional de Telecomunicaciones (UIT): **bibliotecas virtuales** de artículos especializados referidos a la Codificación de Predicción

Lineal (LPC); **direcciones webs** sobre trabajos de investigación como antecedentes del Proyecto a desarrollo; marco teórico para la descripción de los parámetros del modelo de Predicción Lineal (LP); uso de **software** de simulación en la plataforma de desarrollo MATLAB y de las interfases gráficas del SIMULINK.

Se utilizaron como fuentes primarias: **Equipos audio-visuales** como PC o Smart TV, para la reproducción de videos tutoriales sobre vocoders.

3.8 Procesamiento, presentación, análisis e interpretación de los datos

El procesamiento de datos de mi investigación, consistió en la secuencia de actividades planificadas mediante el cual los datos individuales se agrupan y estructuran con el propósito de responder al problema de Investigación, a los objetivos planteados y a las hipótesis por comprobar.

Se efectuó un análisis cualitativo de la información obtenida mediante la simulación del proceso de comprensión de voz. Dichos parámetros del modelo de predicción lineal fueron: estimación del pitch, coeficientes de predicción lineal y la calidad de la voz sintetizada.

Las hipótesis planteadas fueron comprobadas con los resultados obtenidos. Por ser una investigación aplicada, no se requirió la contrastación de las hipótesis por el método estadístico.

Finalmente se presentaron las conclusiones y recomendaciones producto de los resultados obtenidos en mi investigación aplicada.

Capítulo IV: Los codificadores de voz en el procesamiento digital de señales

4.1 Consideraciones previas

El procesamiento digital de señales de voz ha sido y seguirá siendo una de las áreas de desarrollo que presenta mucho interés para su uso en los servicios de telecomunicaciones.

En la actualidad, la transmisión de señales de voz es uno de los servicios que más se utilizan en las redes de telecomunicaciones. En un inicio se pretendía que las redes procesaran únicamente aplicaciones de datos, sin embargo, en la actualidad con el uso intensivo de los servicios de telecomunicaciones se han desarrollado interfases para aplicaciones de voz, las cuales operan aceptablemente.

Relacionado con el proceso de reconocimiento de voz se encuentra su codificación. Según Huidobro y Moya (2003) manifiesta que:

> La codificación consiste en realizar un conjunto de transformaciones a la señal que representa la información a transmitir con el fin de mejorar la eficiencia sin pérdida de calidad de la comunicación, compensando los efectos negativos de la presencia del canal (ruido, desvanecimientos, interferencias, etc.). Para recuperar la información en el otro extremo será necesario un proceso inverso de decodificación (p.138).

Con referencia al objetivo principal de un codificador de voz, según Espinosa (2003) indica que:

> Es comprimir la señal, es decir, emplear pocos bits en la representación de

> voz en forma digital. En las pasadas cuatro o cinco décadas una variedad de técnicas de codificación de voz se han propuesto, analizado y desarrollado. En los vocoders, la descripción paramétrica del sistema bucal humano puede tomar una variedad de formas, ya sea en el dominio del tiempo o en el dominio de la frecuencia (p.44).

4.2 Descripción del problema focalizado

4.2.1 Presentación del nudo crítico

El reconocimiento automático de la voz, es un proceso que requiere en primer lugar, la representación de la señal de voz mediante un conjunto de vectores de parámetros acústicos, que contengan la información suficiente para poder identificar los sonidos en las siguientes etapas del sistema de reconocimiento. Esta etapa se le conoce como parametrización (Espinosa,2003).

En la gran mayoría de sistemas de reconocimiento de voz, este conjunto de vectores de parámetros se obtiene a partir de un análisis espectral localizado de la señal de voz (Hernando, 1993).

La codificación de voz o compresión de voz es un proceso que consiste en obtener una representación digital compacta de las componentes de voz, es decir, lo que se busca es reducir al máximo la cantidad de información transmitida. Al mismo tiempo, se desea conservar la calidad de la voz sintetizada y la reducir complejidad del codificador-decodificador. El proceso de codificación de voz requiere de acciones previas como el muestreo en el tiempo y la cuantificación en amplitud.

Los parámetros que se toman en cuenta para evaluar un algoritmo de codificación de voz son: los coeficientes de autocorrelación, una tasa de bits, la calidad de voz reconstruida técnica de codificación empleada y el retardo o delay presente.

En cuanto al uso de algoritmos de codificación a bajas tasas de bits, Espinosa (2003) indica que “se implementará en un procesador digital de señales, capaz de ejecutar 12 o más millones de instrucciones por segundo (MIPS). El

retardo (la codificación más la decodificación) que se introduce en algoritmos es generalmente de 50 a 60 ms".

De igual forma, cuando Espinosa (2003) se refiere a la codificación de voz a tasas medias, manifiesta que:

> La codificación de voz a tasas medias de bits se lleva a cabo usando un proceso de análisis-síntesis. En la etapa de análisis, la voz se representa en forma compacta de parámetros, la cual se codifica correctamente, en la etapa de síntesis esos parámetros son decodificados y usados en conjunto con otros mecanismos para formar la voz (p.3).

4.2.2 Características relevantes

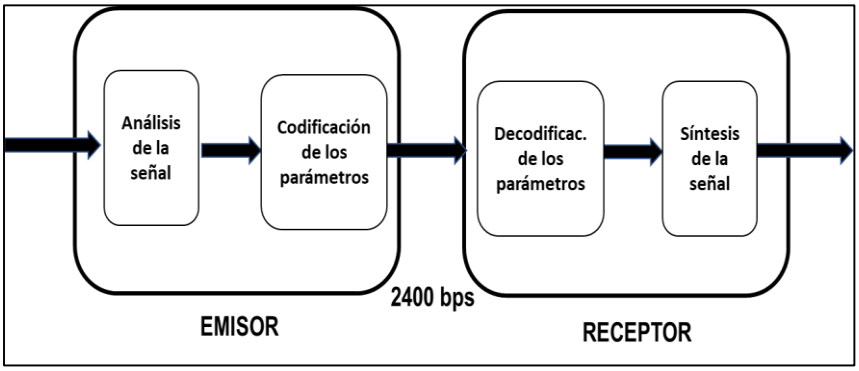
El codificador a diseñar tiene en cuenta la naturaleza de la señal a codificar, en este caso la voz y aprovechan las características de la misma para ganar en eficiencia. Permite trabajar con muy bajos *bit rates*, pero la señal de voz que producen suena demasiado sintética.

Los *vocoders* intentan producir una señal que suene como la voz original, independientemente de si la forma de onda se parece o no. Son las que menor ancho de banda utilizan (por debajo de 2.4 kbps, incluso se pretende llegar hasta 1 kbps), sin embargo, la calidad de voz es baja, su retardo alto, igual que su carga de proceso. Utilizan complejos modelos matemáticos de la voz humana, a través de los cuales obtienen ciertos parámetros particulares de la conversación en curso y son transmitidos.

El resultado es, que se produce voz inteligible a muy bajo *bit rate*, consiguiéndose entender la conversación, pero no se reconoce a la persona con la que se habla (la sensación es la del habla de un robot). Generalmente no se usa comercialmente, y sí militarmente, donde un bajo régimen binario facilita una compleja encriptación (por motivos de seguridad) a mucho menor costo.

Figura 15

Diagrama de bloques general de un vocoder



4.3 Análisis de factores críticos

4.3.1 Análisis Temporal de la voz

Dentro del análisis temporal de la voz, son de suma importancia establecer la caracterización de la voz y sus parámetros temporales, ya. que se aplican directamente a la forma de onda de la señal.

Según Espinosa (2003) referente al análisis temporal de la voz, manifiesta que:

> La señal de voz presenta grandes variaciones de amplitud (sonidos sonoros/sonidos no-sonoro). La energía en periodos cortos de tiempo refleja bien estas variaciones, permitiendo distinguir entre sonidos sonoros y no-sonoros.
>
> Es evidente, que el factor clave del análisis temporal es la ventana haming: su forma y su duración. Si la ventana es muy larga, entonces la energía no cambiará mucho de un segmento al siguiente, con lo que no reflejará correctamente los cambios de energía de la señal. Por otro parte, si es muy corta, la energía variará demasiado, con lo que tampoco nos servirá. Lo que buscamos es un filtrado paso bajo, de tal forma que en la energía quede una señal relativamente suave, sin variaciones bruscas, pero que represente adecuadamente la variación de la energía de la voz (pp. 9-10).

4.3.2 Discriminación voz-ruido

Uno de los principales problemas que existen en el análisis de la voz es poder distinguir cuando está presente una señal de voz y cuándo no. Los sonidos no-sonoros tienen el mismo comportamiento que el ruido.

Al respecto, Espinosa (2003) indica que:

> El problema de localización del principio y final de una palabra es difícil, sobre todo en sistemas de reconocimiento de voz. La técnica que se utiliza es la energía local y tasa de cruce por cero. Cuando el principio o final corresponda a un sonido sonoro, basta con calcular su energía local, que será muy superior a la del ruido (silencio). Si la palabra empieza o termina por sonidos no-sonoro, entonces la energía es muy parecida a la del ruido. Sin embargo, los sonidos no-sonoros tienen un contenido en frecuencia alta muy superior a la del ruido (p.11).

4.3.3 Análisis espectral de la voz

El problema del análisis espectral, definido como la obtención de la distribución frecuencial de potencia de un proceso aleatorio a partir de ciertas medidas realizadas en un intervalo temporal finito de una de sus realizaciones, ha sido objeto de numerosos estudios en los últimos años, de los cuales han surgido infinidad de técnicas.

4.3.4 Parámetros críticos

El codificador de voz que usa el modelo de predicción lineal (LPC) basado en el estándar FS-1015, es el vocoder más simple y a su vez el de baja calidad al momento de reproducir la voz (voz sintetizada), generalmente muestrea la señal a 8000 Hz y utiliza tramas de 30 ms (240 muestras).
Este vocoder analiza la señal de voz de la que extrae 3 tipos de parámetros críticos:
Coeficientes LPC (relacionados con los formantes: F1, F2, …).
Estimación del pitch
Factor de ganancia.

Una vez extraídos los parámetros, se envían al decodificador para que sintetice la señal de voz.
Necesitamos conocer estos parámetros críticos para que el procesamiento de la señal de

voz se adecuada y se pueda optimizar la señal reproducida a la salida del decoder (voz sintetizada).

Los coeficientes LPC

Los coeficientes LPC se pueden obtener resolviendo un sistema de ecuaciones, y la herramienta de simulación matemática "Matlab" define una función que facilita su cálculo. La manera más sencilla de usar dicha función en Matlab es [a,e]=lpc(s, N); donde "a" son los coeficientes LPC, "e", la potencia de la señal de error, "s" es un vector con la señal de la que se desea obtener los coeficientes del filtro de predicción lineal (en este caso, cada trama de la señal de voz), y "N" es el orden del filtro de predicción lineal.

El pitch

El pitch se trata de la frecuencia fundamental, en este caso, de cada trama de sonido analizada. La frecuencia fundamental es la frecuencia más baja de una forma de onda periódica. La estimación de la frecuencia del pitch la realiza el codificador, analizando la señal de error de predicción (que se puede demostrar que es una señal periódica con la misma periodicidad de la señal de voz).

En caso de analizar un sonido sonoro, una sencilla estrategia para estimar el pitch es calcular donde se encuentra el máximo de la señal de autocorrelación del error de predicción. Este máximo será el periodo del pitch (T). Teniendo en cuenta nuestra frecuencia de muestreo utilizada podremos obtener la frecuencia del pitch

Normalmente esta frecuencia se encuentra entre: 60Hz y 300Hz. La decisión de sonido sordo o sonoro se dará utilizando también la señal de autocorrelación del error de predicción, si el máximo de la autocorrelación normalizada supera valores de amplitud de 0.20, por recomendación, se considerará sonora, en caso contrario se considera sorda.

4.4 Dificultades a resolver

Una de las dificultades que se presenta en el modelamiento de un codificador de voz es la determinación de coeficientes LPC. Los diferentes métodos existentes desarrollan innumerables cálculos y requieren de mucha capacidad de memoria de almacenamiento. Existen los métodos directos y los métodos iterativos para determinar los coeficientes LPC. Considerando que los métodos directos, la disyuntiva se presenta entre elegir los dos métodos directos: autocorrelación y covarianza.

4.4.1 Autocorrelación

El método de la autocorrelación consiste en:

- Toma una ventana de voz y fuera de ella asume ceros.
- Requiere proceso de ventaneo.
- Da lugar a una matriz Toeplitz.
- Los LPC's pueden ser calculados eficientemente mediante el algoritmo de Levinson-Durbin.
- Implementación en celosía a través de los PARCOR.
- El filtro es siempre estable.

4.4.2 El método de la covarianza

El método de covarianza consiste en:

- Minimiza los errores al cuadrado sobre una ventana fija.
- No requiere proceso de ventaneo ya que no "extrapola" la señal con ceros.

- Da lugar a una matriz simétrica pero no Toeplitz.

- Los LPC's pueden ser calculados eficientemente mediante la descomposición de Cholesky.

- No siempre da un filtro estable.

4.4.3 Elección del método para calcular los coeficientes LPC

Se determinó utilizar el análisis de autocorrelación. La función de autocorrelación proporciona una medida de la correlación de la señal con una copia desfasada en el tiempo de sí misma. De aquí se extraen los coeficientes "p" de autocorrelación, valores típicos que pueden ser entre 10 y 15. Podemos identificar los coeficientes de autocorrelación en las ecuaciones que minimizan los errores en la estimación de la señal predicha.
Para resolver este conjunto de ecuaciones se recurre al algoritmo de Levinson-Durbin que permite resolver el sistema de ecuaciones de una forma eficiente. Teniendo los coeficientes del filtro se dispone, para la ventana de análisis, de la función de transferencia del modelo del tracto vocal en ese instante.

Capítulo V: Desarrollo de un codificador paramétrico de voz (vocoder) LPC, basado en el estándar FS-1015

En este capítulo se presenta procedimiento a seguir para hacer una propuesta de desarrollo de un codificador paramétrico de voz (vocoder), utilizando el modelo de codificación de predicción lineal (LPC) basado en el estándar FS-1015.

De modo general, un vocoder LPC bajo el estándar FS-1015, proporciona una función de análisis en el extremo transmisor (encoder) y una función de síntesis en el extremo receptor (decoder).

A continuación, se indicará la secuencia de pasos a seguir para el desarrollo del vocoder LPC:

1. Descripción de la propuesta de desarrollo de un vocoder LPC.

2. Descripción de la estructura del vocoder, a través del desarrollo del diagrama de flujo del procesamiento de la voz en codificador y el decodificador FS-1015. Luego se mostrará el diagrama de flujo integrador del vocoder.

3. El programa desarrollado para la simulación del proceso de comprensión de voz mediante la codificación de predicción lineal, utilizando el software MATLAB.

5.1 Descripción de la propuesta de desarrollo de un vocoder LPC

Para la descripción de la propuesta del vocoder LPC, se seguirá el siguiente

procedimiento:

1. Se hará una breve descripción de lo que consiste el codificador (encoder) LPC y luego una breve descripción de la interacción de los bloques que lo conforman.

2. Se hará una breve descripción de lo que consiste el decodificador (decoder) LPC y luego una breve descripción de la interacción de los bloques que lo conforman.

3. Se explicará brevemente las etapas que componen el encoder LPC del estándar FS-1015.

4. Se explicará brevemente las etapas que componen el encoder LPC del estándar FS-1015.

5.1.1 Codificador LPC

El análisis en el extremo transmisor (encoder) de un codificador de voz LPC, consiste en:

- Estimación de la periodicidad / aperiodicidad de la señal y de la frecuencia fundamental.

- Extracción de los coeficientes LPC.

- Cálculo de la amplitud.

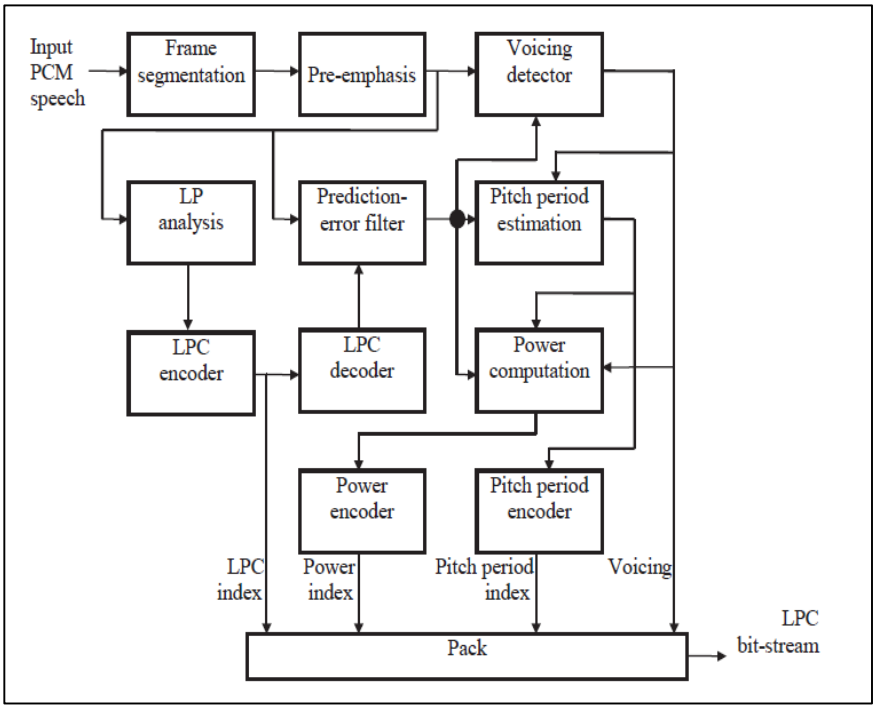
- Cálculo del residuo (diferencia entre la codificación LPC y la señal original).

- Almacenamiento en memoria de los parámetros extraídos.

En la figura 9 se puede apreciar el diagrama de bloques del codificador (encoder) basado en el modelo de codificación de predicción lineal (LPC).

Figura 16

Diagrama de bloques del codificador



Fuente. Recuperado de Chu (2003)

En el codificador LPC, la voz de entrada se muestrea primero a 8 kHz para obtener la voz de PCM, eso es 16 bits / muestra. Luego, la voz se segmenta en tramas sin superposición. Cada trama puede durar entre 10 y 30 ms. Luego un filtro de pre-énfasis se utiliza en la voz para ajustar su espectro. La salida se envía a un bloque de análisis LP, al filtro de error de predicción y también al detector de voz que emite 1 bit como sonoro o sordo. La salida de la voz del detector se envía tanto al bloque de cálculo de potencia como al bloque de la estimación del pitch.

La salida del filtro de error de predicción también se utiliza para la estimación del período del pitch, detección de voz y cálculo de potencia. La potencia esta codificada en el codificador de potencia. El período del pitch es

codificado en el codificador de período del pitch sólo si la trama es de voz. El codificador LPC también se utiliza en el decodificador LPC.

Los coeficientes LP se codifican en el codificador LPC. Todos los bits codificados (para coeficientes LP, potencia, período de tono y voz) se empaquetan y transmiten al receptor en un flujo de bits LPC.

Todos estos parámetros (período de tono, ganancia, decisión de voz y coeficientes de filtro) se determinan a partir de una trama de la señal de voz durante la conversación análisis. Los coeficientes de filtro son fijos para una trama de voz (típicamente 10-20 ms) durante el cual la señal de voz se supone estacionaria. Los coeficientes deben calcularse y utilizarse para una nueva trama de voz.

5.1.2 Decodificador LPC

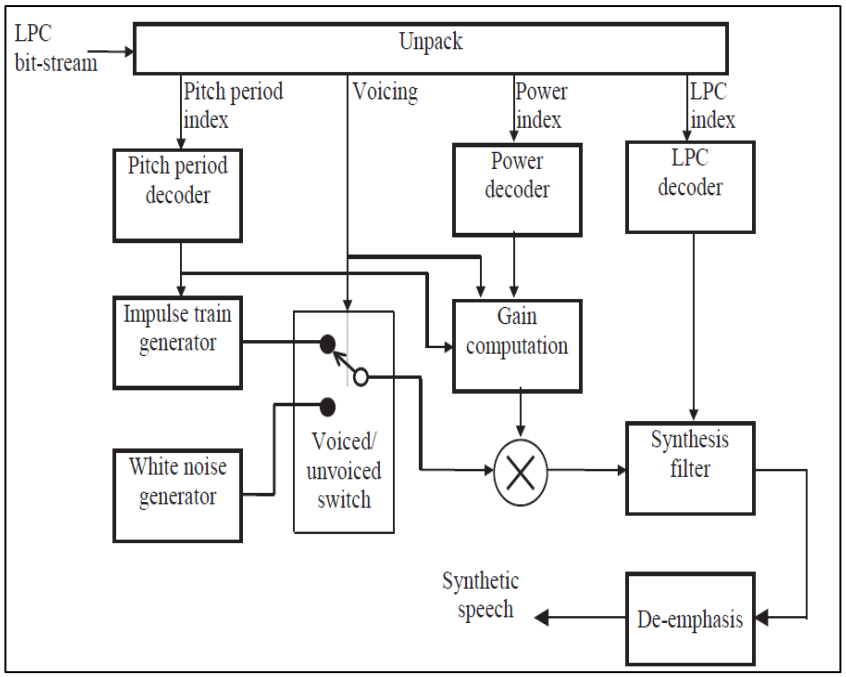
La síntesis en el extremo receptor (decoder), consiste en elegir una forma de onda, bien periódica o bien ruido blanco, para excitar el filtro de síntesis. Está síntesis consiste en:

- Lectura de los parámetros de la memoria
- Generación de la onda periódica/ aperiódica
- Cálculo de la función de transferencia a partir de los coeficientes LPC
- Aplicación de la función de onda procedente de la fuente
- Control de la amplitud
- Conversión digital / analógica (D/A)

En la figura 17 se puede apreciar el diagrama de bloques del decodificador (decoder) basado en el modelo de codificación de predicción lineal (LPC).

Figura 17

Diagrama de bloques del decodificador



Fuente. Recuperado de Chu (2003)

En el decodificador LPC, el flujo de bits codificado LPC es el primero desempaquetado. El índice del período del pitch se envía al decodificador de período del pitch. El bit de voz controla el interruptor de decisión vocal / no vocal. El índice de potencia de los bits, se envían al decodificador de potencia. Los bits del índice de LPC se envían al decodificador LPC para su decodificación, cuya salida se utiliza para determinar los coeficientes del filtro de síntesis. La excitación del interruptor vocal / no vocal se mezcla a la salida con el bloque de cálculo de ganancia y se utiliza para excitar el filtro de síntesis. La salida del filtro de síntesis es reajustada por el filtro de de-énfasis para obtener la señal de voz sintética.

5.1.3 Encoder FS-1015

El encoder del estándar FS 1015 es el algoritmo encargado de convertir la señal de audio capturada con una tasa de 128 kbps (8 KHz, 16 bits por muestra) a

una señal comprimida con una tasa de 2.4 kbps.

El encoder LPC basado en el estándar FS 1015 se compone de las siguientes etapas:

1. Segmentación de la trama de entrada en bloques de 240 muestras

Esta etapa recibe las muestras de audio a la entrada del códec, y las acumula en bloques de 240 muestras para su posterior procesamiento. La razón de utilizar 240 muestras es que comúnmente esta es la máxima longitud en la que una señal de voz mantiene condiciones de estacionariedad. Cada bloque se denominará x_in [n].

2. Filtro de pre-énfasis

El filtro de pre-énfasis se encarga de "aplanar" el espectro del bloque de entrada. Logra esto mediante un filtro pasa altas para reducir la amplitud de las componentes de baja frecuencia. Para el códec FS 1015, la función de transferencia del filtro de pre-énfasis es:

$$H_{emp}(z) = 1 - 0.9375z^{-1}$$

3. Detección de voz

La decisión de si un bloque corresponde a voz (sonidos vocales) o no voz (sonidos no vocales) es un punto crítico en el algoritmo de codificación, ya que determina la cantidad de coeficientes de predicción lineal a utilizar, la fórmula de cálculo de potencia y la estimación del pitch. Este proceso se realiza con otro algoritmo de estimación estadística: un clasificador lineal. Primero, se calcula un conjunto de atributos a partir del bloque de entrada x[n].: Los atributos son los siguientes:

$$LBE = \frac{1}{26}\sum_{i=0}^{26} X[i]X^{*}[i]$$

LBE es la energía de bandas bajas de frecuencia. El "*" representa el conjugado, y $X[k]$ es la transformada de Fourier de $x[n]$.

$$PBR = \left|\frac{\max(R_{xx}[m])}{\min(R_{xx}[m])}\right|$$

PBR es la relación entre el máximo y mínimo de la autocorrelación, siendo $R_{xx}[m]$ la autocorrelación de $x[n]$.

$$ZC = \frac{1}{2}\sum_{i=0}^{238}|x[i] - x[i+1]|$$

Y finalmente, la variable ZC es la relación de cruces por cero de la función $x[n]$.

Luego, se genera una base de datos considerando LBE, PBR y ZC como atributos de cada uno de los bloques de 240 muestras. Sea N el total de bloques de voz de la base de datos, se definen las siguientes variables:

$$\mathbf{X} = \begin{bmatrix} LBE_1 & LBE_2 & \dots & LBE_N \\ PBR_1 & PBR_2 & \dots & PBR_N \\ ZC_1 & ZC_2 & \dots & ZC_N \end{bmatrix}$$

Esta matriz **X** tiene 3 filas que corresponden a los atributos LBE, PBR y ZC, y tiene N columnas que corresponde al total de bloques de voz de la base de datos. Cada bloque de voz está etiquetado con un "1" si el bloque es tipo "voz", y un "0" si es tipo "no voz". Se define la siguiente otra variable:

$$\mathbf{y} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{bmatrix}, c_i = \{0; 1\}$$

Es decir, cada entrada del vector **y** tiene un valor de 1 o 0 según la etiqueta que tenga el bloque de voz "i".

Habiendo definido la matriz de datos **X** y el vector de clases **y**, se calculará un plano que pueda separar de manera óptima los N puntos. La ecuación del plano será:

$$\hat{\mathbf{y}}^T = \mathbf{w}^T\mathbf{X} = [w_0 \quad w_1 \quad w_2]\begin{bmatrix} LBE_1 & LBE_2 & \dots & LBE_N \\ PBR_1 & PBR_2 & \dots & PBR_N \\ ZC_1 & ZC_2 & \dots & ZC_N \end{bmatrix}$$

Donde **w** es un vector de 3 filas y 1 columna que define el plano separador. $\hat{\mathbf{y}}$ es la estimación que se genera tras proyectar la matriz de datos **X** en el plano separador especificado por **w**.

Para este trabajo, se utilizó una base de datos de N = 900 datos. La forma de calcular el plano fue mediante una máquina de soporte vectorial lineal.

4. Análisis de predicción lineal

El proceso de análisis de predicción lineal se explica en la sección de marco teórico. Se obtiene un filtro de predicción lineal, cuyos coeficientes serán h[n]. Asimismo, el algoritmo de Levinson-Durbins utilizado en el análisis de predicción, devuelve un conjunto de coeficientes de reflexión, que serán k[n]. Los coeficientes h[n] y k[n] son dos formas de representar el mismo filtro, pero k[n] se utiliza en el proceso de codificación y cuantización.

Si el bloque de 240 muestras corresponde a voz, entonces se realiza la predicción lineal considerando 10 coeficientes. Por otro lado, si el bloque corresponde a no voz, entonces se realiza la predicción solamente con 4 coeficientes.

5. Codificación y cuantización del filtro de predicción lineal

Tras calcular el filtro de predicción lineal, se procede a codificar y cuantizar

los coeficientes de reflexión $k[n]$. La codificación se realiza según un conjunto de tablas de búsqueda especificadas en el estándar FS 1015, dónde los códigos dependen del número del coeficiente. Asimismo, la cantidad de bits a utilizar también es dependiente del número del coeficiente, según la tabla siguiente:

Tabla 2

Cantidad de bits de cuantización según coeficiente de reflexión

Nro. De coeficiente de reflexión	Cantidad de bits para cuantización
1	5
2	5
3	5
4	5
5	4
6	4
7	4
8	4
9	3
10	2

Nota. Recuperado de Chu (2003)

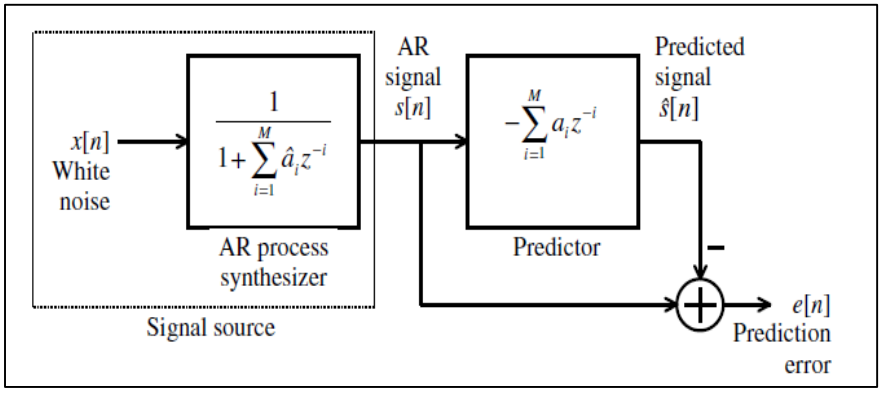
Es decir, los coeficientes de "menor orden" (1ro al 4to) se codifican con más bits que los de "mayor orden".

6. Cálculo del error de predicción

El error de predicción, $e[n]$, se calcula a partir de la señal enfatizada $x[n]$ y la señal de predicción $\hat{x}[n]$. La señal de predicción se calcula utilizando el filtro de predicción lineal, $h[n]$. El proceso se ilustra en la figura 18:

Figura 18

Error de predicción



Nota. Recuperado de Chu (2003)

La forma de onda del error de predicción tendrá similitudes con la forma de onda de la señal enfatizada $x[n]$. Por ello, el error de predicción se utiliza para calcular la potencia del bloque y, si se trata de un bloque tipo voz, para estimar el pitch.

7. Estimación del pitch

El pitch o tono T corresponde al período (en cantidad de muestras) de una señal de voz tipo vocal. Normalmente, se relaciona un pitch mayor con un tono de voz mas "grave" o "grueso"; y un pitch menor, con un tono de voz más "agudo" o "fino". Matemáticamente, se puede estimar el pitch mediante la autocorrelación. En este caso, primero se calcula la autocorrelación del error de predicción, $R_{ee}[m]$. Luego, para una muestra inicial "m" y un número total de muestras "N", se utiliza el siguiente algoritmo para estimar el pitch:

Si un bloque de señal es de tipo no vocal, no se calcula ningún pitch.

8. Codificación y cuantización del pitch

Según el estándar FS 1015, el pitch se define entre 51 y 400 Hz, y con 60 valores solamente. La tabla para codificar el pitch se especifica en el estándar, y el

valor codificado se cuantiza con 7 bits.

9. Estimación de la potencia

Si el bloque corresponde a una señal tipo vocal, la potencia se estima de la siguiente manera:

$$p = \frac{T}{[NT]} \sum e[n]^2$$

Donde $e[n]$ es el error de predicción, T es el pitch y N es 240.

Por otro lado, si corresponde a señal del tipo no vocal, la potencia se estima según:

$$p = \frac{1}{N} \sum e[n]^2$$

10. Codificación y cuantización de la potencia

La potencia primero se escala de 0 a 511, luego se codifica según una tabla especificada en el estándar, y finalmente se cuantiza el valor codificado con 5 bits.

Finalmente, en el encoder, se arma la trama del audio codificado, concatenando las salidas de cada bloque descrito previamente. Esta trama será decodificada por el decoder para posteriormente obtener la señal de voz sintetizada.

5.1.4 Decoder FS-1015

El decoder del FS 1015, toma la trama que sale del encoder, la decodifica y con esos parámetros, sintetiza la voz. En primer lugar, el decoder leerá si el bloque de voz corresponde a voz vocal o no vocal. Si corresponde a voz vocal, leerá el valor del pitch codificado y cuantizado, y lo decodificará. Para dicho tramo, generará un tren de impulsos, donde la separación entre un impulso y el siguiente es igual al pitch decodificado. En caso el bloque es del tipo no vocal, entonces

generará un bloque de ruido blanco, con una potencia igual a la potencia igual a 1.

Posteriormente, el decoder leerá y decodificará la potencia enviada por el encoder. Esta potencia amplificará o atenuará la amplitud de la señal del tren de impulso/ruido blanco. En cualquiera de los casos, el decoder leerá y decodificará los coeficientes de predicción lineal. La señal primero se mezclará con el bloque de cálculo de ganancia, y luego se utilizará como entrada el filtro de predicción, pero en el modo de síntesis.

Finalmente, la salida del filtro de síntesis atravesará un filtro de de-énfasis, que es igual a H_demp (z)=1/(H_emp (z)). Esta señal de salida, corresponde a la señal de voz sintetizada.

5.2 Descripción de la estructura del vocoder

La estructura del vocoder se explicará mediante el empleo de diagramas de flujo, que explican cómo se procesa la información (voz/no voz).

A continuación, se presentarán los siguientes diagramas de flujo que describen el procesamiento de la señal en el vocoder:

1. Diagrama de flujo del encoder FS-2015

2. Diagrama de flujo del decoder FS-2015

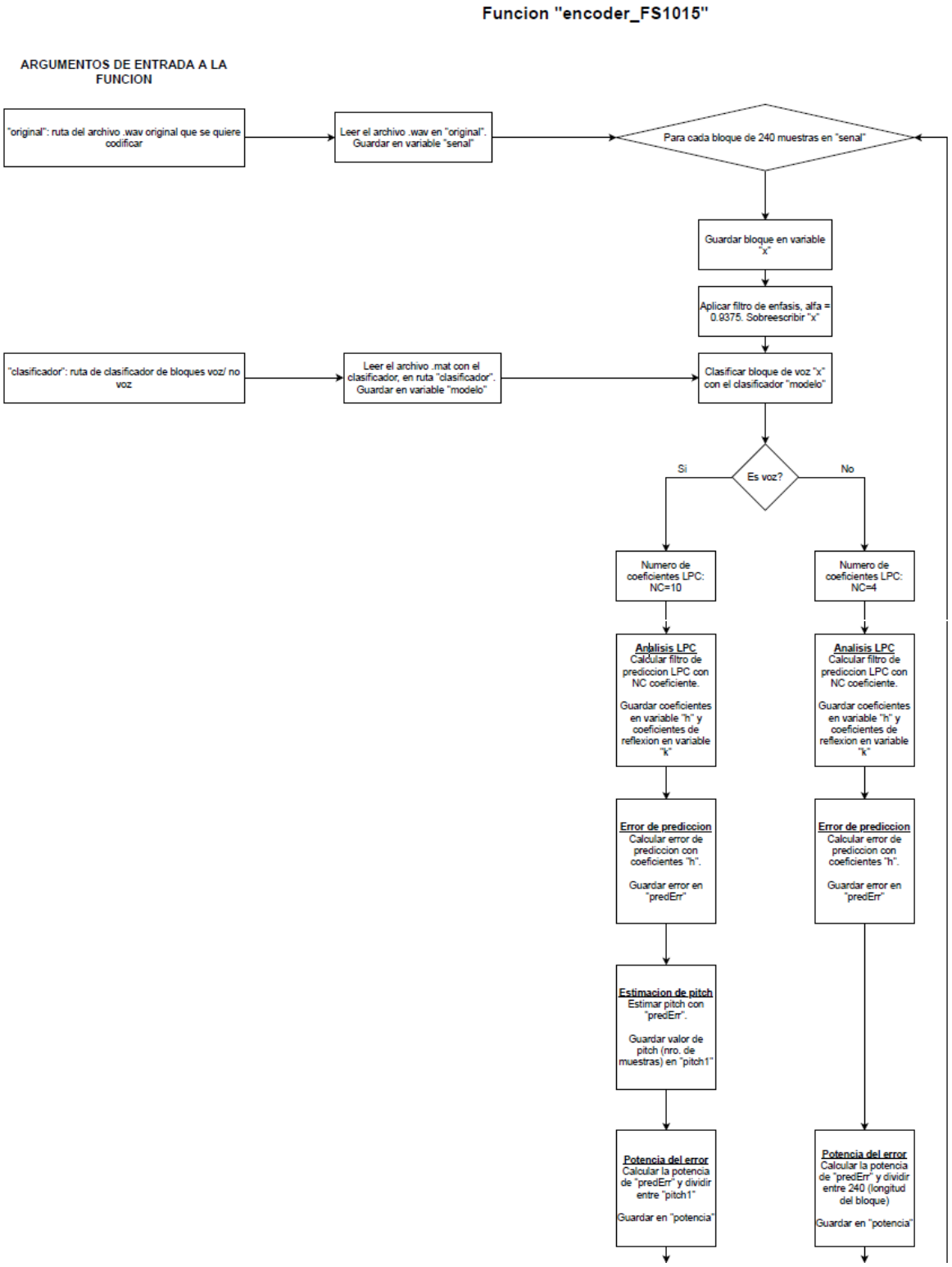
3. Diagrama de flujo integrador del vocoder FS-2015

5.2.1 Diagrama de flujo del encoder FS-1015

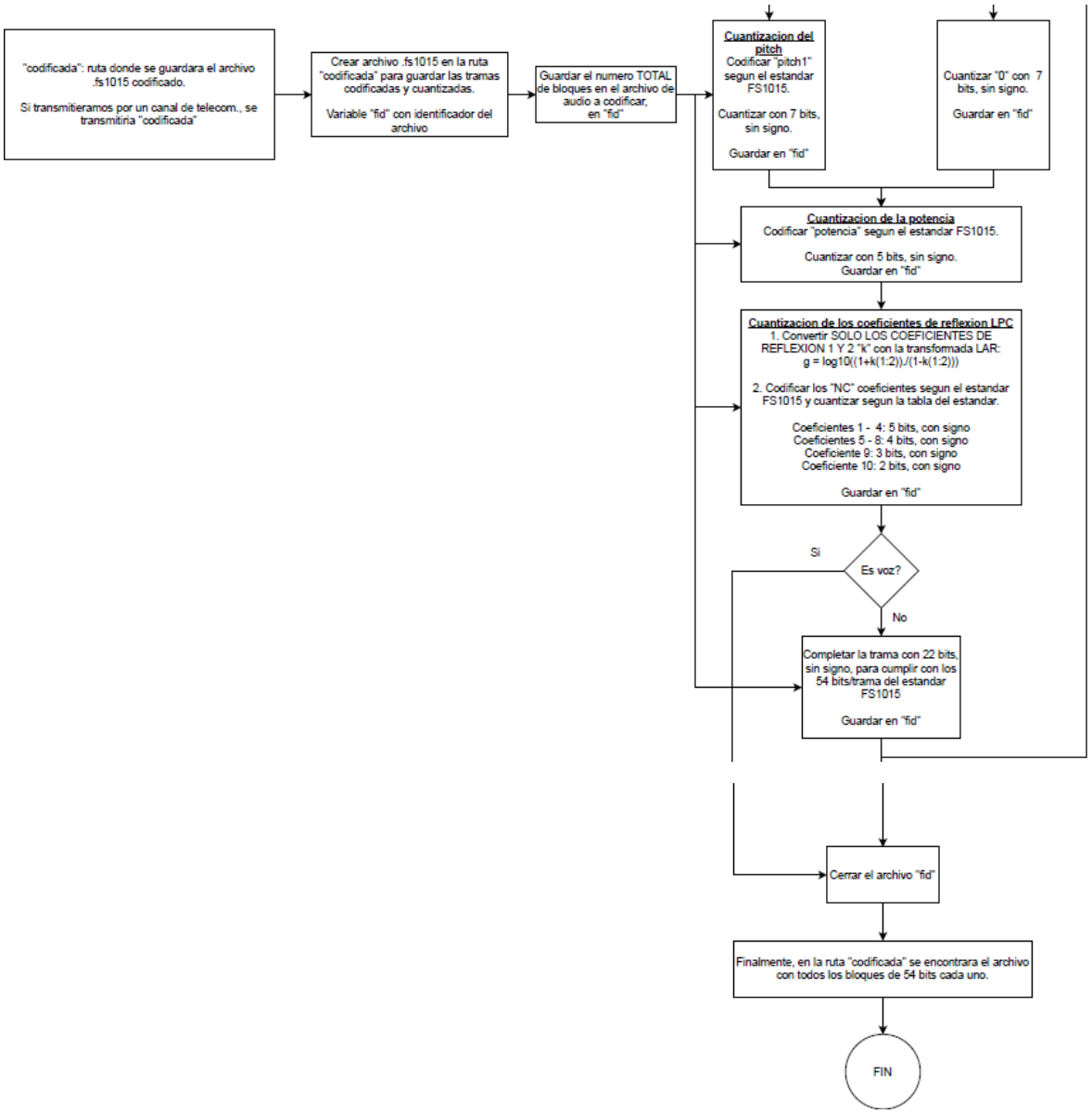
En la figura 19, se muestra el diagrama de flujo de un codificador (encoder) FS-1015.

Figura 19

Diagrama de flujos del encoder FS-1015



Continuación de función “encoder_FS1015”

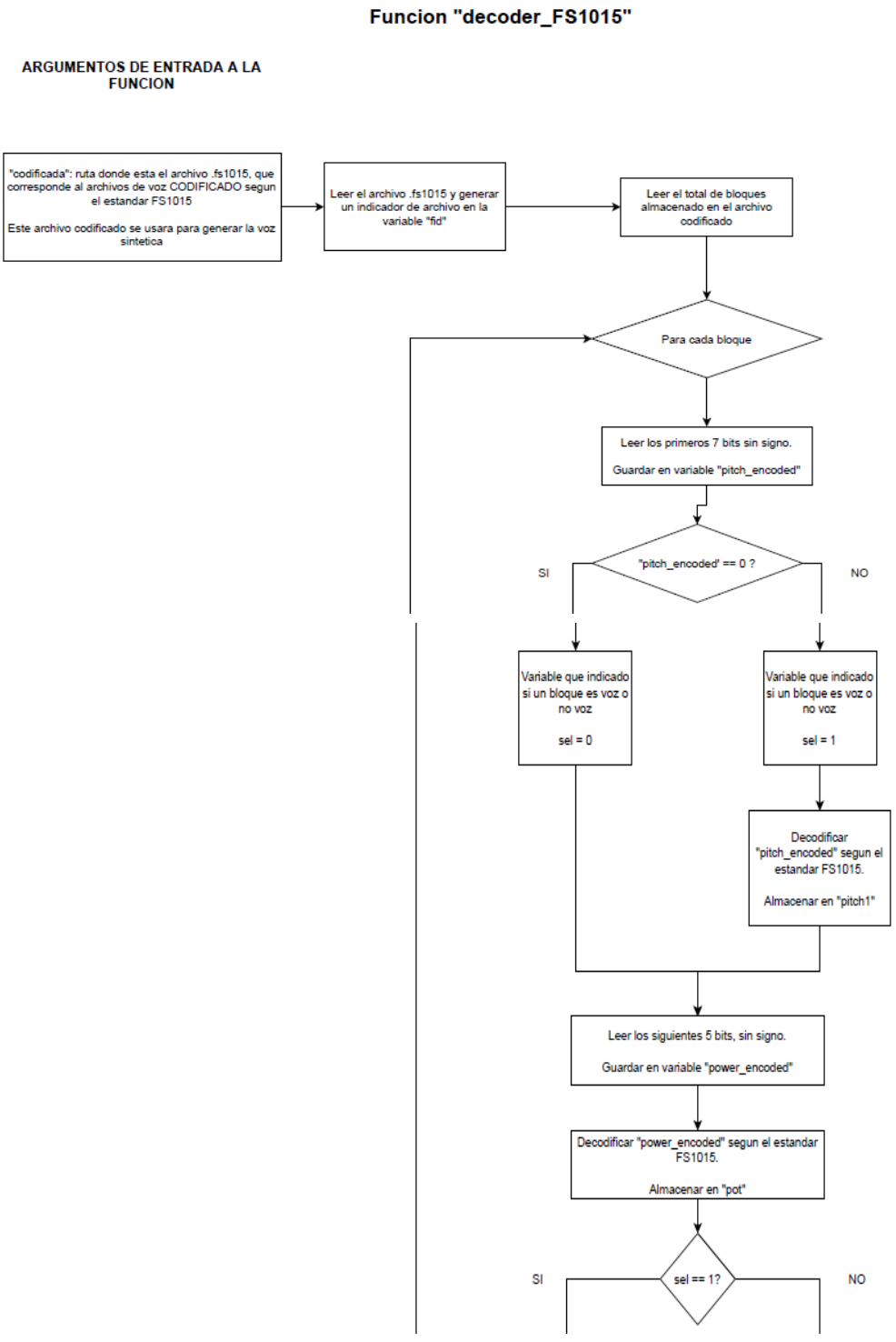


5.2.2 Diagrama de flujo del decoder FS-1015

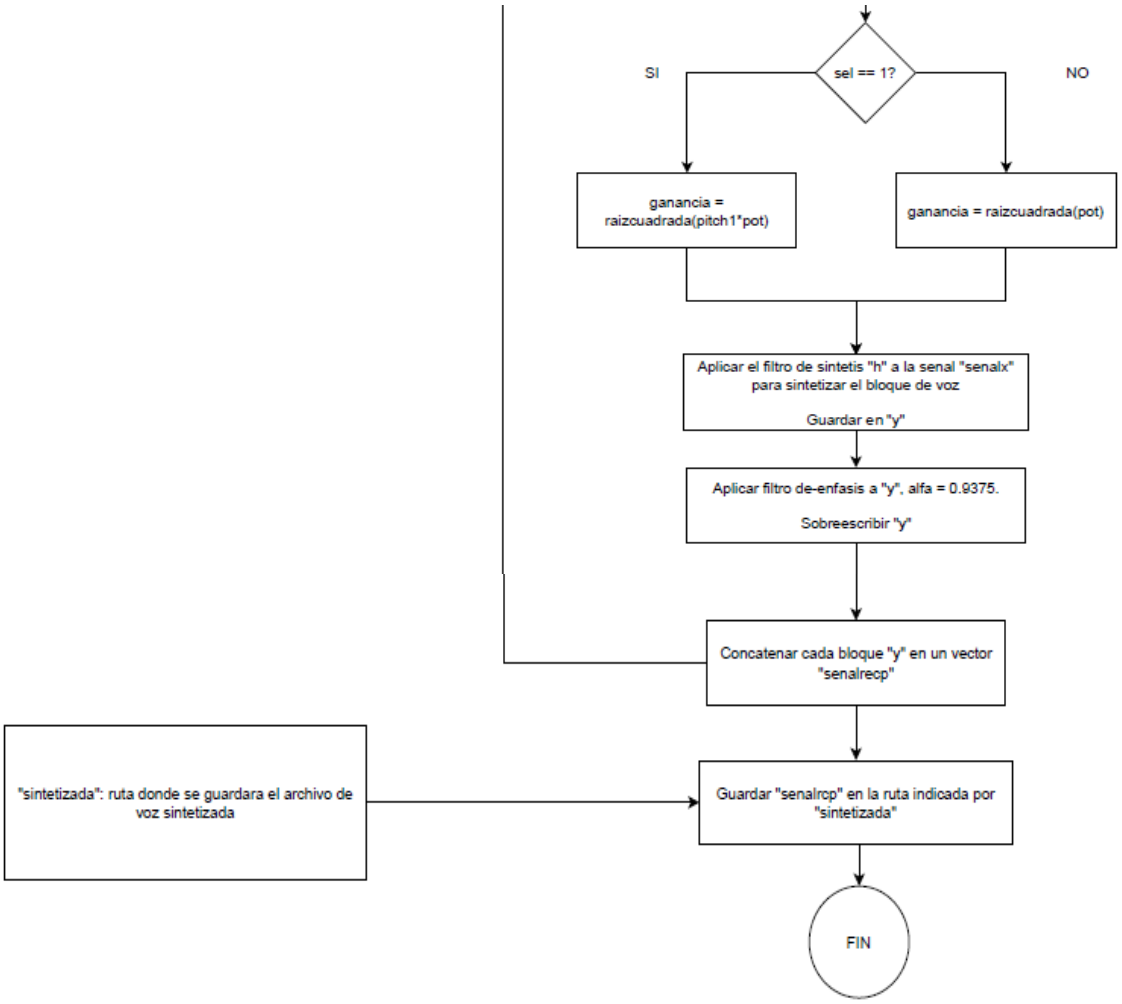
En la figura 20, se muestra el diagrama de flujo del decodificador (decoder) FS-1015.

Figura 20

Diagrama de flujo del decoder FS-1015



Continuación de función “decoder_FS1015”

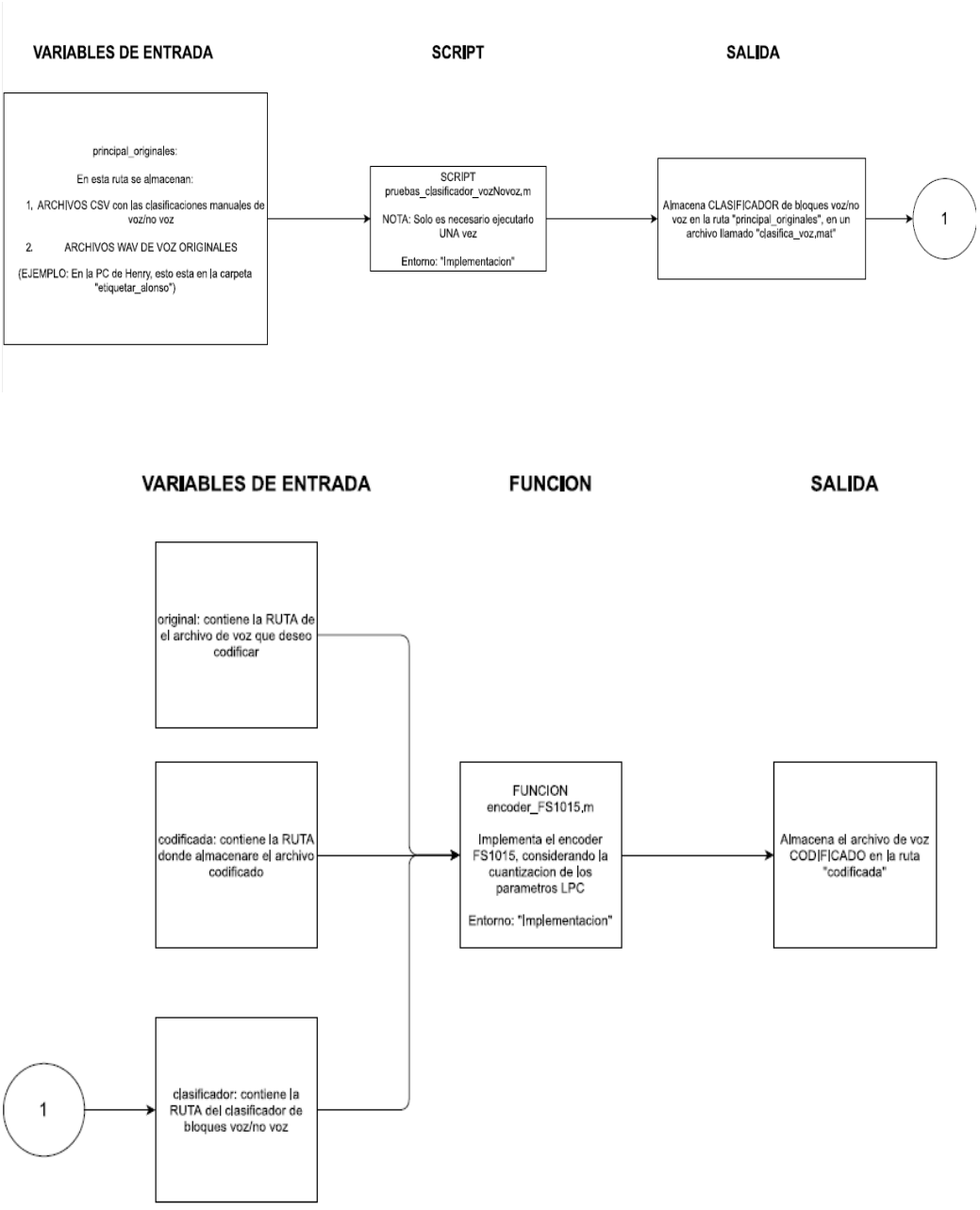


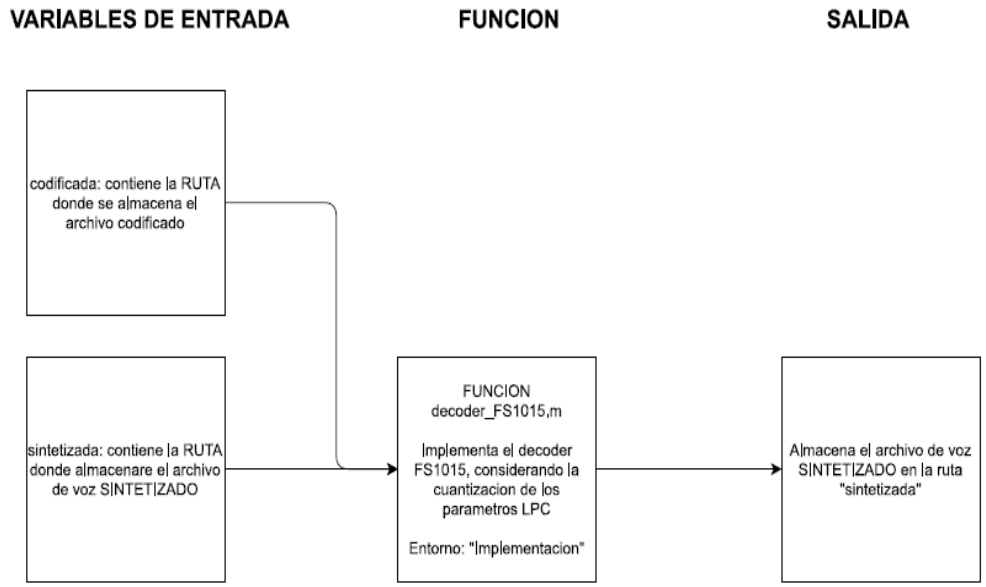
5.2.3 Diagrama de flujo integrado del vocoder FS-1015

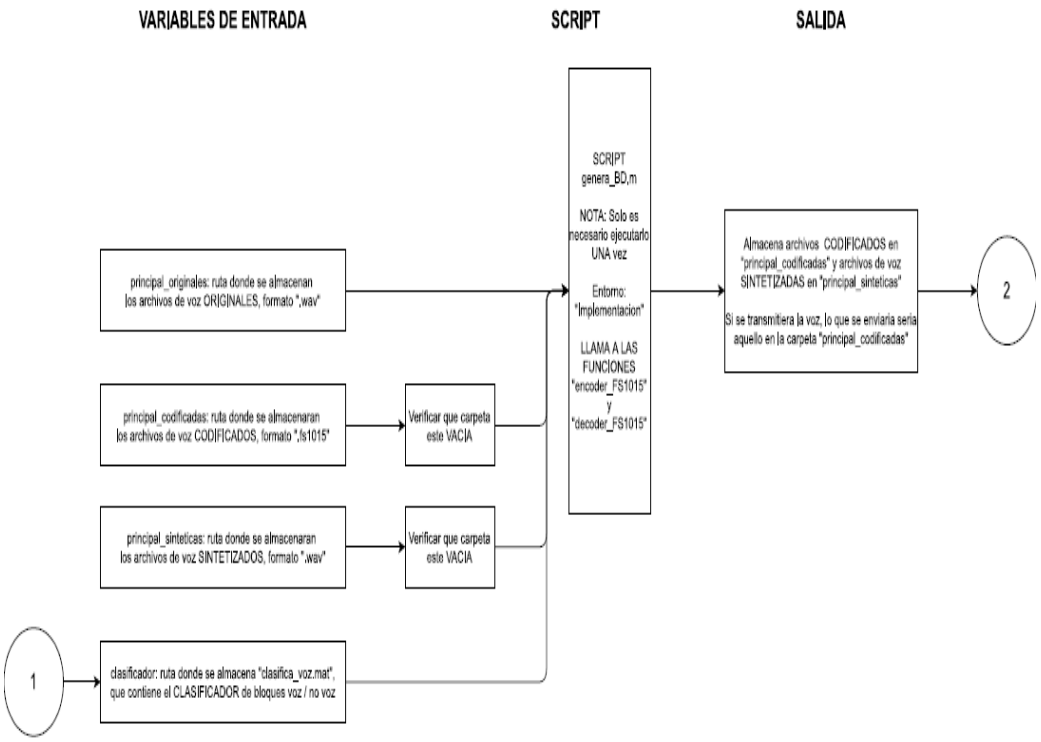
En la figura 21, se muestra el diagrama del flujo integrador del vocoder FS-1015.

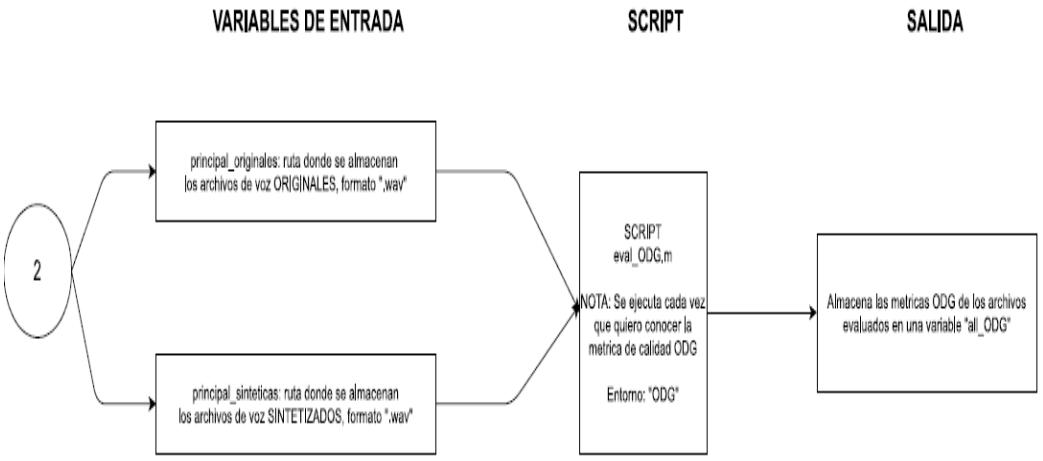
Figura 21

Diagrama de flujo integrador del vocoder FS-1015




VARIABLES DE ENTRADA
FUNCION
SALIDA
codificada: contiene la RUTA donde se almacena el archivo codificado
sintetizada: contiene la RUTA donde almacenare el archivo de voz SINTETIZADO
FUNCION decoder_FS1015,m
Implementa el decoder FS1015, considerando la cuantizacion de los parametros LPC
Entorno: "Implementacion"
Almacena el archivo de voz SINTETIZADO en la ruta "sintetizada"


VARIABLES DE ENTRADA
SCRIPT
SALIDA
principal_originales: ruta donde se almacenan los archivos de voz ORIGINALES, formato *.wav
principal_codificadas: ruta donde se almacenaran los archivos de voz CODIFICADOS, formato *.fs1015
principal_sinteticas: ruta donde se almacenaran los archivos de voz SINTETIZADOS, formato *.wav
Verificar que carpeta este VACIA
Verificar que carpeta este VACIA
clasificador: ruta donde se almacena "clasifica_voz.mat", que contiene el CLASIFICADOR de bloques voz / no voz
1
SCRIPT genera_BD,m
NOTA: Solo es necesario ejecutarlo UNA vez
Entorno: "Implementacion"
LLAMA A LAS FUNCIONES "encoder_FS1015" y "decoder_FS1015"
Almacena archivos CODIFICADOS en "principal_codificadas" y archivos de voz SINTETIZADAS en "principal_sinteticas"
Si se transmitiera la voz, lo que se enviaria seria aquello en la carpeta "principal_codificadas"
2



5.3 Simulación de un vocoder LPC del estándar FS-1015 usando MATLAB

A continuación, se presenta los programas y sus estructuras, tanto del clasificador de voz (voz/no voz), el encoder y del decoder FS-1015 utilizando los scripts del software de simulación MATLAB.

A fin de facilitar la descripción, se hace una presentación referencial de los scripts “encoder_FS1015.m” y “decoder_FS1015.m”.

Los scripts integrales de “encoder_FS1015.m” y “decoder_FS1015.m”, se presentan en el Anexo 1, así como los scripts de las funciones utilizadas para cada etapa.

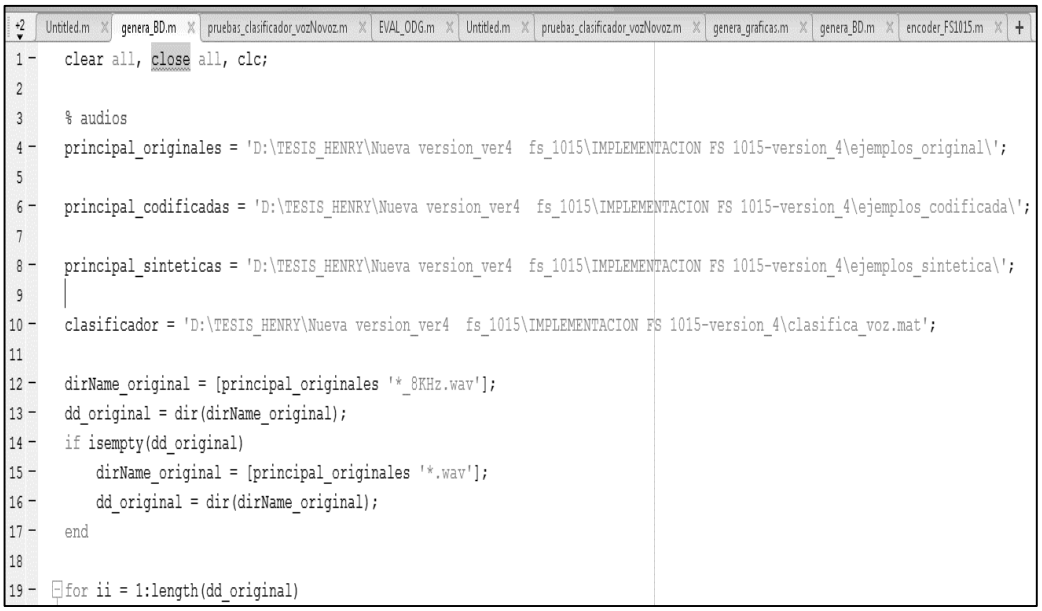
5.3.1 Generación de la base de datos de voz

Se requiere generar una base de datos, que servirá para efectuar las pruebas de clasificación de voz y para evaluar la métrica del grado de diferencia objetiva (ODG).

Se desarrolla el **script "genera_BD.m".** Primero, este script lee todos los

archivos de voz (formato .wav) en la carpeta "principal_originales". Si los archivos tienen frecuencia de muestreo de 48 KHz, el script primero los sub-muestrea a 8 KHz, que es la frecuencia de muestreo válida para el codec FS 1015. Luego, el script codifica cada archivo en la carpeta "principal_originales" con encoder_FS1015() y almacena el archivo codificado en la carpeta "principal_codificadas".

Finalmente, el script decodificado el archivo codificado con decoder_FS1015() y guarda el archivo de voz (formato .wav) sintetizada en la carpeta "principal_sinteticas". Asimismo, el script necesita la RUTA/DIRECCION del clasificador de voz/no voz, que es el archivo "clasifica_voz.mat".



Finalmente, el script **"genera_BD.m"** habrá generado los siguientes archivos, en las carpetas:

- En "principales_originales": los archivos de voz ORIGINALES (formato .wav) con frecuencia de muestreo 8 KHz.

- En "principales_codificadas": los archivos de voz CODIFICADOS (extensión .fs1015).

- En "principales_sinteticas": los archivos de voz SINTETIZADOS (formato .wav) con frecuencia de muestreo 8 KHz.

Los archivos de voz originales y sintetizados se utilizarán con el script de la métrica ODG, para calcular la calidad después de la codificación.

5.3.2 Encoder FS-1015

1. Segmentación de la trama de entrada en bloques de 240 muestras

En el código de MATLAB desarrollado, este primer paso de la codificación se realiza en la función "**encoder_FS1015.m**", línea 65:

```
65 -  for n=1:Numblq % Procesamiento por bloque
66 -      x=senal(m:m+N-1); % Extracción de bloques de N muestras
67
68        %-------------------------------------------------------
```

La variable "senal" contiene la señal de voz completa, cargada a partir de un archivo en formato *wav*. La variable "Numblq" contiene el total de bloques que contiene "señal" para el archivo de voz cargado. La variable "x" almacena el bloque segmentado de N=240 muestras.

2. Filtro de pre-énfasis

El filtro de pre-énfasis se realiza en la función "**encoder_FS1015.m**", línea 70:

```
68        %-------------------------------------
69        % FILTRO DE ENFASIS
70 -      x = filter([1, -alfa],1,x);
71        %-------------------------------------
```

En el estándar FS 1015, la constante del filtro de énfasis es $\alpha = 0.9375$. La variable "x" se sobreescribe para almacenar ahora el bloque de señal enfatizado.

3. Detección de voz

Para la detección de voz, primero se calculan los atributos LBE, PBR y ZC en la función "**encoder_FS1015.m**", y luego se predice con la variable "modelo", que contiene el clasificador de bloques de voz:

```
74        % SELECCION DE VOZ O NO VOZ
75        % low band energy (from 0 Hz to 800 Hz)
76 -      xFFT = fft(x,256);
77        % si 128 es 4 KHz, 800 Hz es 96
78 -      xFFT = xFFT(1:26);
79 -      LBE = sum(xFFT.*conj(xFFT))/length(xFFT);
80        % escalar para ingresar al clasificador
81  %       LBE = (LBE - mean_TD(1))/std_TD(1);
82
83        % PEAK/BOTTOM RATIO OF magnitude difference function // autocorr
84 -      PBR_MDF = xcorr(x,x);
85 -      PBR_MDF = abs(max(PBR_MDF)/min(PBR_MDF));
86        % escalar para ingresar al clasificador
87  %       PBR_MDF = (PBR_MDF - mean_TD(2))/std_TD(2);
88
89        % zero crossing rate
90 -      ZC = 0.5*sum(abs( sign(x(1:end-1)) - sign(x(2:end)) ));
91        % escalar para ingresar al clasificador
92  %       ZC = (ZC - mean_TD(3))/std_TD(3);
93
94  %       disp([LBE,PBR_MDF,ZC]);
95
96       % clasificacion de voz y no voz
97 -      sel = modelo.predict([LBE,PBR_MDF,ZC]);
```

La variable "modelo" con el clasificador de voz/no voz se determinó utilizando el script "**pruebas_clasificador_**vozNovoz**.m**".

Dicho script utiliza una base de datos donde se ha etiquetado manualmente cerca de 800 bloques de voz, según se haya determinado si el bloque es vocal o no vocal. Las etiquetas de los bloques se almacenan en archivos en formato *csv*, donde cada archivo *csv* contiene todas las etiquetas de todos los bloques de un archivo de

voz *wav*. El script se encuentra en el Anexo 1.

4. Análisis de predicción lineal

El análisis de predicción lineal se ejecuta en el script "**encoder_FS1015.m**", línea 111:

```
110        %----------------------------------------
111 -      [h,k]=analisis_lpc(x,NC,N); % Análisis LPC
```

La función "**análisis_lpc.m**" que calcula los coeficientes óptimos *h* y de reflexión *k*, utiliza a su vez 2 funciones más:

```
function  [h,k]=analisis_lpc(x,NC,N)

R=autocorrelacion(x,NC,N);
[h,k]=durbins(R,NC);
```

La función "**autocorrelacion.m**" calcula la autocorrelación de un bloque de voz, y la función "**durbins.m**" calcula los coeficientes resolviendo la ecuación normal.

5. Codificación y cuantización del filtro de predicción lineal

La codificación y cuantización del filtro de predicción (coeficientes de reflexión) se realiza en "**encoder_FS1015.m**":

```
168        % finalmente los coeficientes
169 -      g = log10((1+k(1:2))./(1-k(1:2))); % convertir los coeficientes con la transformacion LAR
170    %     g = k;
171        %%% cuantizacion de coeficientes de reflexion
172
173 -      for nbs = 1:2
174 -          Q = RC_coding(g(nbs),nbs);
175 -          fwrite(fid,Q,['bit' num2str(nb(nbs))]);
176 -      end
177 -      for nbs = 3:NC
178 -          Q = RC_coding(k(nbs),nbs);
179 -          fwrite(fid,Q,['bit' num2str(nb(nbs))]);
180 -      end
```

Para ellos, se emplea la función "**RC_coding.m**", que implementa todas las tablas del estándar FS 1015 para la codificación de los coeficientes de reflexión.

6. Cálculo del error de predicción

El error de predicción se calcula en "**encoder_FS1015.m**":

```
128        %------------------------------------------------------------------------
129 -      xp=0;
130 -      [predErr]=filtro_error_prediction(x,xp,NC,h,N); % Error de predicción
131        %------------------------------------------------------------------------
```

Se emplea la función "**filtro_error_prediction.m**".

7. Estimación del pitch

La estimación del pitch se calcula solamente si el clasificador determina que el bloque es de tipo vocal, en "**encoder_FS1015.m**":

```
133 -       if sel == 1
134 -          pitch1=estimacion_pitch(predErr',N); %Calculo del pitch error
135            %%%%%%%%% almacenar pitch period codificado %%%%%%%%%%%%%%%%%%
136            %pitch_encoded = pitch_encoding(pitch1);
137            %fwrite(fid,pitch_encoded,'ubit7');
138            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
139 -          frameBound = floor(N/pitch1)*pitch1;
140 -          predErr = predErr(1:frameBound);
141 -          potencia = sum(predErr*predErr')/frameBound;
142 -       else
143            % SOBRE EL PITCH: "For error protection purposes, unvoiced frames shall be coded as
144            % seven ZERO bits"
145            %fwrite(fid,0,'ubit7');
146 -          potencia = sum(predErr*predErr')/N;
147 -       end
```

Para este fin, se utiliza la función "**estimacion_pitch.m**".

8. Codificación y cuantización del pitch

La codificación y cuantización del pitch se ejecuta en "**encoder_FS1015.m**":

```
154 -       if sel == 0
155 -           fwrite(fid,0,'ubit7');
156 -       else
157 -           pitch_encoded = pitch_encoding(pitch1);
158 -           fwrite(fid,pitch_encoded,'ubit7');
159 -       end
```

Si el clasificador determinó que el bloque es no vocal, se codifica un 0 con 7 bits. Caso contrario, se emplea la función "**pitch_encoding.m**", que implementa todas las tablas del FS 1015 para codificar el pitch.

9. Estimación de la potencia

La potencia se calcula seguidamente del pitch, en "**encoder_FS1015.m**":

```
133 -       if sel == 1
134 -          pitch1=estimacion_pitch(predErr',N); %Calculo del pitch error
135            %%%%%%%%% almacenar pitch period codificado %%%%%%%%%%%%%%%%%
136            %pitch_encoded = pitch_encoding(pitch1);
137            %fwrite(fid,pitch_encoded,'ubit7');
138            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
139 -          frameBound = floor(N/pitch1)*pitch1;
140 -          predErr = predErr(1:frameBound);
141 -          potencia = sum(predErr*predErr')/frameBound;
142 -       else
143            % SOBRE EL PITCH: "For error protection purposes, unvoiced frames shall be coded as
144            % seven ZERO bits"
145            %fwrite(fid,0,'ubit7');
146 -          potencia = sum(predErr*predErr')/N;
147 -       end
```

La fórmula para el cálculo de la potencia varía según el bloque sea vocal o no vocal.

10. Codificación y cuantización de la potencia

Finalmente, la potencia se codifica y cuantiza en "**encoder_FS1015.m**":

```
161          % luego la potencia (normalizada de 0 a 511
162          %fwrite(fid,potencia,'float');
163 -        power_encoded = power_encoding(potencia*511);
164          %disp(power_encoded);
165          %disp(power_encoded);
166 -        fwrite(fid,power_encoded,'ubit5');
```

Para ello, se emplea la función "**power_encoding.m**", que implementa la tabla del FS 1015 para codificar la potencia, normalizada entre 0 y 511.

5.3.3 Decoder FS-1015

El proceso de decodificación lee las tramas de señal codificadas en el encoder. Todo el proceso descrito a continuación se realiza en la función "**decoder_FS1015.m**".

Primero, se lee el pitch:

```
36      % decodificar bloque
37      % primero, el pitch
38 -    pitch_encoded=fread(fid,1,'ubit7');  %% <=== 3° TERCERO Recuperar el Pitch
39 -    if pitch_encoded == 0
40 -        sel = 0;
41 -    else
42 -        sel = 1;
43 -        pitch1 = pitch_decoding(pitch_encoded);
44 -    end
```

En el decoder, se utilizan las funciones "decoding" con las tablas inversas para decodificar los valores. En este caso, se usa la función "**pitch_decoding.m**", con dichas tablas del estándar FS 1015 implementadas. Al momento de leer el pitch, se conoce si el bloque corresponde a vocal (pitch distinto a 0) o no vocal (pitch igual a 0), lo cual se almacena en la variable "sel".

Luego, se realiza lo mismo para la potencia, con la función "**power_decoding.m**", donde se re-escala de 0 a 1 tras decodificar (recordar que se escaló de 0 a 511 antes de codificar en el encoder).

```
46      % luego, la potencia
47      %pot=fread(fid,1,'float');  %% <=== 4° CUARTO Extracción de la potencia
48 -    power_encoded = fread(fid,1,'ubit5');
49 -    pot = power_decoding(power_encoded)/511;
```

Posteriormente, se leen los coeficientes de reflexión y genera el tren de

impulsos (caso bloque vocal) o ruido blanco (caso bloque no vocal).

Caso bloque vocal (sel = 1):

```
51 -    if sel==1 % Parámetro Codec LPC 1 === VOICE
52 -        NC = 10;
53          % 10 coeficientes cuando es voz
54 -        for nbs = 1:NC
55 -            Q=fread(fid,1,['bit' num2str(nb(nbs))]);
56 -            ks(nbs) = RC_decoding(Q,nbs);
57 -        end
58
59 -        Tren=zeros(1,N); %Vector de ceros
60          %pitch_encoded=fread(fid,1,'ubit7');  %% <=== 3° TERCERO Recuperar el Pitch
61          %pitch1 = pitch_decoding(pitch_encoded);
62
63 -        if prevsel == 1
64 -            Tren(prevpitch:pitch1:N) = 1;
65 -        else
66 -            Tren(1:pitch1:N)=1;  % Generación del tren de impulsos
67 -        end
68
69 -        senalx=Tren;
70 -        prevpitch = pitch1;
```

Caso bloque no vocal (sel = 0):

```
72 -        else  % Detección UNVOICE
73 -            NC = 4;
74
75              % 4 coeficientes cuando es voz
76 -            for nbs = 1:NC
77 -                Q=fread(fid,1,['bit' num2str(nb(nbs))]);
78 -                ks(nbs) = RC_decoding(Q,nbs);
79 -            end
80 -            ks = ks(1:NC);
81
82 -            senalx= randn(1,N);
83
84              % caso no voz: leer bits control de error
85 -            errocont = fread(fid,1,'ubit20');
86 -        end
87 -        prevsel = sel;
```

En cualquiera de los dos casos, se emplea la función "**RC_decoding.m**" para decodificar los coeficientes de reflexión.

Luego, se convierten los coeficientes de reflexión en coeficientes de predicción lineal, para sintetizar la voz:

```
89        % bit de sincronizacion
90 -      sinc = fread(fid,1,'ubit1');
91
92        % decodificar al valor del coeficient
93        %%% convertir LAR a reflection coefficient
94 -      ks(1) = (10^ks(1) - 1)/(10^ks(1) + 1);
95 -      ks(2) = (10^ks(2) - 1)/(10^ks(2) + 1);
96
97        %%% convertir reflection coefficients a LP coefficients
98 -      a = zeros(NC,NC);
99 -      for i=1:NC
100 -         a(i,i)=ks(i);
101 -         j=1;
102 -         while j<i
103 -               a(i,j)=a(i-1,j)-(ks(i)*a(i-1,i-j));     %%%
104 -               j=j+1;
105 -         end
106 -     end
107 -     hs = a(NC,:)';
```

Finalmente, se sintetiza la voz mediante un tren de impulso (caso vocal) o ruido blanco (caso no vocal), con una potencia determinada según lo decodificado del bloque:

```
111 -       if sel==1
112 -           ganancia = sqrt(pitch1*pot);
113 -       else
114 -           ganancia = sqrt(pot);
115 -       end
116
117 -       senalx=senalx*ganancia;
118 -       xp=0;
119 -       e=[]; %
120 -       senalp=[];
121 -       pfd=zeros(1,NC); % Buffer para aplicar el filtraje
122 -       salida_filtro = 0;
123
124 -       for nn=1:N % N=240  % Filtro de sintesis LPC
125 -           xp = senalx(nn) + salida_filtro;
126 -           pfd = [xp pfd(1:NC-1)];
127 -           salida_filtro = pfd*hs;
128 -           senalp = [senalp xp];
129 -       end
130
131 -       y = senalp - mean(senalp);
132
133         %-------------------------------------------------
134         % de-enfasis
```

Capítulo VI: Análisis de los resultados

6.1 Descripción de las acciones previas para la obtención de los resultados

Se utilizó el software MATLAB para la simulación para el procesamiento digital de la señal de voz basado en la codificación LPC. MATLAB es un lenguaje de alto nivel y de ambiente interactivo que permite realizar tareas intensas y con una mayor velocidad que los lenguajes de programación comúnmente usados. El lenguaje está construido por códigos M (M-code), que pueden crear funciones fácilmente ejecutados en la ventana de comandos.

Pero la razón principal para la elección de este lenguaje de programación son las herramientas que proporciona para el procesamiento de señales, y el conjunto de funciones para el procesamiento digital.

Se requiere generar una base de datos, que servirá para efectuar las pruebas de clasificación de voz y para evaluar la métrica del grado de diferencia objetiva (ODG).

Se desarrolló el **script "genera_BD.m".** Primero, este script lee todos los archivos de voz (formato .wav) en la carpeta "principal_originales". Si los archivos tienen frecuencia de muestreo de 48 KHz, el script primero los sub-muestrea a 8 KHz, que es la frecuencia de muestreo válida para el codec FS 1015. Luego, el script codifica cada archivo en la carpeta "principal_originales" con encoder_FS1015() y almacena el archivo codificado en la carpeta "principal_codificadas".

Finalmente, el script decodificado el archivo codificado con decoder_FS1015() y guarda el archivo de voz (formato .wav) sintetizada en la carpeta "principal_sinteticas". Asimismo, el script necesita la RUTA/DIRECCION

del clasificador de voz/no voz, que es el archivo "clasifica_voz.mat".

Se estableció un clasificador de voz/no voz, utilizando el script "**pruebas_clasificador_**vozNovoz**.m**". Dicho script utiliza una base de datos donde se ha etiquetado manualmente cerca de 800 bloques de voz, según se haya determinado si el bloque es vocal o no vocal. Las etiquetas de los bloques se almacenan en archivos en formato *csv*, donde cada archivo *csv* contiene todas las etiquetas de todos los bloques de un archivo de voz *wav*.

Para evaluar el vocoder, se obtiene una señal de muestra y se sacan las características de esta señal para ser comparada con cada una de las características almacenadas en la base de datos.

Luego, al ejecutar los scripts "encoder_FS1015.m" y "decoder_FS1015.m", se obtiene los diferentes parámetros del vocoder.

Para la evaluación de la calidad de la señal sintetizada, se utilizó el algoritmo de evaluación del grado de diferencia objetiva (ODG), que es una métrica para evaluar calidad de audio que se obtiene mediante una prueba objetiva conocida como PEAQ (Perceptual Evaluation of Audio Quality – evaluación perceptual de calidad de audio).

6.2 Evaluación de los resultados obtenidos de la propuesta desarrollada

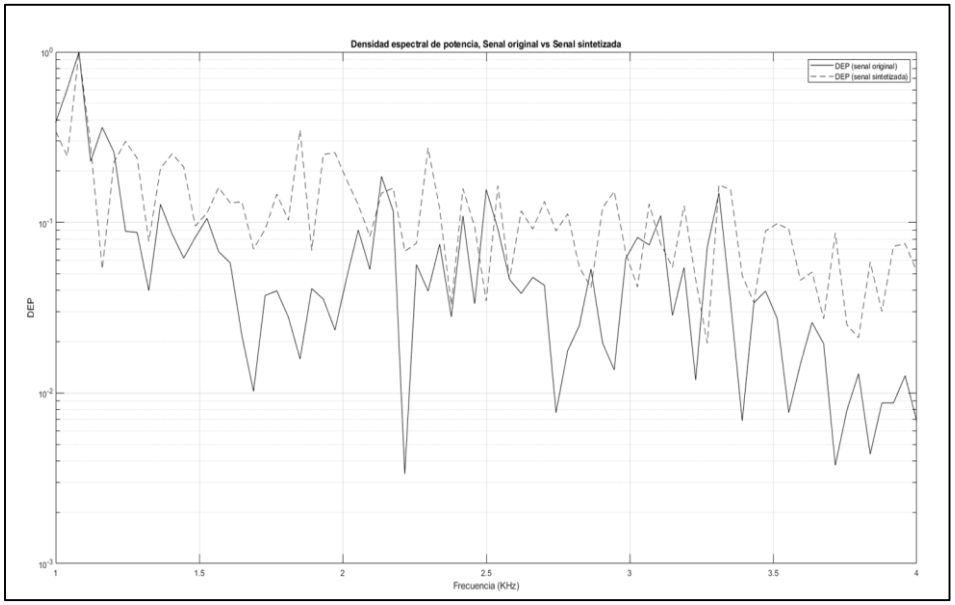
6.2.1 Densidad espectral de potencia (DEP)

Se tomaron dos muestras de señales de la base de datos, tal como se muestran en las figuras 22 y 23.

Dichas figuras muestran una comparación entre la DEP de una señal de voz original (línea sólida), y la DEP de dicha señal tras codificar y sintetizar (línea punteada). Si bien las DEP sintetizadas no son exactamente iguales que las originales, se resalta que guardan un patrón similar, sobre todo en las frecuencias más bajas.

Figura 22

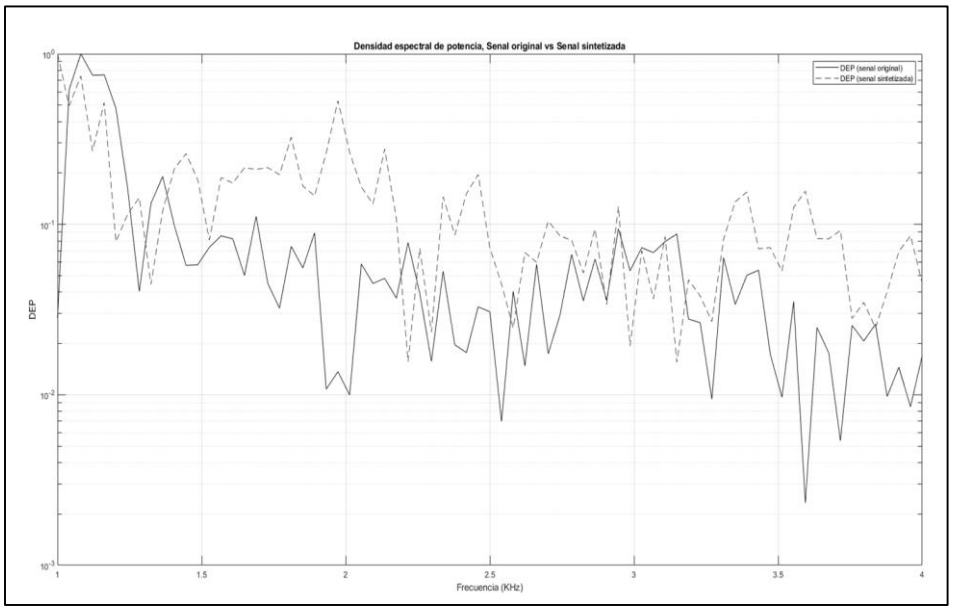
Comparación de la densidad espectral de potencia, muestra 1



Nota. Señal original (línea sólida). Señal sintetizada (línea punteada)

Figura 23

Comparación de la densidad espectral de potencia, muestra 2



Nota. Señal original (línea sólida). Señal sintetizada (línea punteada)

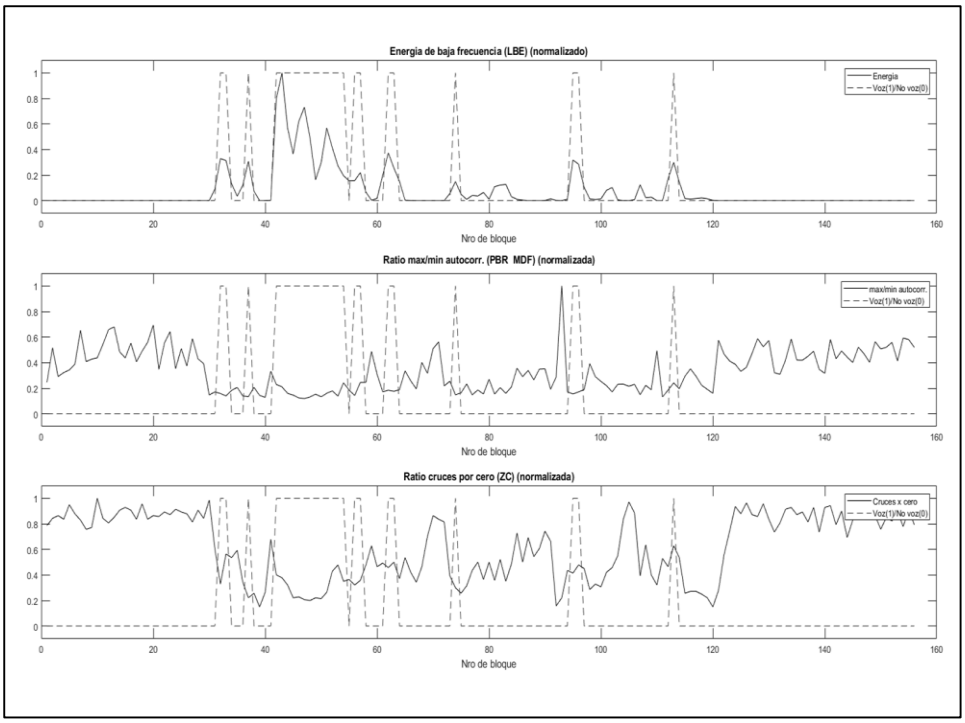
6.2.2 Parámetros para la clasificación de la voz

Se tomaron dos bloques de señales de la base de datos, tal como se muestran en las figuras 24 y 25.

Estas figuras muestran cómo varían los parámetros que se utilizaron como entrada al clasificador estadístico de bloques, para determinar si un bloque es de voz o no voz. Asimismo, se muestra el valor que toma la salida del clasificador en distintos momentos de tiempo. Se puede apreciar que el clasificador determina que un bloque es de voz (línea punteada, valor 1) cuando LBE es más alto y ZC y PBR son más bajos; lo opuesto se puede afirmar cuando el clasificador determina que el bloque es de no voz (línea punteada 0).

Figura 24

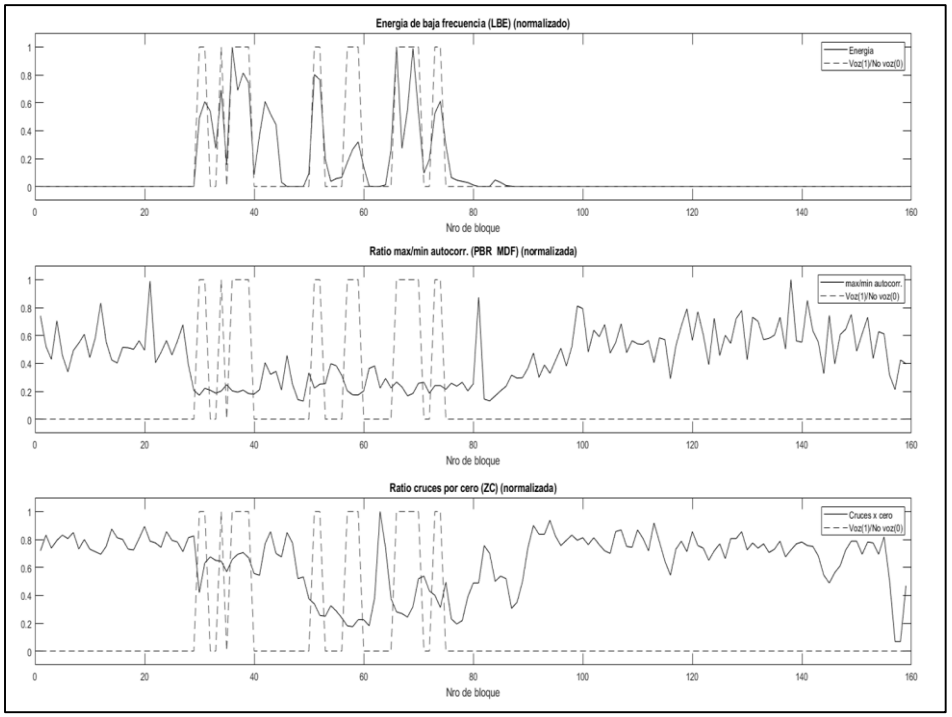
Parámetros para la clasificación de voz, bloque 1.



Nota. Línea punteada valor 1= voz. Línea punteada valor 0 = no voz.

Figura 25

Parámetros para la clasificación de voz, bloque 2



Nota. Línea punteada valor 1= voz. Línea punteada valor 0 = no voz.

6.2.3 Estimación del Pitch y del logaritmo de la potencia

Se tomaron dos muestras de señales de la base de datos, tal como se muestran en las figuras 26 y 27.

Estas figuras ilustran 2 ejemplos de cómo varía el pitch y el logaritmo de la potencia de distintos bloques de señal durante el proceso de codificación. En el caso del pitch, un valor de 0 indica que el bloque corresponde a un bloque no vocal. También se observa que todos los valores de pitch están entre 30 y 80 (muestras). Por otro lado, la potencia es muy pequeña para los casos en los que el pitch es 0 (es decir, para los bloques no vocales).

Figura 26

Estimación del Pitch y del logaritmo de la potencia, muestra 1

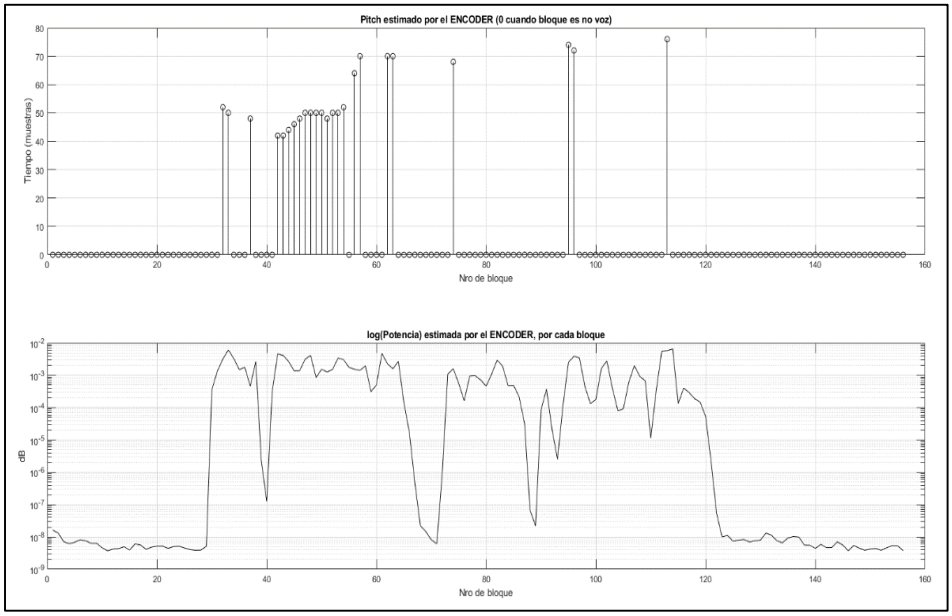
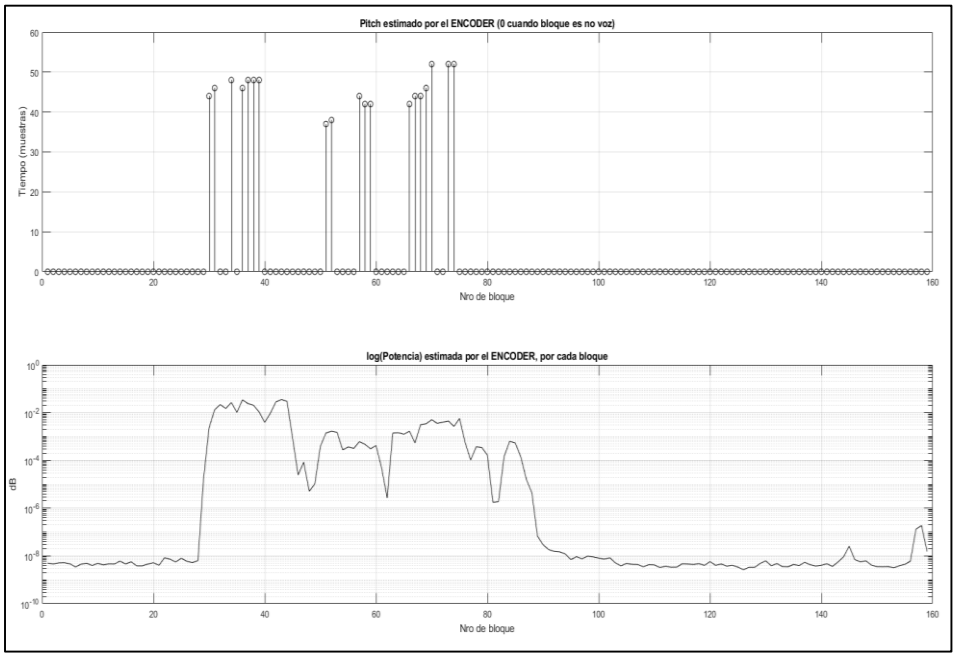


Figura 27

Estimación del Pitch y del logaritmo de la potencia, muestra 2



6.2.4 Tramas de señal original y señal sintetizada

Se tomaron dos muestras de señales de la base de datos, cada uno con su señal original y su señal sintetizada, tal como se muestran en las figuras 28 y 29.

Ambas figuras ilustran ejemplos de 2 archivos de voz que se han codificado y sintetizado. En ambas muestras, la primera línea corresponde a un segmento de silencio en la señal. Se observa que la señal sintetizada (tono claro) no alcanza un valor de 0; sin embargo, sí tiene un valor muy pequeño y tiene mucha semejanza a ruido. Luego, la segunda línea corresponde a un segmento de señal donde se ve la transición de bloque de ruido a señal de voz (tono oscuro). Se puede apreciar que las partes donde el códec detectó voz tienen señales sintetizadas periódicas, resultantes de excitar el filtro de predicción con un tren de impulsos con el pitch estimado en el encoder. Finalmente, la tercera línea muestra un segmento de señal puramente de voz.

Figura 28

Tramas de señal original y señal sintetizada, muestra 1.

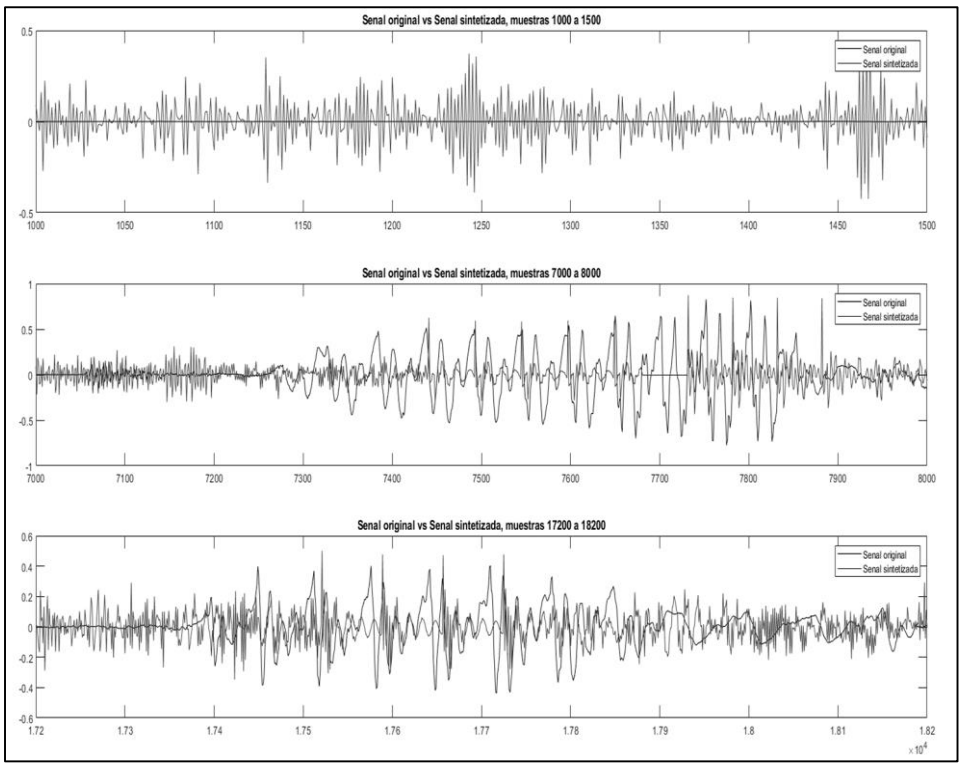
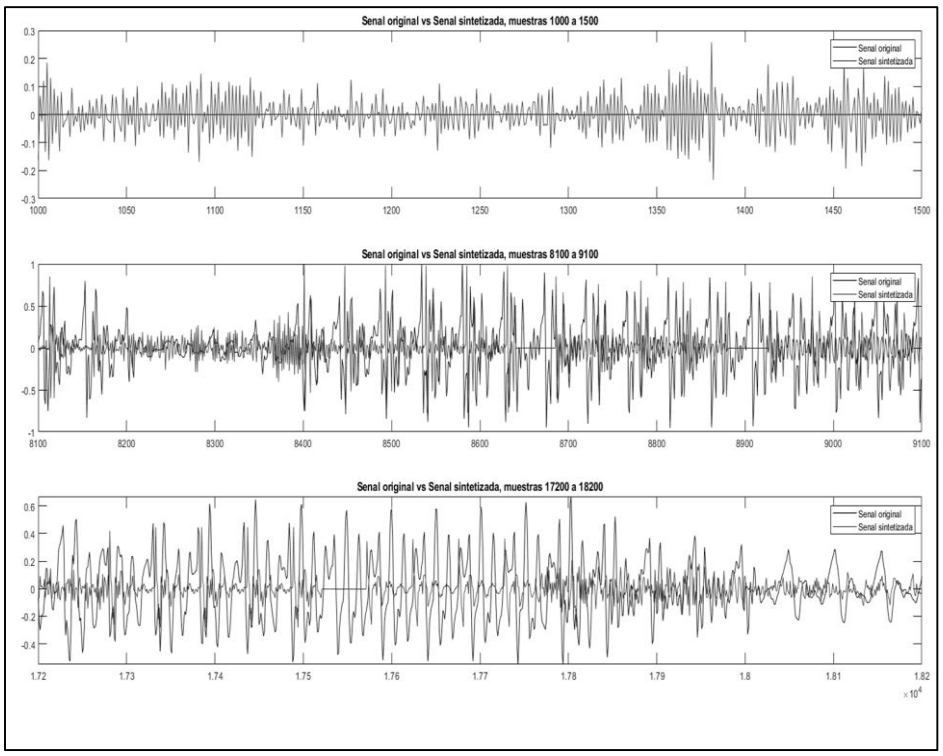


Figura 29

Tramas de señal original y señal sintetizada, muestra 2



6.2.5 Comparación del error de predicción con la señal original

Se tomaron dos muestras de señales de la base de datos, cada una con su señal de error de predicción, tal como se muestran en las figuras 30 y 31.

Como se puede observar en ambas figuras, se efectuó una comparación en el dominio del tiempo de las señales que intervinieron y se pudo verificar que las formas de las señales de error de predicción tienen muchas similitudes con la forma de onda de las señales originales. Por ello, el error de predicción se utiliza para calcular la potencia del bloque y, si se trata de un bloque tipo voz, para estimar el pitch.

Figura 30

Comparación del error de predicción con la señal original, muestra 1

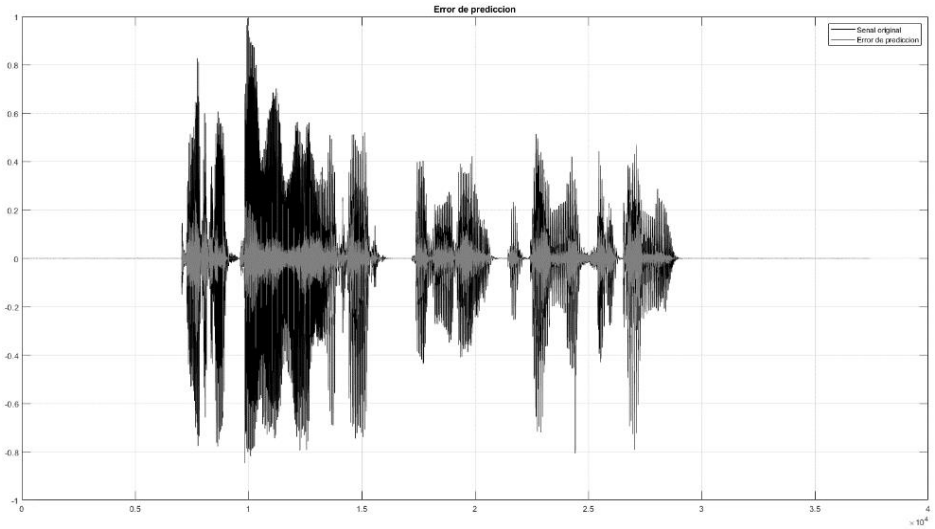
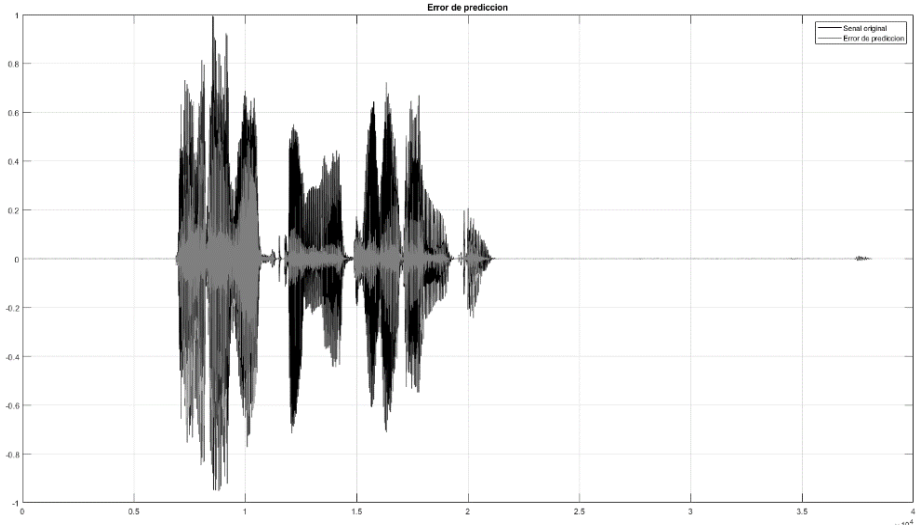


Figura 31

Comparación del error de predicción con la señal original, muestra 2



6.3 Evaluación de la calidad de la señal sintetizada

Para la evaluación de la calidad de la señal sintetizada, se utilizó la métrica del grado de diferencia objetiva (ODG).

El grado de diferencia objetiva (ODG) es una métrica para evaluar calidad de audio que se obtiene mediante una prueba objetiva conocida como PEAQ (Perceptual Evaluation of Audio Quality – evaluación perceptual de calidad de audio). Se encuentra regulada en el estándar ITU-R BS.1387 (Abanto, Kemper y Telles, 2011).

El PEAQ toma una señal de referencia, que suele ser la señal original y lo compara con la señal de prueba (señal sintetizada) que se obtiene a partir de un codificador de voz.

En el programa de mi proyecto, se generó el script **"eval_odg.m".** Este script debe estar en una sola carpeta que contenga los siguientes archivos y carpetas, como se muestra en la figura 32.

Figura 32

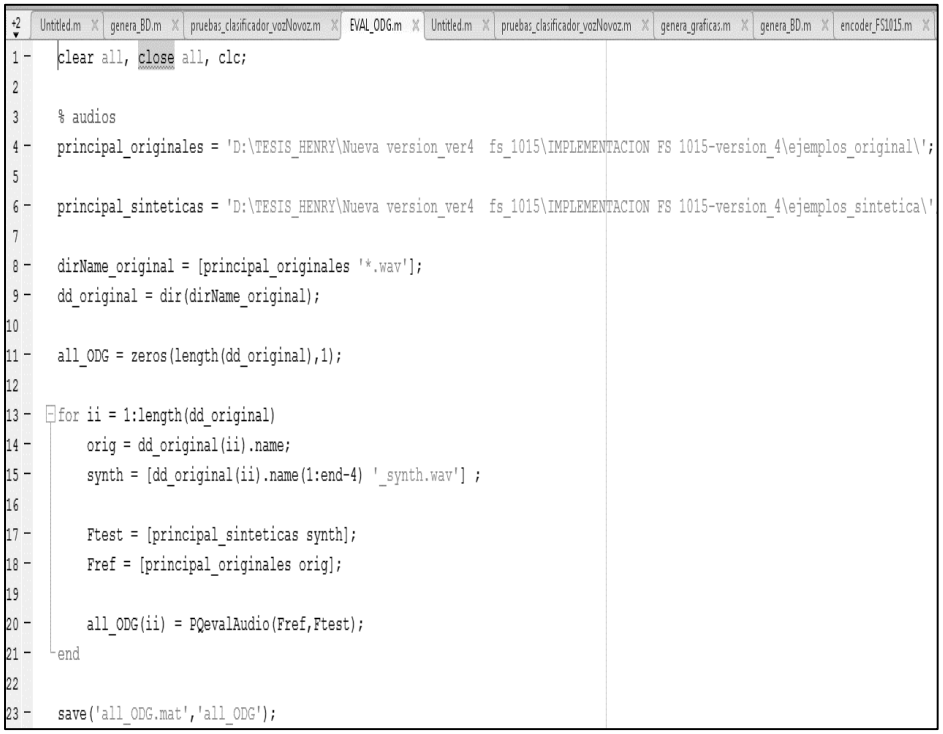
Generación de la métrica ODG

Este script toma los archivos de voz en la carpeta "principales_originales" y los archivos de voz sintetizad en "principales_sinteticas" y calcula la métrica de calidad ODG entre cada par de archivos.

Finalmente, la variable "all_ODG" contendrá los valores de la métrica para todos los archivos evaluados. Adicionalmente, el script generará un archivo "all_ODG.mat" donde también están almacenados los valores de las métricas.

Figura 33

Generación del script "eval_odg.m"



```
clear all, close all, clc;

% audios
principal_originales = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_original\';

principal_sinteticas = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_sintetica\'

dirName_original = [principal_originales '*.wav'];
dd_original = dir(dirName_original);

all_ODG = zeros(length(dd_original),1);

for ii = 1:length(dd_original)
    orig = dd_original(ii).name;
    synth = [dd_original(ii).name(1:end-4) '_synth.wav'] ;

    Ftest = [principal_sinteticas synth];
    Fref = [principal_originales orig];

    all_ODG(ii) = PQevalAudio(Fref,Ftest);
end

save('all_ODG.mat','all_ODG');
```

A continuación, se muestra la tabla 3 que contiene los valores de la métrica obtenidos para los 20 archivos evaluados.

Tabla 3

Valores obtenidos de la métrica ODG

Nombre del archive de voz	ODG	Tasa de bits (kbps)
pef_03034_01521330188.wav	-3.9128	2.4
pef_03034_01533786734.wav	-3.9120	
pef_03034_01534995269.wav	-3.9131	
pef_03034_01541948375.wav	-3.9123	
pef_03034_01578081081.wav	-3.9131	
pef_03034_01600357878.wav	-3.9132	
pef_03034_01624809523.wav	-3.9130	

pef_03034_01630186733.wav	-3.9131	
pef_03034_01650772637.wav	-3.9129	
pef_03034_01692571692.wav	-3.9130	
pem_07508_02057225111.wav	-3.9129	
pem_07508_02059441938.wav	-3.9128	
pem_07508_02105130751.wav	-3.9126	
pem_07508_02120370352.wav	-3.9126	
pem_08421_00004077415.wav	-3.9125	
pem_08421_00033639148.wav	-3.9127	
pem_08421_00033745102.wav	-3.9130	
pem_08421_00037234457.wav	-3.9129	
pem_08421_00039444878.wav	-3.9130	
pem_08421_00059285047.wav	-3.9128	

6.4 Verificación de las hipótesis de investigación

6.4.1 Hipótesis específicas

H1: "La descripción de los principales parámetros del modelo de Predicción Lineal (LP) aplicados a las señales de voz, facilitará el modelamiento de un codificador de voz (vocoder)".

Comprobación

De acuerdo a lo presentado en el ítem 5.1, referente a la descripción de la propuesta de desarrollo de un vocoder LPC, se pudo conocer la incidencia de los principales parámetros en el modelamiento de codificadores de voz.

H2: "El análisis de los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, permitirá una mejor comprensión del procesamiento digital de la voz".

Comprobación

El evaluar la incidencia de los parámetros desarrollados en el ítem 6.2, así como de la calidad de una señal de voz sintetizada mediante la métrica ODG, permitió una mejor comprensión de la importancia que tiene el procesamiento digital de señales, en el desarrollo de nuevas aplicaciones para el reconocimiento de la voz.

H3: "La simulación en la plataforma de desarrollo MATLAB representa una herramienta de gran ayuda para el procesamiento digital de la voz, el análisis de la predicción lineal y síntesis de la voz a partir de los parámetros del modelo".

Comprobación

La razón principal para la elección del lenguaje de programación MATLAB, fueron las herramientas que proporciona para el procesamiento de señales, y el conjunto de funciones para el procesamiento digital, lo que facilitó el desarrollo del vocoder LPC del estándar FS-1015, tal como lo demuestro en el desarrollo del programa, al crear funciones fácilmente ejecutados en la ventana de comandos.

6.4.2 Hipótesis general

H: "La parametrización de la señal de voz mediante el uso del modelo de predicción lineal (LP) basados en el estándar FS-1015, permite desarrollar codificadores de voz (vocoders) con una baja tasa de bits y de baja calidad".

Comprobación

Haciendo uso de los valores obtenidos de la métrica ODG de la tabla 4, se observó que las 20 muestras de las señales sintetizadas se transmiten a una velocidad de 2.4 kbps y que la métrica del grado de diferencia objetiva (ODG) de dichas señales tiene un valor promedio de -3.9.

Según el estándar internacional ITU-R BS.1387, los valores e

interpretación de los ODG son mostrados en la tabla 4.

Tabla 4

Valores e interpretación de los ODG

Valor	Interpretación
0.0	Deterioro imperceptible
-1.0	Deterioro perceptible, pero no molesto
-2.0	Deterioro ligeramente molesto
-3.0	Deterioro molesto
-4.0	Deterioro muy molesto

Nota. Recuperada de Abanto, Kemper y Telles (2011).

De la tabla 4, se obtiene que los vocoder LPC del estándar FS-1015 operan a la velocidad de transmisión de 2.4 kbit/seg, considerada una tasa de transmisión muy baja.

Con los valores obtenidos de la métrica de los ODG que se encuentran en la tabla 4 y con la interpretación de dichos valores según la tabla 5, podemos afirmar, que el vocoder desarrollado en este proyecto es de baja calidad, tal como se estipula para vocoders LPC del estándar FS-1015.

Capítulo VII: Conclusiones y recomendaciones

7.1 Conclusiones

Primera

Utilizando MATLAB, se logró implementar exitosamente un códec de voz según el estándar FS 1015. Se implementó tanto el codificador como el decodificador. El encoder procesa archivos de voz en formato WAV, 8 bits/muestra y frecuencia de muestreo 8 KHz; y devuelve archivos formato FS1015, a una tasa de 2.4 kbps. El decoder procesa archivos FS1015 y devuelve archivos WAV a 8 KHz.

Segunda

Como parte del codificador, se logró implementar exitosamente un clasificador de bloques de voz (vocal/no vocal) en base a un algoritmo estadístico de clasificación lineal. Dicho clasificador no requiere fijar un parámetro específico a un valor (por ejemplo, que el número de cruces por cero en un bloque sea mayor a 100) para clasificar el bloque, sino que se entrenó el algoritmo de máquina de soporte vectorial con bloques de voz etiquetados manualmente. Esto permite que ese único clasificador sea adaptable a diversas voces.

Tercera

Cuarta

Este trabajo implementa y muestra exitosamente el uso de la métrica de ODG para la evaluación de un códec paramétrico de baja calidad y tasa. La principal ventaja de una métrica objetiva como ODG, versus métricas subjetivas como MOS, es que el ODG se puede calcular de manera automática con un algoritmo, mientras que las métricas subjetivas requieren la participación de varias personas que atiendan y evalúen los audios

original y codificado.

7.2 Recomendaciones o propuesta

Primera

El códec del estándar FS-1015 exige que el filtro de predicción lineal tenga 10 coeficientes para los bloques vocales, y 4 coeficientes para los bloques no vocales. Se recomienda para futuras investigaciones que se incremente una mayor cantidad de coeficientes (orden del filtro), lo cual significaría que la predicción sería de mejor nivel y por ende, la calidad d la señal de voz sintetizada.

Segunda

Si bien se empleó un clasificador lineal para la etapa de clasificación de bloques, existen otras técnicas de clasificación no lineales que pueden ser más poderosas y que no necesariamente consumen mayor cantidad de recursos computacionales. Se recomienda probar con algunas arquitecturas de clasificación no lineal para mejorar la clasificación de bloques.

Tercera

Se recomienda implementar el códec en un sistema embebido o FPGA para poder llevarlo a aplicaciones en tiempo real, ya que este trabajo sólo considera aplicaciones “off-line”.

.

Referências bibliográficas

Chu, W. (2003*). Speech coding algorithms: Foundation and Evolution of Standardized Coders.* California, CA, USA: John Wiley & Sons, Inc., Publication.

Gallego, E., Correa, A., Blanco, A., Cuador, J., y Álvarez, R. (2017). La predicción lineal aplicada al reconocimiento distribuido del habla en redes IP. *Computación y Sistemas-Red Académica SCIELO*, 21(1), 137–146.
Recuperado desde: http://www.scielo.org.mx/pdf/cys/v21n1/1405-5546-cys-21-01-00137.pdf

García, A., Gallego, E., Dominguez, I., Correa, A., y Rodríguez, J. (2011). Codificación de voz mediante coeficientes de predicción lineal (LPC) sobre Microblaze. *Ciencia y Tecnología*, 13(15), 53–60.
Recuperado desde: https://rc.upr.edu.cu/jspui/bitstream/DICT/2822/1/1_Codificaci%c3%b3n%20de%20voz%20mediante%20coeficientes%20de%20PL.pdf

García, M., y Rodríguez, J. (2002)**.** *Diseño e implementación de un vocoder hardware digital* (Tesis de Titulo). Universidad San Buenaventura, Bogotá, Colombia. Recuperado desde: http://biblioteca.usbbog.edu.co:8080/Biblioteca/BDigital/43266.pdf

Hernández, A. (2005). *Estudio y simulación de un codificador de voz basado en la recomendación G.729 de la ITU-T* (Proyecto fin de carrera). Universidad de Sevilla, Sevilla, España. Recuperado desde: http://bibing.us.es/proyectos/abreproy/11154/fichero/Memoria%252Fpdf%252Fmemoria.pdf

Ingle, V., y Proakis, J. (2012). *Digital Signal Processing Using MATLAB*. Connecticut, CT, USA: Northeastern University, third edition.

Mitra, S. (2010). *Digital Signal Processing: A Computer-Based Approach*. New York, NY, USA: McGraw-Hill, fourth edition.

Montenegro, P., y Mora, V. (2007). *Análisis y evaluación para la selección de códecs de VoIP* (Trabajo de Graduación). Universidad de Azuay, Cuenca, Ecuador. Recuperado desde: http://dspace.uazuay.edu.ec/bitstream/datos/2271/1/05787.pdf

Orellana, A. (2002)**.** *Análisis y simulación de compresión de voz utilizando* **códigos** *de predicción lineal* (Tesis de Titulo). Escuela Politécnica Nacional, Quito, Ecuador. Recuperado desde: https://bibdigital.epn.edu.ec/bitstream/15000/10909/1/T2057.pdf

Ortiz, J. (2006). *Determinación de coeficientes lpc en tiempo real utilizando un DSP de TI, para señales de voz* (Proyecto fin de carrera). Universidad Autónoma Metropolitana, México DF, México.

Recuperado desde: http://148.206.53.84/tesiuami/UAMI12902.pdf

Valdivia, R. (2009). *Elaborando la tesis: una propuesta*, Tacna, Perú: Fondo Editorial Universidad Privada de Tacna.

Vera, P. (2006). *Desarrollo de técnicas de codificación de audio basadas en modelos de señal paramétricos* (Tesis Doctoral). Universidad de Alcalá, Madrid, España. Recuperado desde: https:// http://www4.ujaen.es/~pvera/tesis.pdf

ANEXO 1

Matriz de consistencia

MATRIZ DE CONTINGENCIA – PROYECTO DE INVESTIGACIÓN

TÍTULO DE PROYECTO: **"INFLUENCIA DE LOS HIDROMETEOROS SOBRE LOS ENLACES ÓPTICOS NO GUIADOS, EN LAS ZONAS RURALES DE LA REGIÓN TACNA EN EL AÑO 2020"**

AUTOR : **Ing. María Elena VILDOZO ZAMBRANO**

PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLES E INDICADORES	METODOLOGÍA
1. INTERROGANTE PRINCIPAL ¿En qué medida el empleo del modelo de predicción lineal (LPC) basado en el estándar FS-1015, permitirá el desarrollo de un codificador paramétrico de voz (vocoder) a bajas tasas de transmisión de bits?	**1. OBJETIVO GENERAL** Desarrollar un codificador paramétrico de voz (vocoder) a bajas tasas de transmisión de bits, empleando el modelo de predicción lineal (LP) basado en el estándar FS-1015.	**1. HIPÓTESIS GENERAL** "La parametrización de la señal de voz mediante el uso del modelo de predicción lineal (LP) basados en el estándar FS-1015, permite desarrollar codificadores de voz (vocoders) con una baja tasa de bits".	**VARIABLE INDEPENDIENTE** **El modelo de predicción lineal (LPC)** **DIMENSIÓN 1**: Determinación de los coeficientes de predicción lineal. **Indicadores** 1. Método de la Autocorrelación **DIMENSIÓN 2:** Determinación de señales sonoras o no sonoras. **Indicadores** 1. Número de cruces por cero 2. Energía de la señal: log Es **DIMENSIÓN 3**: Estimación del pitch. **Indicadores** 1. Método de autocorrelación (ACM)	**TIPO DE INVESTIGACIÓN** Aplicada. **DISEÑO DE LA INVESTIGACIÓN** Experimental. **UNIDAD DE ESTUDIO** Comprende los codificadores que utilizan la compresión de la voz. **POBLACIÓN** No corresponde por tratarse de una investigación aplicada. **MUESTRA** No corresponde por tratarse de una investigación aplicada. **TÉCNICAS DE RECOLECCIÓN DE DATOS** Técnica de análisis documental Técnica de observación experimental de los resultados obtenidos. **INSTRUMENTOS** Fuentes secundarias**:** Libros especializados sobre procesamiento digital de señales (DSP) Catálogos técnicos sobre códecs digitales. Direcciones web sobre técnicas de procesamiento digital de señales usando MATLAB, de tratamiento de señales en tiempo discreto. Software: MATLAB y Simulinkl. Fuentes primarias Equipos audio-visuales
2. INTERROGANTES ESPECÍFICAS ¿Cómo incide en el modelamiento de un codificador paramétrico de voz, los principales parámetros de un modelo de predicción lineal basado en el estándar FS-1015?	**2. OBJETIVOS ESPECÍFICOS** Identificar los principales parámetros del modelo de predicción lineal aplicados a las señales de voz, que faciliten el modelamiento de un codificador de voz (vocoder).	**2. HIPÓTESIS ESPECÍFICAS** "Los principales parámetros del modelo de Predicción Lineal (LP) aplicados a las señales de voz, permitirá el modelamiento de un codificador de voz (vocoder)".		
¿Cómo la simulación en la plataforma de desarrollo MATLAB del proceso de compresión de voz utilizando el estándar FS-1015, permite el desarrollo de un codificador de voz (vocoder)?	Simular en la plataforma de desarrollo MATLAB el proceso de compresión de voz utilizando el estándar FS-1015, que permita el desarrollo de un codificador de voz (vocoder).	"Los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, permitirán obtener resultados del procesamiento elegido".	**VARIABLE DEPENDIENTE** **El codificador de voz (vocoder).** **DIMENSIÓN 1**: Según sus parámetros **Indicadores** 1. Velocidad de transmisión ("bit rate") 2. Calidad de la voz reconstruida. **DIMENSIÓN 2**: Según la técnica de codificación **Indicadores** 1. Codificador paramétrico.	
¿Cómo los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, determina los resultados del procesamiento elegido?	Analizar los parámetros de un codificador de voz según el modelo de predicción lineal (LP) del estándar FS-1015, que determine los resultados del procesamiento elegido.	"La simulación en la plataforma de desarrollo MATLAB del proceso de compresión de voz basados en el estándar FS-1015, permitirá el desarrollo del codificador de voz".		

ANEXO 2

Script de funciones utilizadas en el desarrollo del vocoder

FUNCIONES Y SCRIPTS

1. analisis_lpc.m

```
function  [h,k]=analisis_lpc(x,NC,N)


R=autocorrelacion(x,NC,N);
[h,k]=durbins(R,NC);
```

2. autocorrelacion.m

```
function [R]=autocorrelacion(x,nc,N)
% x: Señal del habla
%nc: número de desplazamientos, coeficientes del filtro u orden del filtro
%N: tamaño del bloque de habla
for m=0:nc
    acum=0;
    for n=0:N-1-m
        acum=acum+(x(n+1)*x(n+m+1));
    end
    R(m+1)=acum; % N+1 valores de autocorrelación
end
```

3. de_enfasis.m

```
function y=de_enfasis(x,N,alfa,m_past)

%  x: bloque de señal del habla (Speech)
% N: Tamaño de bloque <80 muestras - 240 muestras>
% alfa: factor del filtro  0<alfa<1, alfa debe ser cercano a '1'
% y: señal de salida del filtro de pre enfasis

y=[];
y(1)=1./((1./x(1)).*(1-alfa*m_past));
for n=2:N
    y(n)=1./((1./x(n)).*(1-alfa*x(n-1)));
end
```

4. Detection_voice.m

```
function [valor,cruces_por_cero,sum_magnitud]=Detection_voice(x,N,umbral_zc,umbral_sum_magnitud)
% Calculo de energía
e=sum(x.^2);
%     v=ones(1,N);
%     Y=[Y e*v];

    % Calculo del número de cruces por cero
zc=0;
for k=2:N
    zc=zc+0.5*abs(sign(x(k))-sign(x(k-1))); % zc es el número de cruces por cero por bloque de N muestras
end
sum_magnitud=e;
cruces_por_cero=zc;

if zc>umbral_zc  % Clasificación del habla por el número de cruces x cero
    valor=0; % Unvoiced
else
    valor=1; % Voiced
end
```

5. durbins.m

```
function [h,k]=durbins(R,N)
% R: funcion de autocorrelacion
% N: numero de coeficientes del filtro
E=R(1); %enrgia de laseñal siempre es el primer valor claro por que es para
%cero no hay desplaz de la señal
for i=1:N
    acum=0;
    for j=1:i-1
        acum=acum+(a(i-1,j)*R(i-j+1));
    end;
    k(i)=(R(i+1)-acum)/E;  % reflection coefficients
    a(i,i)=k(i);
    j=1;
    while j<i
          a(i,j)=a(i-1,j)-(k(i)*a(i-1,i-j));     %%%
          j=j+1;
    end;
    E=(1-(k(i)^2))*E;
end;
h=a(N,:); %coef optimos
 %en la fila N de lamatriz a todos los valores de esa fila pasalo a h
```

6. estimacion_pitch.m

```
function [pitch]=estimacion_pitch(x,N)

  % ==========Filtro de error de predicción==============
pico=0;
N=240;  % N = 80 - 240m
l=0;
AutoC=[];
% disp('Tamaño x');
% size(x)
% disp('Tamaño zeros');
% size(zeros(N,1))
s=[];
s=[zeros(N,1); x];
% disp('Tamaño s');
% size(s)

for l=20:156
    autoc=0;
    for n=N:2*N
        autoc=autoc+s(n)*s(n-l);
    end
    AutoC=[AutoC autoc];
    if autoc>pico
        pico=autoc;
        lag=l;
    end
end
% disp('El valor del pitch es:');

if (lag > 40) && (lag <= 80)
    lag = lag + mod(lag,2);
elseif (lag > 80) && (lag <= 156)
    lag = lag + (4 - mod(lag,4));
end

pitch=lag;
```

7. filtro_error_prediction.m

```
function [e]=filtro_error_prediction(x,xp,nc,ha,N)

  % ==========Filtro de error de predicción==============
    pfd=zeros(1,nc); % Buffer para aplicar el filtraje
%     xp=0;

    e=[]; %
    for n=1:N % N=240
        e(n)=x(n)-xp;  %  error (d(n)) = muestra del habla x(n) - xp (valor de predicción de x)
        y=xp+e(n);   % Actualización del valor de predicción
        pfd=[y pfd(1:nc-1)];  % Tamaño de pf = N
        xp=pfd*ha';  % Aplicando el filtro de síntesis, h: coeficientes LPC
%         XP=[XP; xp];  % Ejemplo s
%         D=[D; e(n)];
    end
```

8. pitch_decoding.m

```
function pitch = pitch_decoding(pitch_encoded)

% Funcion que devuelve el valor del pitch a partir del pitch codificado, segun la tabla del
% estandar FS1015-LPC10 (subseccion 3.4.1.)
%
% pitch: valor de pitch segun tabla, en segundos (pitch period) (20-156)
% pitch_encoded: codigo del pitch segun la tabla de la subseccion en
% mencion

tablaCod = [19, 11, 27, 25, 29, 21, 23, 22, 30, 14, 15, 7, 39, 38, 46, ...
         42, 43, 41, 45, 37, 53, 49, 51, 50, 54, 52, 60, 56, 58, 26, ...
         90, 88, 92, 84, 86, 82, 83, 81, 85, 69, 77, 73, 75, 74, 78, ...
         70, 71, 67, 99, 97, 113, 112, 114, 98, 106, 104, 108, 100, 101, 76 ];

indPitch = pitch_encoded == tablaCod;
if sum(indPitch) == 0
error('Pitch (codificado) debe estar entre los valores disponibles segun la tabla del estandar FS1015-LPC10 (subseccion 3.4.1.
end

tablaPitch = [20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40, ...
              42,44,46,48,50,52,54,56,58,60,62,64,66,68,70,72,74,76,78,80,...
              84,88,92,96,100,104,108,112,116,120,124,128,132,136,140,144,148,152,156 ];

pitch = tablaPitch(indPitch);
```

9. pitch_encoding.m

```
function pitch_encoded = pitch_encoding(pitch)

% Funcion que devuelve el valor codificado del pitch, segun la tabla del
% estandar FS1015-LPC10 (subseccion 3.4.1.)
%
% pitch: valor de pitch segun tabla, en segundos (pitch period) (20-156)
% pitch_encoded: codigo del pitch segun la tabla de la subseccion en
% mencion

if or((pitch < 20),(pitch>156))
    error('Pitch (periodo) debe estar entre 20 y 156')
end

tablaPitch = [20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40, ...
              42,44,46,48,50,52,54,56,58,60,62,64,66,68,70,72,74,76,78,80,...
              84,88,92,96,100,104,108,112,116,120,124,128,132,136,140,144,148,152,156 ];

indPitch = pitch == tablaPitch;
if sum(indPitch) == 0
error('Pitch (periodo) debe estar entre los valores disponibles segun la tabla del estandar FS1015-LPC10 (subseccion 3.4.1.)'
end
```

```
tablaCod = [19, 11, 27, 25, 29, 21, 23, 22, 30, 14, 15, 7, 39, 38, 46, ...
         42, 43, 41, 45, 37, 53, 49, 51, 50, 54, 52, 60, 56, 58, 26, ...
         90, 88, 92, 84, 86, 82, 83, 81, 85, 69, 77, 73, 75, 74, 78, ...
         70, 71, 67, 99, 97, 113, 112, 114, 98, 106, 104, 108, 100, 101, 76 ];

pitch_encoded = tablaCod(indPitch);
```

10. power_decoding.m

```
function power = power_decoding(power_encoded)

% Funcion que devuelve el valor codificado de la potencia, segun la tabla del
% estandar FS1015-LPC10 (subseccion 3.5.)
%
% power: potencia segun tabla, normalizada de 0-1 a 0-511
% power_encoded: codigo de la potencia segun la tabla de la subseccion en
% mencion

tabla = [ 0,1,2,3,4,5,6,7,8,9,11,13,16,19,23,27,32,39,46,55, ...
          66,79,94,113,135,164,192,230,275,328,392,468 ];

indice = 0:31;

power = tabla(power_encoded == indice);
```

11. power_encoding.m

```
function power_encoded = power_encoding(power)

% Funcion que devuelve el valor codificado de la potencia, segun la tabla del
% estandar FS1015-LPC10 (subseccion 3.5.)

% power: potencia segun tabla, normalizada de 0-1 a 0-511
% power_encoded: codigo de la potencia segun la tabla de la subseccion en
% mencion

tabla1 = [ 0,1,2,3,4,5,6,7,8,9,11,13,15,18,22,26,31,36,43,52,61,73, ...
           87,104,124,148,177,211,252,301,360,429];

tabla2 = [ 0,1,2,3,4,5,6,7,8,10,12,14,17,21,25,30,35,42,51,60,72,86, ...
           103,123,147,176,210,251,300,359,428,511];

indice = 0:31;

temp1 = find(power >= tabla1);
temp2 = find(power <= tabla2);
```

```

if isempty(temp2)
    power_encoded = indice(temp1(end));
elseif isempty(temp1)
    power_encoded = indice(temp2(1));
else
    power_encoded = indice(temp2(1));
end
```

12. pre_enfasis.m

```
function y=pre_enfasis(x,N,alfa,m_past)

%  x: bloque de señal del habla (Speech)
% N: Tamaño de bloque <80 muestras - 240 muestras>
% alfa: factor del filtro  0<alfa<1, alfa debe ser cercano a '1'
% y: señal de salida del filtro de pre enfasis

y=[];
y(1)=x(1)-alfa*m_past;
for n=2:N
    y(n)=x(n)-alfa*x(n-1);
end
```

13. RC_coding.m

```
function indice = RC_coding(RC_value,ncoef)

% Funcion que devuelve el valor codificado del coeficiente RC_value, segun
% la tabla del estandar FS1015-LPC10
%
% RC_value: el valor entre -1 y 1 del coeficiente a cuantizar
% ncoef: nro del coeficiente para saber que tabla usar
% indice: el valor a cuantizar, segun la tabla en el estandar FS1015-LPC10
% (subseccion 3.6.)

% la tabla es la misma para el coef. 1 y para el 2
if ncoef == 2
    ncoef = 1;
end

switch ncoef
    case 1
        % RC1 y RC2
        tabla1 = [ -1, -0.9843, -0.9687, -0.9530, -0.9374, ...
                   -0.9062, -0.8749, -0.8280,  -0.7655, -0.6874, ...
                   -0.6093, -0.5312, -0.4218, -0.3124, -0.2031, ...
                   -0.0937, 0.0938, 0.2032, 0.3125, 0.4219, 0.5313, ...
                    0.6094, 0.6875, 0.7656, 0.8281, 0.8750, 0.9063, ...
                    0.9375, 0.9531, 0.9688, 0.9844 ];
```

```
25
26 -          tabla2 = [ -0.9844, -0.9688, -0.9531, -0.9375, -0.9063, ...
27                       -0.8750, -0.8281, -0.7656,  -0.6875, -0.6094, ...
28                       -0.5313, -0.4219, -0.3125, -0.2032, -0.0938, ...
29                        0.0937, 0.2031, 0.3124, 0.4218, 0.5312, 0.6093, ...
30                        0.6874, 0.7655, 0.8280, 0.8749, 0.9062, 0.9374, ...
31                        0.9530, 0.9687, 0.9843, 1 ];
32
33 -          temp1 = find(RC_value >= tabla1);
34 -          temp2 = find(RC_value < tabla2);
35 -          indice = (0:(length(tabla1)-1))-15;
36 -      case 3
37           % RC3
38 -          tabla1 = [ -1, -0.5890, -0.5455, -0.5018, -0.4582, ...
39                       -0.4147, -0.3711, -0.3275,  -0.2839, -0.2403, ...
40                       -0.1966, -0.1531, -0.1095, -0.0659, -0.0223, ...
41                       0.0213, 0.0649, 0.1140, 0.1576, 0.2012, 0.2448, ...
42                        0.2884, 0.3319, 0.3756, 0.4192, 0.4627, 0.5063, ...
43                        0.5500, 0.5935, 0.6371, 0.6807, 0.7243 ];
```

```
44
45 -          tabla2 = [ -0.5891, -0.5456, -0.5019, -0.4583, -0.4148, ...
46                        -0.3712, -0.3276, -0.2840,  -0.2404, -0.1967, ...
47                        -0.1532, -0.1096, -0.0660, -0.0224, 0.0212, ...
48                         0.0648, 0.1139, 0.1575, 0.2011, 0.2447, 0.2883, ...
49                         0.3318, 0.3755, 0.4191, 0.4626, 0.5062, 0.5499, ...
50                         0.5934, 0.6370, 0.6807, 0.7242, 1 ];
51
52 -          temp1 = find(RC_value >= tabla1);
53 -          temp2 = find(RC_value < tabla2);
54 -          indice = (0:(length(tabla1)))-16;
55 -      case 4
56           % RC4
57 -          tabla1 = [ -1, -0.7626, -0.7236, -0.6845, -0.6454, ...
58                       -0.6064, -0.5673, -0.5282,  -0.4892, -0.4501, ...
59                       -0.4111, -0.3720, -0.3329, -0.2939, -0.2548, ...
60                       -0.2157, -0.1767, -0.1328, -0.0937, -0.0547, ...
61                       -0.0156, 0.0235, 0.0625, 0.1016, 0.1406, 0.1797, ...
62                       0.2188, 0.2578, 0.2969, 0.3360, 0.3750, 0.4141 ];
63
```

```
        tabla2 = [ -0.5891, -0.5456, -0.5019, -0.4583, -0.4148, ...
                   -0.3712, -0.3276, -0.2840,  -0.2404, -0.1967, ...
                   -0.1532, -0.1096, -0.0660, -0.0224, 0.0212, ...
                    0.0648, 0.1139, 0.1575, 0.2011, 0.2447, 0.2883, ...
                    0.3318, 0.3755, 0.4191, 0.4626, 0.5062, 0.5499, ...
                    0.5934, 0.6370, 0.6807, 0.7242, 1 ];

        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)))-16;
    case 5
        % RC5
        tabla1 = [ -1, -0.6046, -0.5323, -0.4599, -0.3876, ...
                   -0.3152, -0.2429, -0.1706,  -0.0983, -0.0213, ...
                   0.0510, 0.1233, 0.1956, 0.2680, 0.3404, 0.4127 ];

        tabla2 = [ -0.6047, -0.5324, -0.4600, -0.3877, -0.3153, ...
                   -0.2430, -0.1707, -0.0984,  -0.0214, 0.0509, ...
                    0.1232, 0.1955, 0.2679, 0.3403, 0.4126, 1 ];

        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)-1))-8;
```

```
    case 6
        % RC6
        tabla1 = [ -1, -0.7010 , -0.6327, -0.5644, -0.4961, ...
                   -0.4278, -0.3595, -0.2912,  -0.2229, -0.1504, ...
                   -0.0821, -0.0138, 0.0545, 0.1228, 0.1911, 0.2594 ];

        tabla2 = [ -0.7011, -0.6328, -0.5645, -0.4962, -0.4279, ...
                   -0.3596, -0.2913, -0.2230,  -0.1505, -0.0822, ...
                   -0.0139, 0.0544, 0.1227, 0.1910, 0.2593, 1 ];

        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)-1))-8;
    case 7
        % RC7
        tabla1 = [ -1, -0.5473 , -0.4808, -0.4144, -0.3479, ...
                   -0.2815, -0.2151, -0.1487,  -0.0823, -0.1116, ...
                   0.0548, 0.1212, 0.1876, 0.2541, 0.3205, 0.3870 ];

        tabla2 = [ -0.5474, -0.4809, -0.4145, -0.3480, -0.2816, ...
                   -0.2152, -0.1488, -0.0824,  -0.0117, 0.0547, ...
                   0.1211, 0.1875, 0.2540, 0.3204, 0.3869, 1 ];
```

```
        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)-1))-8;
    case 8
        % RC8
        tabla1 = [ -1, -0.6241 , -0.5567, -0.4894, -0.4220, ...
                   -0.3547, -0.2873, -0.2200,  -0.1526, -0.0810, ...
                   -0.0137, 0.0537, 0.1210, 0.1884, 0.2557, 0.3231 ];

        tabla2 = [ -0.6242, -0.5568, -0.4895, -0.4221, -0.3548, ...
                   -0.2874, -0.2201, -0.1527,  -0.0811, -0.0138, ...
                   0.0536, 0.1209, 0.1883, 0.2556, 0.3130, 1 ];

        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)-1))-8;
    case 9
        % RC9
        tabla1 = [ -1, -0.3076 , -0.1907, -0.0737, 0.0433, ...
                   0.1639, 0.2808, 0.3978 ];

        tabla2 = [ -0.3077, -0.1908, -0.0738, 0.0432, 0.1638, ...
                   0.2807, 0.3977, 1 ];
```

```

        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)-1))-4;
    case 10
        % RC10
        tabla1 = [ -1, -0.3116 , -0.1202, 0.0744 ];

        tabla2 = [ -0.3117, -0.1203 , 0.0743, 1 ];

        temp1 = find(RC_value >= tabla1);
        temp2 = find(RC_value < tabla2);
        indice = (0:(length(tabla1)-1))-2;
end

if isempty(temp2)
    indice = indice(temp1(end));
elseif isempty(temp1)
    indice = indice(temp2(1));
else
    indice = indice(temp2(1));
end
```

14. RC_decoding.m

```
function RC_value = RC_decoding(indice,ncoef)

% Funcion que devuelve el valor cuantizado del coeficiente, segun el nro de
% indice introducido, segun la tabla del estandar FS1015-LPC10
%
% indice: el valor de indice, segun la tabla en el estandar FS1015-LPC10
% (subseccion 3.6.)
% ncoef: nro del coeficiente para saber que tabla usar
% RC_value: valor del coeficiente cuantizado

% la tabla es la misma para el coef. 1 y para el 2
if ncoef == 2
    ncoef = 1;
end

switch ncoef
    case 1
        tablaCoefs = [ -0.9844, -0.9688, -0.9531, -0.9375, -0.9218, ...
                    -0.8906, -0.8438, -0.7812,  -0.7187, -0.6406, ...
                    -0.5625, -0.4688, -0.3593, -0.2500, -0.1406, ...
                    0.0313, 0.1406, 0.2500, 0.3593, 0.4688, 0.5625, ...
                     0.6406, 0.7187, 0.7812, 0.8438, 0.8906, 0.9218, ...
                     0.9375, 0.9531, 0.9688, 0.9844 ];

        tablaInds = (0:(length(tablaCoefs)-1))-15;

        RC_value = tablaCoefs(indice == tablaInds);
    case 3
        tablaCoefs = [ -0.6033, -0.5598, -0.5164, -0.4729, -0.4295, ...
                    -0.3860, -0.3426, -0.2991,  -0.2557, -0.2122, ...
                    -0.1687, -0.1253, -0.0818, -0.0384, 0.0051, ...
                    0.0485, 0.0920, 0.1355, 0.1789, 0.2224, 0.2658, ...
                     0.3093, 0.3527, 0.3962, 0.4396, 0.4831, 0.5266, ...
                     0.5700, 0.6135, 0.6569, 0.7004, 0.7438 ];

        tablaInds = (0:(length(tablaCoefs)))-16;

        RC_value = tablaCoefs(indice == tablaInds);
    case 4
        tablaCoefs = [ -0.7774, -0.7383, -0.6993, -0.6602, -0.6211, ...
                    -0.5821, -0.5430, -0.5040,  -0.4649, -0.4258, ...
                    -0.3868, -0.3477, -0.3086, -0.2696, -0.2305, ...
                    -0.1914, -0.1524, -0.1133, -0.0743, -0.0352, 0.0039, ...
                     0.0429, 0.0820, 0.1211, 0.1601, 0.1992, 0.2382, ...
                     0.2773, 0.3164, 0.3554, 0.3945, 0.4336 ];

        tablaInds = (0:(length(tablaCoefs)))-16;
```

```
            RC_value = tablaCoefs(indice == tablaInds);
        case 5
            tablaCoefs = [ -0.6358, -0.5635, -0.4912, -0.4190, -0.3467, ...
                       -0.2744, -0.2022, -0.1299, -0.0577, 0.0146, ...
                       0.0869, 0.1591, 0.2314, 0.3037, 0.3759, 0.4482 ];

            tablaInds = (0:(length(tablaCoefs)-1))-8;

            RC_value = tablaCoefs(indice == tablaInds);
        case 6
            tablaCoefs = [ -0.7315, -0.6631, -0.5948, -0.5264, -0.4581, ...
                       -0.3897, -0.3213, -0.2530, -0.1846, -0.1162, ...
                       -0.0479, 0.0205, 0.0888, 0.1572, 0.2256, 0.2939 ];

            tablaInds = (0:(length(tablaCoefs)-1))-8;

            RC_value = tablaCoefs(indice == tablaInds);
        case 7
            tablaCoefs = [ -0.5762, -0.5098, -0.4434, -0.3770, -0.3106, ...
                       -0.2442, -0.1778, -0.1114, -0.0450, 0.0214, ...
                       0.0878, 0.1542, 0.2206, 0.2870, 0.3534, 0.4198 ];

            tablaInds = (0:(length(tablaCoefs)-1))-8;

```

```
            RC_value = tablaCoefs(indice == tablaInds);
        case 8
            tablaCoefs = [ -0.6539, -0.5865, -0.5191, -0.4517, -0.3843, ...
                       -0.3169, -0.2496, -0.1822, -0.1148, -0.0474, ...
                       0.0200, 0.0874, 0.1548, 0.2222, 0.2895, 0.3569 ];

            tablaInds = (0:(length(tablaCoefs)-1))-8;

            RC_value = tablaCoefs(indice == tablaInds);
        case 9
            tablaCoefs = [ -0.3634, -0.2462, -0.1290, -0.0118, 0.1054, ...
                       0.2226, 0.3398, 0.0570 ];

            tablaInds = (0:(length(tablaCoefs)-1))-4;

            RC_value = tablaCoefs(indice == tablaInds);
        case 10
            tablaCoefs = [ -0.4043, -0.2129, -0.0215, 0.1699 ];

            tablaInds = (0:(length(tablaCoefs)-1))-2;

            RC_value = tablaCoefs(indice == tablaInds);
    end

```

15. encoder_FS1015.m

```
function L = encoder_FS1015(original, codificada, clasificador)

% Funcion para ejecutar el encoder segun FS1015.
% original: direccion de archivo donde se almacena el audio a codificar
% codificada: direccion de archivo donde se guardara el audio codificado
% L: valor cualqueira para que la funcion retorne algo
%
% almacena los archivos necesario para que el decoder haga su trabajo

L = 0;

%%%
% PROBANDO CLASIFICADOR LINEAL ENTRENADO: variable se llama modelo
%%%
load(clasificador);
% VARIABLE modelo: modelo entrenado para clasificar voz/no voz
% VARIABLE mean_TD: promedio para escalar la entrada al clasificador
% VARIABLE std_TD: desv estandar para escalar la entrada al clasificador

% los archivos originales estan muestreados a 48 KHz
[senal, fs] = audioread(original);

```

```
if fs > 8e3
    % bajar frecuencia de muestro a 8 KHz
    DS = fs/(8e3); % factor por el que debo bajar la frecuencia original
    senal = decimate(senal,DS); % decimacion de frecuencia para bajar a 8 KHz
    % senal = senal(1:24e3); % tomo solo las primera 24000 muestras
    %(100 bloques de 240 muestras cada uno)
    fs = fs/DS; % la nueva frecuencia de muestreo es 8 KHz

    % almacenar el archivo decimado
    audiowrite([original(1:end-4) '.wav'],senal,fs);
end

N=240;  % Tamaño del bloque, rango 80<N<240
Numblq=floor(length(senal)/N); % Cálculo del número de bloques
m=1;   % Variable para posicionamiento del inicio de cada bloque
NC=10; % Número de coeficientes LPC

senal=senal(1:Numblq*N);
senal = senal - mean(senal);
senal = senal/max(abs(senal));

fid=fopen(codificada,'w');  %% <======= Aquí se crea el archivo de grabación

```

```
%%%%%%%%% almacenar nro de bloques %%%%%%%%%%%%%%%%%
fwrite(fid,Numblq,'int32');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%% almacenar frec de muestreo %%%%%%%%%%%%%%%%%
fwrite(fid,fs,'float');
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Parámteros LPC
alfa = 0.9375;
% ver pag. 239 libro CHU para saber el nro de bits para cada
% coeficiente
nb = [5, 5, 5, 5, 4, 4, 4, 4, 3, 2];
%%%%%%%%%%%%%%%%%

% ----- Preprocesamiento
b = fir1(48,100/fs/2,'high');  % Filtro FIR pasa alta con
%frecuencia de corte de 100Hz
senal1=filter(b,1,senal);
senal=senal1;

for n=1:Numblq % Procesamiento por bloque
    x=senal(m:m+N-1); % Extracción de bloques de N muestras

    %--------------------------------------------------------------------------
    % FILTRO DE ENFASIS
    x = filter([1, -alfa],1,x);
    %--------------------------------------------------------------------------

     %--------------------------------------------------------------------------
     % SELECCION DE VOZ O NO VOZ
     % low band energy (from 0 Hz to 800 Hz)
     xFFT = fft(x,256);
     % si 128 es 4 KHz, 800 Hz es 96
     xFFT = xFFT(1:26);
     LBE = sum(xFFT.*conj(xFFT))/length(xFFT);
     % escalar para ingresar al clasificador
%      LBE = (LBE - mean_TD(1))/std_TD(1);

     % PEAK/BOTTOM RATIO OF magnitude difference function // autocorr
     PBR_MDF = xcorr(x,x);
     PBR_MDF = abs(max(PBR_MDF)/min(PBR_MDF));
     % escalar para ingresar al clasificador
%      PBR_MDF = (PBR_MDF - mean_TD(2))/std_TD(2);
```

```
91          % zero crossing rate
92 -        ZC = 0.5*sum(abs( sign(x(1:end-1)) - sign(x(2:end)) ));
93          % escalar para ingresar al clasificador
94  %         ZC = (ZC - mean_TD(3))/std_TD(3);
95
96  %         disp([LBE,PBR_MDF,ZC]);
97
98          % clasificacion de voz y no voz
99 -        sel = modelo.predict([LBE,PBR_MDF,ZC]);
100         %fwrite(fid,sel,'ubit1');
101         %-------------------------------------------------------------------------
102         % DETERMINA CUANTOS COEFICIENTES USAR CUANDO ES VOZ Y CUANDO ES NO VOZ
103
104 -       if sel == 1 % voz
105 -           NC = 10;
106             %%% usa los 10 coeficientes cuando es VOZ
107 -       else % no voz
108 -           NC = 4;
109             %%% usa 4 COEFICIENTES CUANDOES NO VOZ
110 -       end
111
```

```
112         %-------------------------------------------------------------------------
113 -       [h,k]=analisis_lpc(x,NC,N); % Análisis LPC
114         % h: coeficientes LP
115         % k: coeficientes de reflexion
116
117         %g = log10((1+k(1:2))./(1-k(1:2))); % convertir los coeficientes con la
118 -       %transformacion LAR
119 %         g = k;
120         %%% cuantizacion de coeficientes de reflexion
121
122         %for nbs = 1:2
123         %    Q = RC_coding(g(nbs),nbs);
124         %    fwrite(fid,Q,['bit' num2str(nb(nbs))]);
125         %end
126         %for nbs = 3:NC
127         %    Q = RC_coding(k(nbs),nbs);
128         %    fwrite(fid,Q,['bit' num2str(nb(nbs))]);
129         %end
130
131         %-------------------------------------------------------------------------
132 -       xp=0;
133 -       [predErr]=filtro_error_prediction(x,xp,NC,h,N); % Error de predicción
134         %-------------------------------------------------------------------------
```

```
136 -       if sel == 1
137 -          pitch1=estimacion_pitch(predErr',N); %Calculo del pitch error
138            %%%%%%%%% almacenar pitch period codificado %%%%%%%%%%%%%%%%%
139            %pitch_encoded = pitch_encoding(pitch1);
140            %fwrite(fid,pitch_encoded,'ubit7');
141            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
142 -          frameBound = floor(N/pitch1)*pitch1;
143 -          predErr = predErr(1:frameBound);
144 -          potencia = sum(predErr*predErr')/frameBound;
145 -       else
146            % SOBRE EL PITCH: "For error protection purposes, unvoiced frames
147 -          %shall be coded as
148            % seven ZERO bits"
149            %fwrite(fid,0,'ubit7');
150 -          potencia = sum(predErr*predErr')/N;
151 -       end
152
153        %%%%%%%%% EXTRACCIÓN DEL PARÁMETRO "Power index" %%%%%%%%%%%%%%%%%
154        %fwrite(fid,potencia,'float');
155
156        % Luego de hacer todo el calculo, codificar:
157        % primero el pitch
```

```
158 -      if sel == 0
159 -          fwrite(fid,0,'ubit7');
160 -      else
161 -          pitch_encoded = pitch_encoding(pitch1);
162 -          fwrite(fid,pitch_encoded,'ubit7');
163 -      end
164
165        % luego la potencia (normalizada de 0 a 511
166        %fwrite(fid,potencia,'float');
167 -      power_encoded = power_encoding(potencia*511);
168        %disp(power_encoded);
169        %disp(power_encoded);
170 -      fwrite(fid,power_encoded,'ubit5');
171
172        % finalmente los coeficientes
173 -      g = log10((1+k(1:2))./(1-k(1:2))); % convertir los coeficientes con la
174 -      %transformacion LAR
175    %     g = k;
176        %%% cuantizacion de coeficientes de reflexion
177
178 -      for nbs = 1:2
179 -          Q = RC_coding(g(nbs),nbs);
180 -          fwrite(fid,Q,['bit' num2str(nb(nbs))]);
181 -      end
```

```
182 -       for nbs = 3:NC
183 -           Q = RC_coding(k(nbs),nbs);
184 -           fwrite(fid,Q,['bit' num2str(nb(nbs))]);
185 -       end
186
187 -       if sel == 0
188             % caso UNVOICE debo llenar con control de errores
189 -           fwrite(fid,0,'ubit20');
190 -       end
191
192         % bit de sincronizacion
193 -       fwrite(fid,0,'ubit1');
194
195 -       m=m+N;
196 -   end
197 -   fclose(fid);
```

16. decoder_FS1015

```
1   function L = decoder_FS1015(codificada, sintetizada)
2
3   % Funcion para ejecutar el decoder segun FS1015.
4   % codificada: direccion de archivo donde se encuentra el audio codificado
5   % sintetizada: direccion de archivo donde se guardara el audio sintetizado
6   % en .wav
7   % L: valor cualqueira para que la funcion retorne algo
8
9   L = 0;
10
11  NC = 10;
12  N = 240;
13  Senalrecp=[];
14
15  fid = fopen(codificada,'r'); % Se abre el archivo del habla codificado
16
17  %%%%%%%%% leer nro de bloques %%%%%%%%%%%%%%%%%
18  Numblq=fread(fid,1,'int32');
19  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
20
21  %%%%%%%%% leer frec de muestreo %%%%%%%%%%%%%%%%%
22  fs=fread(fid,1,'float');
23  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
24
```

```
% -------------------------------- CODIFICADOR LPC -------------------------
%%%%%%%%%%%%%%%%%
alfa = 0.9375;
prevsel = 0;
% ver pag. 239 libro CHU para saber el nro de bits para cada
% coeficiente
nb = [5, 5, 5, 5, 4, 4, 4, 4, 3, 2];

for n=1:Numblq % Procesamiento por bloque
    ks = zeros(size(nb));

    % decodificar bloque
    % primero, el pitch
    pitch_encoded=fread(fid,1,'ubit7');  %% <=== 3° TERCERO Recuperar el Pitc
    if pitch_encoded == 0
        sel = 0;
    else
        sel = 1;
        pitch1 = pitch_decoding(pitch_encoded);
    end

    % luego, la potencia
    %pot=fread(fid,1,'float');  %% <=== 4° CUARTO Extracción de la potencia
    power_encoded = fread(fid,1,'ubit5');
    pot = power_decoding(power_encoded)/511;

    if sel==1 % Parámetro Codec LPC 1 === VOICE
        NC = 10;
        % 10 coeficientes cuando es voz
        for nbs = 1:NC
            Q=fread(fid,1,['bit' num2str(nb(nbs))]);
            ks(nbs) = RC_decoding(Q,nbs);
        end

        Tren=zeros(1,N); %Vector de ceros
        %pitch_encoded=fread(fid,1,'ubit7');  %% <=== 3° TERCERO Recuperar el
        %pitch1 = pitch_decoding(pitch_encoded);

        if prevsel == 1
            Tren(prevpitch:pitch1:N) = 1;
        else
            Tren(1:pitch1:N)=1;  % Generación del tren de impulsos
        end
```

```
        senalx=Tren;
        prevpitch = pitch1;

    else  % Detección UNVOICE
        NC = 4;

        % 4 coeficientes cuando es voz
        for nbs = 1:NC
            Q=fread(fid,1,['bit' num2str(nb(nbs))]);
            ks(nbs) = RC_decoding(Q,nbs);
        end
        ks = ks(1:NC);

        senalx= randn(1,N);

        % caso no voz: leer bits control de error
        errocont = fread(fid,1,'ubit20');
    end
    prevsel = sel;

    % bit de sincronizacion
    sinc = fread(fid,1,'ubit1');

```

```
    % decodificar al valor del coeficient
    %%% convertir LAR a reflection coefficient
    ks(1) = (10^ks(1) - 1)/(10^ks(1) + 1);
    ks(2) = (10^ks(2) - 1)/(10^ks(2) + 1);

     %%% convertir reflection coefficients a LP coefficients
     a = zeros(NC,NC);
     for i=1:NC
         a(i,i)=ks(i);
         j=1;
         while j<i
               a(i,j)=a(i-1,j)-(ks(i)*a(i-1,i-j));     %%%
               j=j+1;
         end
     end
     hs = a(NC,:)';

     % pot=fread(fid,1,'float');  %% <=== 4° CUARTO Extracción de la potencia

     if sel==1
         ganancia = sqrt(pitch1*pot);
     else
         ganancia = sqrt(pot);
     end
```

```
116
117 -       senalx=senalx*ganancia;
118 -       xp=0;
119 -       e=[]; %
120 -       senalp=[];
121 -       pfd=zeros(1,NC); % Buffer para aplicar el filtraje
122 -       salida_filtro = 0;
123
124 -       for nn=1:N % N=240  % Filtro de sintesis LPC
125 -           xp = senalx(nn) + salida_filtro;
126 -           pfd = [xp pfd(1:NC-1)];
127 -           salida_filtro = pfd*hs;
128 -           senalp = [senalp xp];
129 -       end
130
131 -       y = senalp - mean(senalp);
132
133         %-------------------------------------------------------------------------
134         % de-enfasis
135 -       y = filter(1,[1, alfa],y);
136
137 -       Senalrecp=[Senalrecp y];
138
139 -   end
```

```
140 -   fclose(fid);
141
142     % Senalrecp = Senalrecp - mean(Senalrecp);
143 -   Senalrecp = Senalrecp/max(abs(Senalrecp));
144
145     % ALMACENA AUDIO EN FORMATO WAV
146 -   audiowrite(sintetizada,Senalrecp,fs);  %% <======= Aquí se crea el
147 -   %archivo de audio sintetizado
```

17. genera_BD.m

```
Untitled.m | genera_BD.m | pruebas_clasificador_vozNovoz.m | EVAL_ODG.m | Untitled.m | pruebas_clasificador_vozNovoz.m | genera_graficas.m | genera_BD.m | encoder_FS1015.m
4 -   principal_originales = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_original\';
5
6 -   principal_codificadas = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_codificada\';
7
8 -   principal_sinteticas = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_sintetica\';
9
10 -  clasificador = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\clasifica_voz.mat';
11
12 -  dirName_original = [principal_originales '*_8KHz.wav'];
13 -  dd_original = dir(dirName_original);
14 -  if isempty(dd_original)
15 -      dirName_original = [principal_originales '*.wav'];
16 -      dd_original = dir(dirName_original);
17 -  end
18
19 -  for ii = 1:length(dd_original)
20 -      orig = dd_original(ii).name;
21 -      disp(orig)
22 -      coded = [orig(1:end-4) '.fs1015'];
23 -      synth = [dd_original(ii).name(1:end-4) '_synth.wav'] ;
24
25 -      encoder_FS1015([principal_originales, orig], [principal_codificadas, coded], clasificador);
26 -      decoder_FS1015([principal_codificadas, coded], [principal_sinteticas, synth]);
27 -  end
```

18. pruebas_clasificador_vozNovoz.m

```
clear all, close all, clc;

principal_originales = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\etiquetar_henry\';

dirName_original = [principal_originales '*.wav'];
dd_original = dir(dirName_original);

% crear varaible para guardar en archivo .mat para el entrenamiento
% se tomaron 100 frames de cada cancion, por lo que habran 100*(nro de
% canciones) filas
% columna 1 = nombre del audio
% columna 2 = nro de bloque
% columna 3 = LBE
% columna 4 = PBR_MDF
% columna 5 = ZC
% columna 6 = 1 = voz, else = no voz
cell_save_trainData = cell(100*length(dd_original), 6);
trainData = (-1e4)*ones(100*length(dd_original), 4); % empieza desde la col. 3 del archivo cell

kk = 1;

for ii = 1:length(dd_original)
```

```
    orig = dd_original(ii).name;

    % leer archivo CSV con las etiquetas
    % columna 1: nro de bloque
    % columna 2: 1 = voz, else = no voz
    csvFile = [principal_originales orig(1:end-4) ' ETIQUETAS.csv'];
    %labelMat = readmatrix(csvFile);
    labelMat = table2array(readtable(csvFile,'ReadVariableNames',0,'Delimiter',';'));
    [f,c] = size(labelMat);
    if c == 1
        labelMat = table2array(readtable(csvFile,'ReadVariableNames',0));
    end

    % los archivos originales estan muestreados a 48 KHz
    [senal, fs] = audioread([principal_originales, orig]);

    % bajar frecuencia de muestro a 8 KHz
    DS = fs/(8e3); % factor por el que debo bajar la frecuencia original
    senal = decimate(senal,DS); % decimacion de frecuencia para bajar a 8 KHz
    senal = senal(1:24e3); % tomo solo las primera 24000 muestras (100 bloques de 240 muestras cada uno)
    fs = fs/DS; % la nueva frecuencia de muestreo es 8 KHz
```

```
46 -    N=240;  % Tamaño del bloque, rango 80<N<240
47 -    Numblq=floor(length(senal)/N); % Cálculo del número de bloques
48 -    m=1;   % Variable para posicionamiento del inicio de cada bloque
49 -    NC=10; % Número de coeficientes LPC
50
51 -    senal=senal(1:Numblq*N);
52 -    senal = senal - mean(senal);
53 -    senal = senal/max(abs(senal));
54
55      % Parámteros LPC
56 -    alfa = 0.9375;
57      %%%%%%%%%%%%%%%%%
58
59      % ----- Preprocesamiento
60 -    b = fir1(48,100/fs/2,'high');  % Filtro FIR pasa alta con frecuencia de corte de 100H
61 -    senal1=filter(b,1,senal);
62 -    senal=senal1;
63
64 -    for n=1:Numblq % Procesamiento por bloque
65          % col 1: nombre del audio
66 -        cell_save_trainData{kk,1} = [principal_originales, orig];
```

```
68          % col 2: nro del bloque
69 -        cell_save_trainData{kk,2} = n;
70
71 -        x=senal(m:m+N-1); % Extracción de bloques de N muestras
72
73          % FILTRO DE ENFASIS
74 -        x = filter([1, -alfa],1,x);
75
76          % retirar aquellas senales con potencia muy baja (silencio)
77 -        pwrThreshold = -60;
78 -        pwr = pow2db(var(x));
79
80 -        if pwr < pwrThreshold
81 -            m=m+N;
82 -            kk = kk + 1;
83 -            continue
84 -        end
```

```
86              % low band energy (from 0 Hz to 800 Hz)
87 -            xFFT = fft(x,256);
88              % si 128 es 4 KHz, 800 Hz es 26
89 -            xFFT = xFFT(1:26);
90 -            LBE = sum(xFFT.*conj(xFFT))/length(xFFT);
91              % col 3: LBE
92 -            cell_save_trainData{kk,3} = LBE;
93 -            trainData(kk,1) = LBE;
```

```
95          % PEAK/BOTTOM RATIO OF magnitude difference function // autocorr
96          PBR_MDF = xcorr(x,x);
97          PBR_MDF = abs(max(PBR_MDF)/min(PBR_MDF));
98          % col 4: PBR_MDF
99          cell_save_trainData{kk,4} = PBR_MDF;
100         trainData(kk,2) = PBR_MDF;
101
102         % zero crossing rate
103         ZC = 0.5*sum(abs( sign(x(1:end-1)) - sign(x(2:end)) ));
104         % col 5: ZC
105         cell_save_trainData{kk,5} = ZC;
106         trainData(kk,3) = ZC;
107
108         % col 6: etiqueta
109         cell_save_trainData{kk,6} = labelMat(n,2);
110         trainData(kk,4) = labelMat(n,2);
111
112         m=m+N;
113
114         kk = kk + 1;
115     end
116 end
```

```
118 trainData = trainData(trainData ~= -1e4);
119 trainData = reshape(trainData, length(trainData)/4 , 4);
120 [f,c] = size(trainData);
121 mean_TD = mean(trainData);
122 std_TD = std(trainData);
123 for jj = 1:(c-1)
124     trainData(:,jj) = (trainData(:,jj) - mean(trainData(:,jj)))/std(trainData(:,jj));
125 end
126
127 save([principal_originales 'trainData.mat'],'trainData');
128
129 [coeff,score,latent,tsquared,explained] = pca(trainData(:,1:3));  explained
130
131 ii_voz = find(trainData(:,end) == 1);
132 ii_novoz = find(trainData(:,end) ~= 1);
```

```
134 figure(1)
135 subplot(311)
136 scatter(score(ii_voz,1),score(ii_voz,2),'b'),xlabel('PCA1'),ylabel('PCA2'); hold on,
137 scatter(score(ii_novoz,1),score(ii_novoz,2),'r'),xlabel('PCA1'),ylabel('PCA2'); hold off;
138 subplot(312)
139 scatter(score(ii_voz,1),score(ii_voz,3),'b'),xlabel('PCA1'),ylabel('PCA3'); hold on,
140 scatter(score(ii_novoz,1),score(ii_novoz,3),'r'),xlabel('PCA1'),ylabel('PCA3'); hold off;
141 subplot(313)
142 scatter(score(ii_voz,2),score(ii_voz,3),'b'),xlabel('PCA2'),ylabel('PCA3'); hold on,
143 scatter(score(ii_novoz,2),score(ii_novoz,3),'r'),xlabel('PCA2'),ylabel('PCA3'); hold off;
144
145 figure(2)
146 scatter3(score(ii_voz,1),score(ii_voz,2),score(ii_voz,3),'b'), hold on
147 scatter3(score(ii_novoz,1),score(ii_novoz,2),score(ii_novoz,3),'r'), hold off;
148 axis equal
149 xlabel('1st Principal Component'), ylabel('2nd Principal Component'), zlabel('3rd Principal Component')
150
151 figure(3)
152 scatter3(trainData(ii_voz,1),trainData(ii_voz,2),trainData(ii_voz,3),'b'), hold on
153 scatter3(trainData(ii_novoz,1),trainData(ii_novoz,2),trainData(ii_novoz,3),'r'), hold off;
154 axis equal
155 xlabel('ENERGIA'), ylabel('RATIO MAX/MIN'), zlabel('RATIO CRUCES X CERO')
```

```
157     %% train classifier
158 -   trainData(trainData ~= 1) = -1;
159     % modelo = fitclinear(trainData(:,1:end-1),trainData(:,end), 'Learner', 'logistic','Verbose',1);
160     % modelo = fitclinear(trainData(:,1:end-1),trainData(:,end), 'Learner', 'svm','Verbose',1);
161 -   modelo = fitcsvm(trainData(:,1:end-1),trainData(:,end),'Verbose',1);
162 -   save([principal_originales 'clasifica_voz.mat'],'modelo','mean_TD', 'std_TD');
```

19. eval_odg.m

```
+2  Untitled.m | genera_BD.m | pruebas_clasificador_vozNovoz.m | EVAL_ODG.m | Untitled.m | pruebas_clasificador_vozNovoz.m | genera_graficas.m | genera_BD.m | encoder_FS1015.m
1 -   clear all, close all, clc;
2
3     % audios
4 -   principal_originales = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_original\';
5
6 -   principal_sinteticas = 'D:\TESIS_HENRY\Nueva version_ver4  fs_1015\IMPLEMENTACION FS 1015-version_4\ejemplos_sintetica\'
7
8 -   dirName_original = [principal_originales '*.wav'];
9 -   dd_original = dir(dirName_original);
10
11 -  all_ODG = zeros(length(dd_original),1);
12
13 -  for ii = 1:length(dd_original)
14 -      orig = dd_original(ii).name;
15 -      synth = [dd_original(ii).name(1:end-4) '_synth.wav'] ;
16
17 -      Ftest = [principal_sinteticas synth];
18 -      Fref = [principal_originales orig];
19
20 -      all_ODG(ii) = PQevalAudio(Fref,Ftest);
21 -  end
22
23 -  save('all_ODG.mat','all_ODG');
```

ANEXO 3

ESTÁNDAR FEDERAL 1015

ESTÁNDAR FEDERAL 1015

TELECOMMUNICATIONS: ANALOG TO DIGITAL CONVERSION OF VO!CE BY 2,400 BIT/SECOND LINEAR PREIJICT!VE CODING

3.3.2 TABLA DE ASIGNACION DE BITS: La asignación de los 54 bits en una trama LPC será como se muestra en la siguiente tabla.

	Voiced	Nonvoiced
Pitch & Voicing	7	7
RMS Amplitude	5	5
RC(1)	5	5
RC(2)	5	5
RC(3)	5	5
RC(4)	5	5
RC(5)	4	
RC(6)	4	
RC(7)	4	
RC(8)	4	
RC(9)	3	
RC(10)	2	
Error Control		20
Synchronization	1	1
Unused		1
Total	54	54

Nota: RC = coeficiente de reflexión

Nonvoiced = no voz

3.4 TABLA PITCH AND VOICE

ENCODING: La información de tono y sonoridad se codificará como un campo de siete bits. Para fines de protección contra errores, las tramas no voceadas se codificarán como siete bits CERO y las tramas con transmisión de voz se codificarán como siete bits UNO.

Para las tramas sonoras, se seleccionará y codificará uno de los 60 valores de tono seleccionados (51-400 Hz) como se muestra a continuación. Tenga en cuenta que el período de tono es la frecuencia de muestreo (8.000 Hz),

Pitch Freq	Pitch Period	Code	Pitch Freq	Pitch Period	Code	Pitch Freq	Pitch Period	Code	Pitch Freq	Pitch Period	Code
51	156	76	83	96	78	138	58	26	235	34	46
53	152	101	87	92	74	143	56	58	242	33	38
54	148	100	91	88	75	148	54	56	250	32	39
56	144	108	95	84	73	154	52	60	258	31	7
57	140	104	100	80	77	160	50	52	266	30	15
59	136	106	103	78	69	167	48	54	276	29	14
61	132	98	105	76	85	174	46	50	286	28	30
63	128	114	108	74	81	184	44	51	296	27	22
65	124	112	111	72	83	190	42	49	308	26	23
67	120	113	114	70	82	200	40	53	320	25	21
69	116	97	118	68	86	205	39	37	333	24	29
71	112	99	121	66	84	210	38	45	348	23	25
74	108	67	125	64	92	216	37	41	364	22	27
77	104	71	129	62	88	222	36	43	381	21	11
80	100	70	133	60	90	228	35	42	400	20	19

DECODING: La siguiente tabla se utilizará para decodificar el campo de tono y voz de siete bits para determinar si una trama es no voceada (U), está en transición de sonorización (T), no es válida (0, o está sonora). Si es sonora, se utilizará el valor decodificado como el período de tono (frecuencia de muestreo de 8.000 Hz dividida por la frecuencia de tono).

Code	Pitch Period	Code	Pitch Period	Code	Pitch Period	Code	Pitch Period	Code	Pitch Period
0	(U)	26	58	52	50	78	96	104	140
1	(U)	27	22	53	40	79	(0)	105	(0)
2	(U)	28	(0)	54	48	80	(0)	106	136
3	(0)	29	24	55	(0)	81	74	107	(0)
4	(U)	30	28	56	54	82	70	108	144
5	(0)	31	(0)	57	(0)	83	72	109	(0)
6	(0)	32	(U)	58	56	84	66	110	(0)
7	31	33	(0)	59	(0)	85	76	111	(T)
8	(U)	34	(0)	60	52	86	68	112	124
9	(0)	35	(0)	61	(0)	87	(0)	113	120
10	(0)	36	(0)	62	(0)	88	62	114	128
11	21	37	39	63	(T)	89	(0)	115	(0)
12	(0)	38	33	64	(U)	90	60	116	(0)
13	(0)	39	32	65	(0)	91	(0)	117	(0)
14	29	40	(0)	66	(0)	92	64	118	(0)
15	30	41	37	67	108	93	(0)	119	(T)
16	(U)	42	35	68	(0)	94	(0)	120	(0)
17	(0)	43	36	69	78	95	(T)	121	(0)
18	(0)	44	(0)	70	100	96	(0)	122	(0)
19	20	45	38	71	104	97	116	123	(T)
20	(0)	46	34	72	(0)	98	132	124	(0)
21	25	47	(0)	73	84	99	112	125	(T)
22	27	48	(0)	74	92	100	148	126	(T)
23	26	49	42	75	88	101	152	127	(T)
24	(0)	50	46	76	156	102	(0)		
25	23	51	44	77	80	103	(0)		

3.5 RMS AMPITUDE

Amplitud de voz preacentuada de raíz cuadrada media (RMS), escalada de 0 a 511 (es decir, de 9 bits). Se codificarán en 5 bits y se decodificarán nuevamente en 9 bits como se muestra en la siguiente tabla. Al decodificar, debe utilizarse la interpolación para obtener valores intermedios.

From	To	Code	Decode	From	To	Code	Decode
0	0	0	0	31	35	16	32
1	1	1	1	36	42	17	39
2	2	2	2	43	51	18	46
3	3	3	3	52	60	19	55
4	4	4	4	61	72	20	66
5	5	5	5	73	86	21	79
6	6	6	6	87	103	22	94
7	7	7	7	104	123	23	113
8	8	8	8	124	147	24	135
9	10	9	9	148	176	25	164
11	12	10	11	177	210	26	192
13	14	11	13	211	251	27	230
15	17	12	16	252	300	28	275
18	21	13	19	301	359	29	328
22	25	14	23	360	428	30	392
26	30	15	27	429	511	31	468

3.6 REFLECTION COEFFICIENTS

Reflection Coefficients 1 and 2. The first Reflection Coefficient (RC1) shall be coded as 5 bits. Coding and decoding shall be as shown in the following table. The second Reflection Coefficient (RC2) shall be coded and decoded the same as RC1. Note that coded values are the index numbers expressed in binary. Also, sign convention is such that RC1 ''' R1/Ro, where R1 and Ro are autocorrelation functions with sample delays of 1 and 0 respectively. These conventions are used for all reflection coefficients.

From	To	Code	Index	Decode
-.9999	-.9844	10001	-15	-.9844
-.9843	-.9688	10010	-14	-.9688
-.9687	-.9531	10011	-13	-.9531
-.9530	-.9375	10100	-12	-.9375
-.9374	-.9063	10101	-11	-.9218
-.9062	-.8750	10110	-10	-.8906
-.8749	-.8281	10111	-9	-.8438
-.8280	-.7656	11000	-8	-.7812

From	To	Code	Index	Decode
-.7655	-.6875	11001	-7	-.7187
-.6874	-.6094	11010	-6	-.6406
-.6093	-.5313	11011	-5	-.5625
-.5312	-.4219	11100	-4	-.4688
-.4218	-.3125	11101	-3	-.3593
-.3124	-.2032	11110	-2	-.2500
-.2031	-.0938	11111	-1	-.1406
-.0937	+.0937	00000	0	+.0313
.0938	.2031	00001	1	.1406
.2032	.3124	00010	2	.2500
.3125	.4218	00011	3	.3593
.4219	.5312	00100	4	.4688
.5313	.6093	00101	5	.5625
.6094	.6874	00110	6	.6406
.6875	.7655	00111	7	.7187
.7656	.8280	01000	8	.7812
.8281	.8749	01001	9	.8438
.8750	.9062	01010	10	.8906
.9063	.9374	01011	11	.9218
.9375	.9530	01100	12	.9375
.9531	.9687	01101	13	.9531
.9688	.9843	01110	14	.9688
.9844	.9999	01111	15	.9844

Reflection Coefficient 3. The third Reflection Coefficient (RC3) shall be coded as *5* bits. Coding and decoding shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.5891	10000	-16	-.6033
-.5890	-.5456	10001	-15	-.5598
-.5455	-.5019	10010	-14	-.5164
-.5018	-.4583	10011	-13	-.4729
-.4582	-.4148	10100	-12	-.4295
-.4147	-.3712	10101	-11	-.3860
-.3711	-.3276	10110	-10	-.3426
-.3275	-.2840	10111	-9	-.2991
-.2839	-.2404	11000	-8	-.2557
-.2403	-.1967	11001	-7	-.2122
-.1966	-.1532	11010	-6	-.1687
-.1531	-.1096	11011	-5	-.1253
-.1095	-.0660	11100	-4	-.0818
-.0659	-.0224	11101	-3	-.0384
-.0223	.0212	11110	-2	.0051
.0213	.0648	11111	-1	.0485
.0649	.1139	00000	0	.0920
.1140	.1575	00001	1	.1355
.1576	.2011	00010	2	.1789
.2012	.2447	00011	3	.2224
.2448	.2883	00100	4	.2658
.2884	.3318	00101	5	.3093
.3319	.3755	00110	6	.3527
.3756	.4191	00111	7	.3962
.4192	.4626	01000	8	.4396
.4627	.5062	01001	9	.4831
.5063	.5499	01010	10	.5266
.5500	.5934	01011	11	.5700
.5935	.6370	01100	12	.6135
.6371	.6807	01101	13	.6569
.6807	.7242	01110	14	.7004
.7243	.9999	01111	15	.7438

Reflection Coefficient 4. The fourth Reflection Coefficient shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.7627	10000	-16	-.7774
-.7626	-.7236	10001	-15	-.7383
-.7236	-.6846	10010	-14	-.6993
-.6845	-.6455	10011	-13	-.6602

From	To	Code	Index	Decode
-.6454	-.6065	10100	-12	-.6211
-.6064	-.5674	10101	-11	-.5821
-.5673	-.5283	10110	-10	-.5430
-.5282	-.4893	10111	-9	-.5040
-.4892	-.4502	11000	-8	-.4649
-.4501	-.4112	11001	-7	-.4258
-.4111	-.3721	11010	-6	-.3868
-.3720	-.3330	11011	-5	-.3477
-.3329	-.2940	11100	-4	-.3086
-.2939	-.2549	11101	-3	-.2696
-.2548	-.2158	11110	-2	-.2305
-.2157	-.1768	11111	-1	-.1914
-.1767	-.1329	00000	0	-.1524
-.1328	-.0938	00001	1	-.1133
-.0937	-.0548	00010	2	-.0743
-.0547	-.0157	00011	3	-.0352
-.0156	.0234	00100	4	.0039
.0235	.0624	00101	5	.0429
.0625	.1015	00110	6	.0820
.1016	.1405	00111	7	.1211
.1406	.1796	01000	8	.1601
.1797	.2187	01001	9	.1992
.2188	.2577	01010	10	.2382
.2578	.2968	01011	11	.2773
.2969	.3359	01100	12	.3164
.3360	.3749	01101	13	.3554
.3750	.4140	01110	14	.3945
.4141	.9999	01111	15	.4336

Reflection Coefficient *5*. The fifth Reflection Coefficient (RC5), used when a frame is voiced, shall be coded as 4 bits. Coding and decoding shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.6047	1000	-8	-.6358
-.6046	-.5324	1001	-7	-.5635
-.5323	-.4600	1010	-6	-.4912
-.4599	-.3877	1011	-5	-.4190
-.3876	-.3153	1100	-4	-.3467
-.3152	-.2430	1101	-3	-.2744
-.2429	-.1707	1110	-2	-.2022
-.1706	-.0984	1111	-1	-.1299
-.0983	-.0214	0000	0	-.0577
-.0213	.0509	0001	1	.0146
.0510	.1232	0010	2	.0869
.1233	.1955	0011	3	.1591
.1956	.2679	0100	4	.2314
.2680	.3403	0101	5	.3037
.3404	.4126	0110	6	.3759
.4127	.9999	0111	7	.4482

Reflection Coefficient 6. The sixth Reflection Coefficient (RC6), used when a frame is voiced, shall be coded as 4 bits. Coding and decoding shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.7011	1000	-8	-.7315
-.7010	-.6328	1001	-7	-.6631
-.6327	-.5645	1010	-6	-.5948
-.5644	-.4962	1011	-5	-.5264
-.4961	-.4279	1100	-4	-.4581
-.4278	-.3596	1101	-3	-.3897
-.3595	-.2913	1110	-2	-.3213
-.2912	-.2230	1111	-1	-.2530
-.2229	-.1505	0000	0	-.1846
-.1504	-.0822	0001	1	-.1162
-.0821	-.0139	0010	2	-.0479
-.0138	.0544	0011	3	.0205
.0545	.1227	0100	4	.0888

From	To	Code	Index	Decode
.1228	.1910	0101	5	.1572
.1911	.2593	0110	6	.2256
.2594	.9999	0111	7	.2939

Reflection Coefficient 7. The seventh Reflection Coefficient (RC7), used when a frame ls voiced, shall be coded as 4 bits. Coding and decoding shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.5474	1000	-8	-.5762
-.5473	-.4809	1001	-7	-.5098
-.4808	-.4145	1010	-6	-.4434
-.4144	-.3480	1011	-5	-.3770
-.3479	-.2816	1100	-4	-.3106
-.2815	-.2152	1101	-3	-.2442
-.2151	-.1488	1110	-2	-.1778
-.1487	-.0824	1111	-1	-.1114
-.0823	-.0117	0000	0	-.0450
-.0116	.0547	0001	1	.0214
.0548	.1211	0010	2	.0878
.1212	.1875	0011	3	.1542
.1876	.2540	0100	4	.2206
.2541	.3204	0101	5	.2870
.3205	.3869	0110	6	.3534
.3870	.9999	0111	7	.4198

Reflection Coefficient 8. The eighth Reflection Coefficient (RC8), used when a frame is voiced, shall be coded as 4 bits. Coding and decoding shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.6242	1000	-8	-.6539
-.6241	-.5568	1001	-7	-.5865
-.5567	-.4895	1010	-6	-.5191
-.4894	-.4221	1011	-5	-.4517
-.4220	-.3548	1100	-4	-.3843
-.3547	-.2874	1101	-3	-.3169
-.2873	-.2201	1110	-2	-.2496
-.2200	-.1527	1111	-1	-.1822
-.1526	-.0811	0000	0	-.1148
-.0810	-.0138	0001	1	-.0474
-.0137	.0536	0010	2	.0200
.0537	.1209	0011	3	.0874
.1210	.1883	0100	4	.1548
.1884	.2556	0101	5	.2222
.2557	.3230	0110	6	.2895
.3231	.9999	0111	7	.3569

Reflection Coefficient 9. The ninth Reflection Coefficient (RC9), used when a frame is voiced, shall be coded as 3 bits. Coding and decoding shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.3077	100	-4	-.3634
-.3076	-.1908	101	-3	-.2462
-.1907	-.0738	110	-2	-.1290
-.0737	.0432	111	-1	-.0118
.0433	.1638	000	0	.1054
.1639	.2807	001	1	.2226
.2808	.3977	010	2	.3398
.3978	.9999	011	3	.4570

Reflection Coefficient 10. The tenth Reflection Coefficient (RCJO), used when a frame is voiced, shall be coded as 2 bits. Coding and decodlng shall be as shown in the following table.

From	To	Code	Index	Decode
-.9999	-.3117	10	-2	-.4043
-.3116	-.1203	11	-1	-.2129
-.1202	.0743	00	0	-.0215
.0744	.9999	01	1	.1699