DIVISION DE ESTUDIOS DE POSGRADO

DE LA FACULTAD DE INGENIERIA

CODIFICACION POR TRANSFORMADA DE SEÑALES DE VOZ UTILIZANDO CUANTIZACION VECTORIAL

PABLO VALLE MARTINEZ

TESIS

Presentada a la División de Estudios de
Posgrado de la

FACULTAD DE INGENIERIA

de la

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

como requisito para obtener
el grado de

MAESTRO EN INGENIERIA

(ELECTRICA)

CIUDAD UNIVERSITARIA
JULIO, 1990

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

I N D I C E

R E S U M E N

La codificación de señales de voz para su transmisión en canales con restricciones de ancho de banda, ha sido una de las aplicaciones del procesamiento digital de señales que más investigación ha generado.

El objetivo de esta tesis fué desarrollar un sistema de compresión de señales de voz, utilizando codificación por transformada y cuantización vectorial, para su transmisión en canales con ancho de banda de 16000 bits por segundo.

Se comentan algunos algoritmos de codificación por transformada que han sido reportados en la literatura, se describe la cuantización vectorial y se revisan los algoritmos de distribución de bits más utilizados .

Se presenta el esquema de compresión de señales de voz

desarrollado, detallando cada uno de los algoritmos involucrados en la operación de este, mencionando en cada caso las consideraciones y parámetros empleados. Se hace énfasis en el algoritmo de distribución de bits que se propone, asi como en la forma de cuantizar la fase de los coeficientes de la transformada de Fourier.

A continuación se describen las pruebas realizadas y los resultados obtenidos con el sistema, mencionando algunos datos de operación tanto en lenguaje de alto nivel como en lenguaje ensamblador de un procesador de señales (punto fijo).

Adicionalmente se plantea una variante del sistema de compresión propuesto, la cual permite utilizar el sistema para comprimir varias señales de voz y transmitirlas por un canal de 16000 bits por segundo. Se describe el esquema y se presentan los resultados obtenidos.

I. CODIFICACION POR TRANSFORMADA.

I.1 INTRODUCCION.

La codificación por transformada involucra una transformación lineal en la cual el espacio de señal se mapea a un espacio diferente, en donde las muestras transformadas se comprimen para transmisión o almacenamiento. (Fig. 1.1)

El proceso de codificación involucra básicamente dos operaciones en el transmisor : transformación y cuantización; y dos operaciones en el receptor : decodificación y transformación inversa.

Existen diferentes tipos de transformaciones, generalmente la selección de una de ellas se basa en consideraciones de realización práctica, aún y cuando es posible diseñar una transformación óptima.

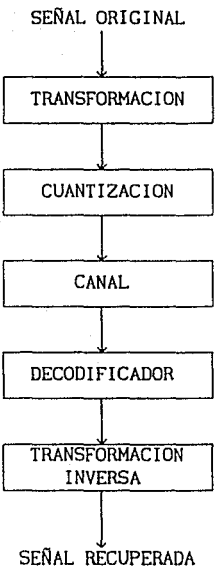

Figura 1.1 Esquema de Codificación por Transformada

La función de la transformación es hacer que las muestras transformadas sean más independientes respecto a las muestras de la señal original, de tal forma que la operación de cuantización se pueda realizar de una forma más eficiente.

En los sistemas de codificación por transformación, el número total de bits disponibles para cuantizar un conjunto de muestras transformadas es fijo, y es necesario distribuir estos bits de tal forma que se minimice la distorsión total de cuantización.

A continuación se describirán brevemente algunas transformacines lîneales : componente principal, Fourier, Hadamard y Haar.

I.2 TRANSFORMACION DE COMPONENTE PRINCIPAL

Se considera una fuente correlacionada estacionaria de variancia σ^2. Se forma un vector x con N muestras sucesivas de la señal, el vector se transforma linealmente utilizando una matriz unitaria A :

$$y = A \cdot x \qquad (1.1)$$

con

$$A^{-1} = A^T \qquad (1.2)$$

Los elementos de y son los coeficientes transformados del esquema de codificación; cada uno de estos elementos es cuantizado independientemente, generando un vector $\hat{y}$. Este vector de coeficientes cuantizados se transmite al receptor, donde se transforma utilizando la matriz inversa A^{-1}:

$$\hat{x} = A^{-1} \cdot \hat{y} \qquad (1.3)$$

Los elementos de $\hat{x}$ son las muestras reconstruídas. Para matrices unitarias la distorsión cuadrática promedio total del esquema de codificación es igual al error de cuantización total [1]:

$$\bar{D} = \frac{1}{N} E\left[(x - \hat{x})^T \cdot (x - \hat{x}) \right] = \frac{1}{N} E\left[(y - \hat{y})^T \cdot (y - \hat{y}) \right] \qquad (1.4)$$

La minimización de $\bar{D}$ involucra : 1) la selección de una forma de asignación de bits óptima, y 2) la selección de una matriz de transformación A óptima.

Para el caso de la transformada de componente principal, la matriz A es tal que los coeficientes del vector y están descorrelacionados. Para lograr esta transformación la matriz A tiene como columnas los vectores característicos normalizados de la matriz de covariancia de la señal original [2].

Las características principales de la transformación por componente principal son :

- La transformación por componente principal descorrelaciona las muestras de la señal original.
- Se debe formar una nueva matriz de transformación (a partir de los vectores característicos de la matriz de covariancia) para cada nuevo vector a procesar.
- Dado un bloque de N muestras, la transformación por componente principal agrupa la mayor cantidad de variancia en los primeros K coeficientes (comparada con cualquier otra transformación) donde K < N. Esto permite que se puedan despreciar los coeficientes de mayor orden para obtener una compresión.
- Minimiza el error cuadrático medio entre el bloque original de muestras y el bloque reconstruido. Este error es igual a la suma de las variancias de los coeficientes despreciados.
- No existe un algoritmo "rápido" para el cálculo de la transformada por componente principal en el mismo sentido que existe un algoritmo rápido para el cálculo de la transformada de Fourier.

I.3. TRANSFORMADA DE FOURIER

La transformada de Fourier es probablemente la transformación más comunmente utilizada. En este caso se mencionará la transformada discreta de Fourier. Así como en el caso de señales continuas existen unas funciones seno y coseno ponderadas e infinitas tales que, al sumarse dan una buena aproximación de la señal continua original , en el caso de la transformada discreta de Fourier existe un conjunto de funciones ortogonales discretas

$$W_{k,m} = e^{-\frac{j2\pi km}{N}} \quad , \ (\ j = \sqrt{-1} \ ) \qquad (1.5)$$

que ponderadas adecuadamente y sumadas, dan una buena aproximación de la secuencia original de N muestras. Si X(m) es una secuencia de N valores reales o complejos, donde m = 0, 1, ..., N-1, los coeficientes transformados están dados por:

$$Cx(k) = \frac{1}{N} \sum_{m=0}^{N-1} X(m) \, e^{-\frac{j2\pi nkm}{N}} \quad , \ k = 0,1,2,\ldots,N-1 \qquad (1.6)$$

La correspondiente transformada inversa está dada por

$$X(m) = \sum_{k=0}^{N-1} Cx(k) \, e^{\frac{j2\pi km}{N}} \quad , \ m = 0,1,2,\ldots,N-1 \qquad (1.7)$$

Una de sus principales ventajas es que existe un algoritmo rápido para el cálculo de la transformada , llamada Transformada Rápida de Fourier (FFT).

I.4. TRANSFORMADA DE HADAMARD

La transformada de Hadamard utiliza una matriz de transformación fija, que toma la forma de una onda cuadrada de dos niveles. Los dos niveles son +1 y -1; la matriz de 2 x 2 es de la siguiente forma :

$$H_2 = \begin{vmatrix} 1 & 1 \\ 1 & -1 \end{vmatrix} \qquad (1.8)$$

La transformada de Hadamard tiene una propiedad interesante: Transformaciones de mayor orden a 2 y cuya dimensionalidad sea múltiplo de 2, se pueden generar a partir de transformaciones de orden menor como sigue:

$$H_{2N} = \begin{vmatrix} H_N & H_N \\ H_N & -H_N \end{vmatrix} \qquad (1.9)$$

De la misma forma que para la transformada de Fourier, para la transformada Hadamard existe un algoritmo que permite calcular ésta de forma eficiente.

I.5 TRANSFORMADA HAAR

La transformada Haar utiliza las funciones ortonormales Haar. Existe una simulitud entre la matriz de transformación Haar y la matriz de transformación Hadamard. Los renglones de la matriz también representan ondas rectangulares (con amplitudes 0, 1, -1, multiplicados por potencias de $\sqrt{2}$). Un ejemplo de matriz Haar 8 x 8 es :

$$H_8 = \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{vmatrix} \qquad (1.10)$$

En forma matricial, la transformación Haar de la secuencia de datos X es :

$$HAAR = \frac{1}{N} [\, H_N \,][\, X \,] \qquad (1.11)$$

y su inversa es :

$$X = \frac{1}{N} [\, H_N \,]^t [\, HAAR \,] \qquad (1.12)$$

donde t significa transpuesta.

I.6 TRANSFORMADA COSENO.

La Transformada Coseno de una secuencia de datos X(m) , m = 1 , 2 ,... , (M-1) está definida como [3]:

$$G_x(0) = \frac{\sqrt{2}}{M} \sum_{m=0}^{M-1} X(m)$$

$$G_x(k) = \frac{2}{M} \sum_{m=0}^{M-1} X(m) \cos \frac{(2m+1)k\pi}{2M}, \quad k = 1, 2, \ldots, (M-1) \qquad (1.13)$$

donde $G_x(k)$ es el k-ésimo coeficiente de la Transformada Coseno Discreta.

I.7 ALGUNOS COMENTARIOS SOBRE DESEMPEÑO DE LAS TRANSFORMADAS.

La transformada de componente principal (Karhunen-Loeve KLT) da el mejor desempeño en la codificación por transformada, su principal desventaja es la falta de un algoritmo rápido para el cálculo de los coeficientes; además, la matriz es dependiente de la señal y el cálculo de sus coeficientes no es simple.

En cuanto a las transformadas Hadamard (WHT), Discreta de Fourier (DFT) y Coseno (DCT), todas son independientes de la señal y tienen algoritmos rápidos para el cálculo de los coeficientes transformados. Las transformadas mencionadas son subóptimas porque los coeficientes transformados no están totalmente descorrelacionados.

Para comparar el desempeño de estas transformadas [2], se considera un proceso estacionario de Markov orden 10, se calcula la matriz de covariancia R_{xx}, y con la matriz A, se obtiene la matriz de covariancia $R_{yy} = A\, R_{xx}\, A^T$, en el dominio de la transformación. Se obtienen las variancias de los coeficientes transformados y utilizando la siguiente expresión se calcula la ganancia en SNR de cada transformada sobre PCM (fig. 1.2) :

$$G_{TC} \cong \frac{\overline{D_{PCM}}}{\overline{D}} = \frac{\sigma^2}{\left[\prod_{j=1}^{N} \sigma_j^2 \right]^{1/N}} \qquad (1.14)$$

G_{TC} es el incremento en relación señal a ruido (SNR) sobre PCM.

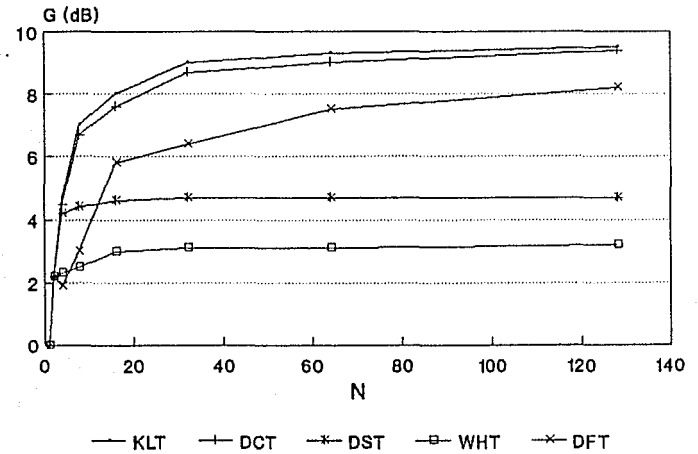

Figura 1.2 Ganancias en SNR sobre PCM respecto a la longitud N del bloque de muestras para varias transformadas.

Nótese que para longitudes de bloque N grandes, el desempeño de las transformadas discretas de Fourier y Coseno, se aproximan al óptimo de la transformada por componente principal.

II. CUANTIZACION Y DISTRIBUCION DE BITS.

II.1 CUANTIZACION

La cuantización de una secuencia de muestras se puede analizar desde dos puntos de vista: (1) Dado un número fijo de niveles de cuantización se desea minimizar el ruido de cuantización; este ruido o distorsión se genera al representar un rango de amplitudes dentro de un intervalo de cuantización con sólo un valor de amplitud (llamado nivel de cuantización) de ese intervalo; (2) Dado un valor fijo de ruido de cuantización o distorsión, se desea minimizar el número de bits por muestra que satisfacerá la especificación de distorsión. El último punto de vista está relacionado con la función tasa-distorsión y , de hecho, la función tasa-distorsión puede utilizarse como límite inferior teórico para la tasa de la fuente, en ciertos modelos de la fuente.

Existen básicamente tres formas de cuantización:

1. Cuantización de memoria cero.
2. Cuantización por bloque.
3. Cuantización secuencial.

Cuantización de memoria cero, es la cuantización de una muestra a la vez. Cuantización por bloque es la representación de un bloque de muestras por un bloque de valores de salida, seleccionado de un conjunto discreto de posibles bloques. La cuantización secuencial es la cuantización de una muestra utilizando información de las muestras vecinas ya sea en bloque o muestra a muestra.[2]

Existen un gran número de tipos de cuantizadores reportados en la literatura [2] [5], esta tesis se limitará a describir los cuantizadores de bloque ya que como más adelante se mencionará, se utilizan en el sistema final.

II.2 CUANTIZADORES DE BLOQUE

Un cuantizador de N niveles y dimensión k, es un mapeo, q, que asigna a cada vector de entrada, $x = (x_0,\ldots,x_{k-1})$, un vector de reproducción, $\hat{x} = q(x)$, seleccionado de un conjunto finito de vectores de reproducción, $\hat{A} = \{ y_i;\ i=1,\ldots,N \}$. El cuantizador q queda completamente descrito por su alfabeto de reproducción $\hat{A}$ junto con la partición, $S = \{S_i\ ;\ i=1,\ldots,N\}$, del espacio de vectores de entrada en conjuntos $S_i = \{ x : q(x) = y_i \}$ de vectores de entrada que se mapean en

el i-ésimo vector de reproducción. A este tipo de cuantizadores se les conoce como cuantizadores de bloque o cuantizadores vectoriales. [4]

Un cuantizador de N niveles será óptimo si minimiza la distorsión, esto es, q^* es óptimo si para todos los otros cuantizadores q del conjunto de N vectores de reproducción $D(q^*) \leq D(q)$. El objetivo del diseño de un cuantizador vectorial es obtener un cuantizador óptimo si es posible, y si no, obtener un cuantizador localmente óptimo.

El algoritmo de diseño de un cuantizador vectorial es el siguiente:

1. Se establece el número de niveles máximo deseado $N = 2^R$, donde R es un entero. Se fijan también, el valor de K, que corresponde al número de componentes de cada vector, n, que es la longitud de la secuencia de entrenamiento, ε, el umbral de mínima distorsión y se inicializa un contador M con 1, donde M indica el nivel actual.

2. Dada la secuencia de entrada $\{x_j ; j = 0, \ldots, n-1\}$, se define $A = \{x_j ; j = 0, \ldots, n-1\}$ como el alfabeto de la secuencia de entrenamiento. Se define también $\hat{A}(1) = \hat{x}(A)$, como el centroide de la secuencia de entrenamiento. El centroide se determina de acuerdo con una medida de distorsión. En el caso de la medida de distorsión de error cuadrático es el centroide Euclideano o la suma vectorial de todos los vectores de entrada codificados en un símbolo dado, y está dado por

$$x(A) = \frac{1}{\| A \|} \sum_{i=1}^{n} x_i \qquad (2.1)$$

donde ||A|| denota al número de vectores de entrenamiento.

3. División (splitting). Dada $\hat{A}(M) = \{y_i,\ i = 1,\ \ldots,\ M\}$, se divide cada vector y_i, en $y_i + \varepsilon$ e $y_i - \varepsilon$, donde ε es un vector de perturbación fijo. Se define $\hat{A}_0(2M) = \{y_i + \varepsilon, y_i - \varepsilon,\ i = 1,\ \ldots,\ M\}$ y se remplaza M por 2M.

4. Se inicializa m=0 y $D_{-1} = \infty$.

5. Dado $\hat{A}_m(M) = \{y_1,\ \ldots,\ y_M\}$, se encuentra su partición óptima $P(\hat{A}_m(M)) = \{\ S_i;\ i = 1,\ \ldots,\ M\ \}$, esto es, $x_j \in S_i$ si $d(x_j, y_i) \leq d(x_j, y_l)$, para toda l.

6. Se calcula la distorsión resultante

$$D_m = D(\{\hat{A}_m(M), P(\hat{A}_m(M))\})$$
$$= n^{-1} \sum_{j=0}^{n-1} \min_{y \in \hat{A}_m} d(x_j, y) \qquad (2.2)$$

7. Si $(D_{m-1} - D_m)/D_m \leq \varepsilon$, entonces paso 9. En caso contrario continúa.

8. Se encuentra el alfabeto de reproducción óptimo $\hat{A}_{m+1}(M) = \hat{x}(P(\hat{A}_m(M))) = \{\hat{x}(S_i);\ i = 1,\ \ldots, M\}$ para $P(\hat{A}_m(M))$. Se remplaza m por m+1 y se repite desde el paso 5.

9. Se hace la asignación $\hat{A}(M) = \hat{A}_m(M)$. El cuantizador final de M

niveles queda descrito por $\hat{A}(M)$. Si M = N , termina, con el cuantizador final $\hat{A}(N)$. En caso contrario repite desde el paso 3.

Este algoritmo se describe en [4], mientras que un análisis detallado de la cuantización vectorial en general, se encuentra en [5] y [6]. En [6] se demuestra el mejor desempeño de un cuantizador vectorial comparado con la cuantización escalar de cada componente de un vector.

II.2 DISTRIBUCION DE BITS.

Como se mencionó en el capítulo anterior, para minimizar la distorsión entre las muestras originales y las reproducidas en un sistema de codificación por transformada, se debe seleccionar una matriz de transformación óptima y realizar una distribución de bits también óptima [1].

Considerando un esquema de codificación por transformada donde los coeficientes y_i; i=1,2,...,N se cuantizan independientemente, las variancias de los coeficientes transformados difieren en general, y se requieren R_i bits/muestra para el coeficiente y_i de variancia σ_i^2 si no se desea exceder una distorsión de error cuadrático medio D_i [1]:

$$R_i = \delta + \frac{1}{2} \log_2 \frac{\sigma_i^2}{D_i} \quad ; \quad i = 1,2,\ldots,N \qquad (2.3)$$

El segundo término de la expresión anterior, da la mínima tasa para variables aleatorias Gaussianas independientes e idénticamente

distribuidas, y δ es un valor de corrección que toma en cuenta el desempeño de los cuantizadores prácticos; depende del tipo de cuantizador seleccionado y de la función densidad de probabilidad de la señal a cuantizar. Despreciando la dependencia de δ sobre R_i, el número óptimo de bits que se necesitan para el cuantizador Q_i se obtiene minimizando la distorsión promedio

$$\bar{D} = \frac{1}{N} \sum_{i=1}^{N} D_i \qquad (2.4)$$

con la restricción de una tasa de transmisión constante:

$$\bar{R} = \frac{1}{N} \sum_{i=1}^{N} R_i = \text{constante} \qquad (2.5)$$

El resultado al problema de optimización anterior es que todos los coeficientes resultado de la transformación tienen que tener la misma distorsión ($D_i = \bar{D}$ para toda i). La distribución óptima de bits es entonces

$$R_i = \bar{R} + \frac{1}{2} \log_2 \frac{\sigma_i^2}{\left[\prod_{j=1}^{N} \sigma_j^2 \right]^{1/N}} \quad \text{(bits/muestra)} \qquad (2.6)$$

El cálculo del número de bits para los coeficientes de la transformación, puede resultar en un número negativo para aquellos coeficientes con variancia muy pequeña, en ese caso se selecciona un cuantizador de cero bits, por ejemplo se omite la transmisión de esa muestra. Los cuantizadores restantes deben volver a ser optimados, en la

literatura se proponen métodos para efectuar esto, algunos de ellos toman encuenta la restricción de manejar números enteros en la asignación de bits e imponen además un número máximo de bits permitido.

Con una asignación de bits óptima, la distorsión promedio es :

$$\bar{D} = 2^{2\delta} \cdot 2^{-2\bar{R}} \cdot \left[\prod_{j=1}^{N} \sigma_j^2 \right]^{1/N} \qquad (2.7)$$

De donde se concluye que la distorsión suministrada por un esquema de codificación por transformada (CT) depende de la distribución de las variancias; en particular $\bar{D}$ está determinada por la media geométrica de las variancias.

La desventaja del método anterior es que considera un conjunto de cuantizadores idénticos con función de cuantización exponencial y que es posible obtener un número negativo de bits para los coeficientes de variancia pequeña, a partir de esa distribución se busca obtener otra por prueba y error en la que se tenga un número de bits entero y positivo para cada coeficiente.

Segal [7] propuso un modificación al método anterior para mejorar el desempeño. Para un esquema de codificación dado, se define una función de error cuadrático medio k(R), y se considera una fuente Gaussiana de memoria cero con variancia = 1 . k(R) es una función decreciente. Si se asignan R_j bits/componente del vector y, el error cuadrático medio total será

$$D = \sum_{i=1}^{N} \sigma_i^2 \, k(R_i) \qquad (2.8)$$

esta cantidad debe minimizarse sujeta a las restricciones

$$R_i \geq 0 \qquad \bar{R} = \frac{1}{N} \sum_{i=1}^{N} R_i = \text{constante} \qquad (2.9)$$

La solución al problema anterior de optimización está dada por las siguientes expresiones:

$$R_i = \begin{cases} R_i^* = h\left(\dfrac{\theta^*}{\sigma_j^2} k'(0) \right), & \text{si } 0 < \theta^* < \sigma_i^2 \\ 0, & \text{si } \theta^* \geq \sigma_i^2 \end{cases} \qquad (2.10)$$

donde θ^* es la única raíz de la ecuación

$$S(\theta) = \sum_{i:\sigma_i^2 \geq \theta} h\left(\frac{\theta}{\sigma_j^2} k'(0) \right) = C \qquad (2.11)$$

El valor de mínima distorsión es

$$D^*(\theta^*) = \sum_{\sigma_j^2 \geq \theta^*} \sigma_j^2 \, k(R_i^*) + \sum_{\sigma_j^2 < \theta^*} \sigma_j^2 \qquad (2.12)$$

Como puede observarse, Segall eliminó la restricción de que las funciones de los cuantizadores fuera exponencial, pero requiere ahora que sea estrictamente convexa. Como antes, todos los cuantizadores se consideran idénticos, el autor describe una solución óptima para el

problema, pero permitiendo también, en las distribuciones, números reales, los cuales se restringen a no ser negativos.

En la mayoría de las aplicaciones, el número de bits que se pueden asignar a un cuantizador está limitado superiormente por un valor máximo, debido a que existe un límite en que tan fino puede ser un cuantizador. Esto es de particular importancia para un cuantizador vectorial debido a que el tamaño del alfabeto crece exponencialmente con el número de bits.

Algunas veces es conveniente también establecer un límite inferior para el número de bits a asignar a un cuantizador, esto reduce la cantidad de memoria necesaria para almacenar los alfabetos.

Hasta ahora, ninguno de los métodos para distribución de bits, es aplicable en el caso de diferentes rangos de valores de bits enteros persimibles.

Un algoritmo de distribución de bits óptimo para valores enteros no negativos fue propuesto por Fox. El algoritmo es aplicable para un conjunto de cuantizadores diferentes, sin embargo, la consideración básica es que las funciones de cuantización deben ser convexas y estrictamente decrecientes con respecto al número de bits. En este algoritmo no se incluyen los límites superior e inferior mencionados.

En muchos casos prácticos, los cuantizadores no son idénticos, esto

se debe a que las secuencias a cuantizar no tienen la misma estadística, y cada cuantizador debe ser optimado para su propia secuencia de entrada. La restricción de que las funciones de los cuantizadores sean convexas no siempre ocurre, que sean convexas lo sugiere la teoría de tasa-distorsión en los casos donde el cuantizador se aproxima al límite de la tasa-distorsión, el cual implica bloques de codificación muy grandes. Los cuantizadores prácticos operan sobre pequeños bloques de datos y no alcanzan el límite de la tasa-distorsión.

Una solución óptima para el problema de distribución se puede obtener utilizando una programación dinámica. Sin embargo la complejidad de esta técnica es bastante grande. Si se desean distribuir R bits en un conjunto de cuantizadores, la programación dinámica requiere del orden de R^2 operaciones por cuantizador.

Finalmente, Shoham y Gersho, proponen una solución para el problema de distribución de bits.

El problema se plantea como sigue: Se considera un conjunto de M cuantizadores de tasa variable, y un conjunto de ganancias representadas con el vector G = (g_1, g_2, ..., g_M). La entrada al k-ésimo cuantizador se normaliza primero por la correspondiente ganancia g_k y se cuantiza con i bits, donde i pertenece al conjunto de valores positivos enteros { p_k, ... , q_k }. El errors cuadrático resultante para todos los bits admisibles, forman la función del cuantizador normalizada para el k-ésimo cuantizador y se denotará por { $\theta_k(i)$, i= p_k, p_{k+1}, ..., q_k }. Si se utilizan i bits, el error cuadrático en la

salida del k-ésimo cuantizador es $(g_k)^2\theta_k(i)$.

Al igual que en los casos anteriores, el problema es minimizar el error cuadrático total, para distribuir R bits de la mejor forma en el grupo de cuantizadores. La función a minimizar es

$$\sum_{i=1}^{M} (g_i)^2 \theta_i(R_i) \qquad (2.13)$$

sujeto a

$$p_i \leq R_i \leq q_i , \quad i = 1, \ldots, M \qquad (2.14)$$

y

$$\sum_{i=1}^{M} R_i \leq R \qquad (2.15)$$

Debe notarse que a diferencia de los algoritmos mencionados anteriormente, en este caso se permite que los dominios $\{ p_k, \ldots, q_k \}$ sean diferentes para cada cuantizador (cada k); además se incluye como restricción que los valores de bits asignados sean positivos y enteros.

La solución se obtiene al resolver el problema sin restricción

$$\min_{B \in S} \left\{ \sum_{k=1}^{M} W_k(b_k) + \lambda \sum_{k=1}^{M} b_k \right\} \qquad (2.16)$$

donde

$$W_k(b_k) = (g_k)^2 \theta_k(b_k) \quad , \; k = 1, \ldots, M$$

$$S = \{ B : b_k \in \{ p_k, \ldots, q_k \}, \quad k = 1, \ldots, M \}$$

La solución se obtiene minimizando cada término de la expresión (2.16) en forma independiente. Entonces $b_k^*(\lambda)$ es la k-ésima componente del vector óptimo $B^*(\lambda)$ y $S_k = \{ p_k, \ldots, q_k \}$ y es la solución de

$$\min_{i \in S_k} \{ W_k(i) + \lambda i \} \qquad (2.17)$$

Dado λ, se puede resolver para todos los b_k^*, sumarlos y obtener

$$R^*(\lambda) = R(B^*(\lambda)) = \sum_{k=1}^{M} b_k \qquad (2.18)$$

y comparar este valor con el valor de R deseado. Si $R^*(\lambda) = R$ se ha obtenido la solución deseada. Sin embargo la solución de $b_k^*(\lambda)$ puede no ser única.

En resumen, el algoritmo es el siguiente :

1. Obtener un valor inicial de λ. Para ello se sugiere tomar como valor inicial

$$\lambda = \frac{1}{M} \sum_{k=1}^{M} \{ W_k(a_k) - W_k(a_k + 1) \} \qquad (2.19)$$

donde a_k es el valor entero más próximo a $(p_k + q_k) / 2$

2. Resolver el problema sin restricción utilizando la ecuación 2.18. Si λ es no singular, sólo existe una solución y una restricción $R^*(\lambda)$. Si λ es singular, entonces existen al menos dos soluciones diferentes. Encontrar las dos soluciones que generan las restricciones mínima y máxima, $R_l^*(\lambda)$ y $R_h^*(\lambda)$ respectivamente, del conjunto de posibles soluciones $\{ B^*(\lambda) \}$.
3. Si la restricción deseada R es tal que

$$R_l^*(\lambda) \leq R \leq R_h^*(\lambda)$$

entonces se obtienen todas las soluciones del conjunto { $B^*(\lambda)$ } utilizando 2.18 y se encuentra aquella para la cual la restricción denotada $R_a^*(\lambda)$ es la más cercana a R y menor que ella. Si $R_a^*(\lambda) = R$ se ha obtenido una solución óptima, si no, la solución obtenida es aproximada B_a^*. Se detiene la búsqueda.

4. Si $R < R_l^*(\lambda)$, se encuentra la siguiente λ singular utilizando la expresión

$$\lambda = \min_{k \in M^-} \min_{i \in S_k^-} \frac{W_k(i) - W_k^*(b_k(\lambda))}{b_k^*(\lambda) - i} \qquad (2.20)$$

donde

$S_k^- = \{ p_k, \ldots, b_k^*(\lambda)-1 \}$ el cual no esta vacío si $b_k^*(\lambda) \geq 1$

$M^- = \{ k : k \in \{ 1, \ldots, M \} : S_k^-$ no está vacío $\}$;

y la solución presente $B^*(\lambda)$ la cual corresponde a $R_l^*(\lambda)$. Se repite desde el punto 2.

5. Si $R > R_h^*(\lambda)$, se encuentra la siguiente λ singular utilizando la expresión

$$\lambda = \max_{k \in M^+} \max_{i \in S_k^+} \frac{W_k(b_k^*(\lambda)) - W_k(i)}{i - b_k^*(\lambda)} \qquad (2.21)$$

donde

$S_k^+ = \{ b_k^*(\lambda)+1, \ldots, q_k \}$ el cual no esta vacío si $b_k^*(\lambda)+1 \leq q_k$

$M^+ = \{ k : k \in \{ 1, \ldots, M \} : S_k^+$ no está vacío $\}$;

y la solución presente $B^*(\lambda)$ la cual corresponde a $R_{hl}^*(\lambda)$. Se repite desde el punto 2.

Si la búsqueda termina con una solución aproximada, lo cual significa que la solución óptima es inaccesible, entonces esta solución se puede ajustar agregando algunos bits hasta que la suma de la distribución de bits sea igual a R. El criterio para agregar bits es el provocar la mayor reducción en la distorsión

$$\sum_{i=1}^{M} (g_i)^2 \theta_i(b_i) \Big|_{B^*(\lambda)}$$

Se agrega un bit de acuerdo con los siguientes pasos :

6. Se encuentra k', que es la solución de

$$\min_{k \in M^+} \{ W_k(b_k^* + 1) - W_k(b_k^*) \}$$

7. Se actualiza la solución agregando un bit a b_k^*.
8. Se repiten los pasos 6 y 7 hasta que la suma de los bits distribuidos sea igual a R.

En el mismo artículo se presentan unas variantes al algoritmo anterior que pretenden reducir la complejidad del mismo, estas son:

Variante I:

1. Se encuentran los valores de λ_1 y λ_3 que satisfacen

$$R^*(\lambda_1) \leq R \leq R^*(\lambda_3) \qquad (2.22)$$

Una forma de lograr lo anterior es como sigue:

1.1. Se encuentra un valor inicial λ_1 utilizando 2.19.

1.2. Se resuelve 2.17 con S_k anterior para todo k y se calcula $R^*(\lambda_1)$.

1.3. Si $R^{*}(\lambda_1)$ es mayor que R, entonces $\lambda_3 = \lambda_1$ y $B^{*}(\lambda_3) = B^{*}(\lambda_1)$. Se incrementa λ_1 sumándole una constante previamente seleccionada. Se repite desde 1.2.

1.4. Si $R^{*}(\lambda_1)$ es menor que R, se decrementa λ_1 restándole la constante previamente seleccionada. Se repite 1.2.

1.5. Se repite lo anterior hasta que se obtengan los valores de λ_1 y λ_3 que satisfagan 2.22.

2. Se continúa con el procedimiento básico pero modificando las regiones de búsqueda de la siguiente forma :

$$S_k = \{ i : b_k^{*}(\lambda_3) \leq i \leq b_k^{*}(\lambda_1) \} , \quad k = 1, \ldots, M \qquad (2.23)$$

$$S_k^{+} = \{ b_k^{*}(\lambda_3) + 1, \ldots, b_k^{*}(\lambda_1) \} , \quad k = 1, \ldots, M \qquad (2.24)$$

$$S_k^{-} = \{ b_k^{*}(\lambda_1), \ldots, b_k^{*}(\lambda_3) - 1 \} , \quad k = 1, \ldots, M \qquad (2.25)$$

Debe notarse que al ir obteniendo las soluciones para cada λ, las regiones de búsqueda pueden modificarse dinámicamente, decrementando continuamente la intensidad de búsqueda.

El valor constante de actualización del paso 1.1 determina la velocidad con que el procedimiento se aproximará a las primeras dos soluciones deseadas. Si la constante es grande, las dos soluciones se pueden obtener en una o dos iteraciones, sin embargo, pueden estar muy alejadas lo que implica regiones de búsqueda grandes. Por otro lado si la constante es pequeña, se requieren muchas iteraciones para obtener las soluciones, pero las regiones de búsqueda resultante serán pequeñas ya que las soluciones estarán próximas. El valor de la constante debe obtenerse experimentalmente comparando la complejidad de la búsqueda para diferentes valores.

Debido a que durante las pruebas experimentales, se observó que nunca ocurrían más de dos soluciones para λ singular se propone la segunda variante del algoritmo :

Variante II.

1. Encontrar los valores iniciales λ_1 y λ_3 como en la variante 1.
2. Inciando con λ_3 se obtienen soluciones y valores singulares resolviendo sucesivamente 2.21 para nuevo λ_3 y nueva solución $B^*(\lambda_3)$. Se actualizan las regiones de búsqueda como se describió en la variante 1.
3. Cuando $R^*(\lambda_3)$ sea menor o igual que R, se detiene la búsqueda.
4. Si es necesario se ajusta la solución $B^*(\lambda_3)$ para obtener una solución final aproximada B_a^* como el los pasos 6-8 del algoritmo básico.

Como se pudo observar, este algoritmo no se basa en la naturaleza del conjunto de cuantizadores, optimiza la distribución de bits para el desempeño real de cada cuantizador.

III. SISTEMA DE COMPRESION

III.1. SISTEMAS REPORTADOS EN LA LITERATURA.

Existe una gran variedad de algoritmos de compresión de señales que utilizan como parte fundamental la codificación por transformada. En particular, para señales de voz se han propuesto muchos esquemas, desde los que codifican directamente los coeficientes de la transformada como los que codifican parámetros que representan al sistema de producción de voz en el ser humano.

En [12] se puede encontrar un resumen de técnicas que se han utilizado para la compresión de señales de voz, algunas no utilizan codificación por transformada.

De los algoritmos para compresión de señales de voz que utilizan codificación por transformada y que han sido reportados recientemente

se comentarán dos, el presentado por Moriya et.al.[9] y por Cuperman [11].

El algoritmo de Moriya et.al. es el siguiente : inicialmente se analiza la señal de entrada utilizando predicción lineal para obtener un análisis en pequeños intervalos y una predicción del periodo de tono. La señal residuo se transforma utilizando la Transformada Discreta Coseno mencionada en el primer capítulo (TDC) . La matriz de la TDC puede diagonalizar aproximadamente la matriz de correlación R . Para un orden de transformación m grande, R puede reemplazarse por una matriz Toeplitz que contenga los coeficientes de la autocorrelación r_j asociados al j-ésimo retardo (j = 0, ..., m-1). Las componentes de la diagonal o factores que ponderarán a los cuantizadores λ_i se aproximan utilizando la siguiente expresión :

$$\lambda_i \cong \sum_{j=1}^{m-1} \left(\frac{m - j}{m} \right) r_j \cos\left[\frac{\pi (i - 1) j}{m} \right] + \frac{1}{2} r_0 \qquad (3.1)$$

El número de muestras a transformar se selecciona de 128 o 256. Las componentes de TDC se separan en varios subvectores para reducir la complejidad computacional de la búsqueda en los alfabetos para una tasa fija. Estos subvectores se construyen reacomodando las componentes de la TDC en orden de acuedo con el eje de frecuencia, de tal forma que la media geométrica de los factores que ponderarán a los cuantizadores para cada subvector sean estadisticamente uniformes. Después, cada componente de los vectores se decima a intervalos fijos apartir de las componentes de la TDC. Los parámetros de la envolvente espectral (orden

10 , parámetros LSP) se cuantizan escalarmente o vectorialmente.

Por otro lado, el sistema propuesto por Cuperman [11], procesa vectores de M muestras consecutivas de una fuente correlacionada. Se multiplica este vector de entrada por una matriz unitaria A y el vector transformado $y = A\ x$ se cuantiza utilizando m cuantizadores vectoriales de dimensiones k_i , teniendo alfabetos de tamaño N_i donde i = 1, 2, ..., m. La entrada al primer cuantizador vectorial CV_1 son las primeras k_1 componentes del vector y, la entrada al segundo cuantizador vectoria CV_2 son las siguientes k_2 componentes y así sucesivamente. el vector binario b obtenido a la salida de los cuantizadores vectoriales se transmite. En el receptor se utilizan los "cuantizadores vectoriales inversos" CVI_i, i = 1, 2, ..., m, para obtener el vector y^*. Finalmente, se obtiene la transformada inversa de y^* utilizando A^{-1} para obtener el vector reconstruido x^*.

Cuperman utiliza la Transformada Discreta Coseno para la matriz de transformación A, debido a que su desempeño es el más próximo al óptimo de la transformada por componente principal, como se mencionó en el primer capitulo de esta tesis.

La codificación en cada uno de los cuantizadores vectoriales se realiza de acuerdo con el criterio de mínima distorsión (vecino más cercano) y utilizando búsqueda completa en el alfabeto.

Un punto interesante del esquema presentado por Cuperman es el que

se refiere a la distribución de bits para cada cuantizador vectorial.

Para derivar la distribución de bits óptima, utiliza la teoría asintótica de cuantización vectorial [13], y encuentra que la relación tasa-distorsión para un cuantizador vectorial es :

$$r_i = R + 0.5 \log_2 \frac{c_i \, ||p||_i}{\prod_{j=1}^{m} (c_j \, ||p||_j)^{k_j/M}} \qquad (3.2)$$

donde c_i es el coeficiente de cuantización , $||p||_i$ es la norma de orden $k_i/(k_i+2)$ de p_i, la función distribución de probabilidad multivariable del vector y, k_i es la dimensión del cuantizador vectorial CV_i y R la tasa total de transmisión.

En el caso de que la señal de entrada fuera Gaussiana la distribución óptima sería :

$$r_i = R + \beta_i + 0.5 \log_2 \frac{(\det \Psi_i)^{1/k_i}}{\prod_{j=1}^{m} (\det \Psi_j)^{1/M}} \qquad (3.3)$$

donde β_i es un término que depende de los coeficientes de cuantización y de las dimensiones del vector. Si todos los vectores tienen la misma dimensión $k_i = k$ entonces $\beta_i = 0$. Si no es el caso, entonces

$$\beta_i = 0.5 \log_2 \frac{\gamma_i}{\prod_{j=1}^{m} (\gamma_i)^{k_j/M}} \qquad (3.4)$$

donde

$$\gamma_i = c_j \left[1 + \frac{2}{k_j} \right]^{k_j/2 + 1} \qquad (3.5)$$

Si se considera que el vector resultado de la transformación no está correlacionado, entonces, la ecuación 3.3 se puede escribir como

$$r_i = R + \beta_i + 0.5 \log_2 \frac{\left[\prod_{j=1}^{k_i} \sigma_{ij}^2 \right]^{1/k_j}}{\left[\prod_{j=1}^{m} \prod_{h=1}^{k_j} \sigma_{jh}^2 \right]^{1/M}} \qquad (3.6)$$

donde σ_{ij} es la variancia de la j-ésima componente en el i-ésimo vector.

Debido a que el procedimiento anterior fué derivado para una señal de entrada Gaussiana, es necesario incluir un ajuste para adaptar la distribución de bits a la estadística variable de la señal de voz, para ello utiliza el modelo de estacionaridad local.

Dadas las dimensiones de los vectores k_i, $i = 1, 2, \ldots, m$, los coeficientes β_i permanencen constantes en el proceso de adaptación. Se calcula una nueva distribución de bits para cada bloque de nM muestras utilizando en 3.6 las variancias estimadas $\hat{\sigma}^2_{ij}$, dadas por

$$\hat{\sigma}^2_{ij} = \frac{1}{n} \sum_{h=1}^{n} y^2_{ij(h)} \qquad (3.7)$$

donde $y_{ij(h)}$ es el valor de la j-ésima componente en el i-ésimo vector en el h-ésimo marco, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, k_i$, $h = 1, 2, \ldots,$

n. El algoritmo final se puede resumir como sigue:

1. Para cada bloque de nM muestras se calcula la variancia utilizando 3.7.
2. Se calcula la nueva distribución de bits utilizando 3.6.
3. Si $r_j < 0$ para cualquier $j = 1, 2, \ldots, m$; entonces $r_j = 0$.
4. Si $k_j r_j > \log_2 N_{max}$ para cualquier $j = 1, 2, \ldots, m$; donde N_{max} es el tamaño máximo del alfabeto, entonces $k_j r_j = N_{max}$.
5. Se redistribuyen los bits disponibles a otros alfabetos. El algoritmo de redistribución está basado en calcular el decremento en distorsión cuando se aumenta un bit a la distribución de bits en cada CV_i.

En los esquemas presentados hasta aquí, es necesario enviar al receptor información adicional para la distribución de bits utilizada.

III.2 INTRODUCCION AL SISTEMA DE COMPRESION

Una vez comentados dos algoritmos de compresión de señales de voz que utilizan codificación por transformada, mencionaremos ahora el sistema de compresión de señales de voz utilizando codificación por transformada y cuantización vectorial que se propone.

El objetivo primordial del sistema es, poder transmitir una señal de voz en forma digital, en un canal de ancho de banda de 16 kbps, de tal forma que la calidad de la señal recuperada en el receptor sea aceptable. El término aceptable se refiere, tanto a que se entienda

perfectamente el mensaje como se identifique a la persona que habla.

Las características principales que se desean para el sistema son las siguientes:

- Comprimir la señal de voz para su transmisión en un canal de 16kbps, con buena calidad en la señal recuperada en el receptor.
- El sistema debe poder ser implantado en un procesador de señales, con el objetivo de que opere en tiempo real.

Como se menciona en [12], existen un gran número de algoritmos que se han reportado en la literatura para la compresión de señales de voz, los cuales se pueden clasificar en tres grupos : los codificadores de voz (vocoders), que codifican una serie de parámetros que representan al sistema de producción de la voz en el ser humano; los codificadores de forma de onda, que como su nombre lo indica, codifican directamente la forma de onda de la señal y los híbridos, que son resultado de la combinación de los anteriores.

Los codificadores de voz, tienen la ventaja de que al transmitir sólo un conjunto de n parámetros que representan a M muestras, donde $n << M$, la relación de compresión que se obtiene es muy grande, sin embargo tienen la desventaja de que es muy difícil identificar a la persona que habla.

Por otro lado, los codificadores de forma de onda, presentan ventaja en cuanto a la calidad de la señal en el receptor, pero esta

calidad es directamente proporcional a la cantidad de información que hay que transmitir, no permitiendo grandes relaciones de compresión.

En [12] se presenta un esquema que permite comprimir señales de voz para velocidades de transmisión menores a 9600 bits por segundo, con el cual se obtiene una muy buena calidad en la señal recuperada en el receptor, además se plantea una posible arquitectura para la implantación en tiempo real del esquema. Figura 3.1.

El principal inconveniente de este esquema es precisamente su posibilidad de implantación en tiempo real, ya que requiere de un conjunto de varios procesadores operando en paralelo. Actualmente podía proponerse otra arquitectura (con procesadores más modernos) para su implantación, sin embargo, dentro de los algoritmos que están involucrados en el esquema, uno de ellos, el escalamiento armónico en el dominio del tiempo, es "costoso" en términos computacionales para la operación del sistema en tiempo real, pensando en que se requiere almacenar un gran número de muestras para efectuar el proceso de reducción y en el de expansión. Este algoritmo es uno de los puntos importantes del esquema que permite lograr velocidades de transmisión de 4800 bps con buena calidad.

Otra condición deseable en un sistema de compresión en tiempo real es que el mismo procesador de señales efectúe la compresión y la expansión, cosa que en el esquema presentado en [12] requeriría varios procesadores para la compresión y varios para la expansión. Para que el

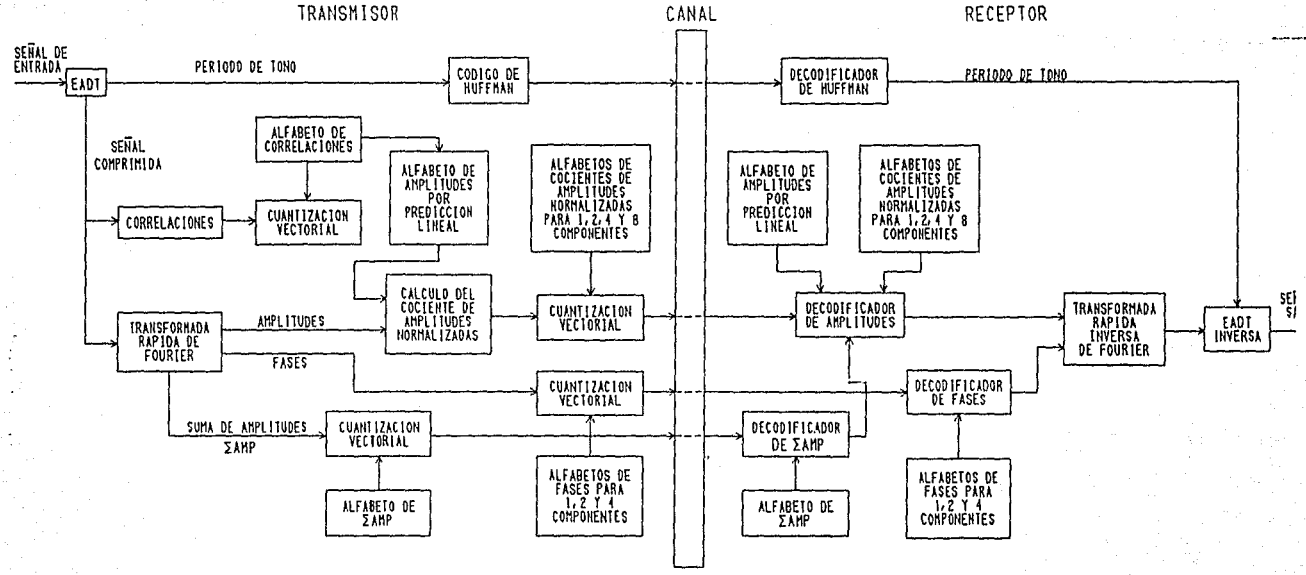

Figura 3.1 Sistema de Compresión

mismo procesador realice todo el proceso, el algoritmo de compresión y el de expansión deben poder ejecutarse en un intervalo de tiempo dado por la frecuencia de muestreo y el número de muestras por marco de análisis utilizadas. Por ejemplo, si el marco de análisis es de 128 muestras, que fueron muestreadas a 8000 muestras/segundo se dispone de 16 ms para efectuar codificación de un bloque a transmitir y la decodificación de un bloque recibido.

Pensando en lo anterior, el sistema objeto de esta tesis puede ser implantado en un procesador de señales, ya que los algoritmo seleccionados permiten la operación en tiempo real del mismo. El sistema que se describirá está basado en el esquema presentado en [12], al cual se le han hecho modificaciones sustanciales para cumplir con los objetivos mencionados y además se ha mejorando el desempeño del sistema al proponer una nueva estrategia de distribución de bits.

III.3 DESCRIPCION DEL SISTEMA.

Como se ha mencionado, el sistema que se presenta está basado en la codificación por transformada y cuantización vectorial, a continuación se describen cada uno de los algoritmos involucrados en el sistema, así como los parámetros que se utilizan en cada uno de ellos y las consideraciones prácticas realizadas. Inicialmente se comentará la operación en forma general, para posteriormente mencionar la operación en punto fijo en el procesador de señales.

Al utilizar codificación por transformada, como de mencionó en el primer capítulo, se forma un vector de muestras, el cual se multiplica por la matriz de transformación para obtener los coeficientes transformados. En este caso en particular, el vector se formará por muestras de la señal de voz. El tamaño N de este vector puede variar, para efectos de la descripción del sistema se utilizará $N = 128$.

III.3.1 TRANSMISION.

DETERMINACION DEL NUMERO DE BITS PARA CADA MARCO DE ANALISIS.

Para determinar el número de bits disponible que se tiene en cada marco de análisis, se debe conocer la tasa de transmisión del canal, por ejemplo 16000 bits por segundo, así como la frecuencia de muestreo que se utilizó para la conversión de la señal analógica a digital, por ejemplo 8000 muestras por segundo. El cociente de las anteriores, determina el número de bits disponibles para codificar una muestra, continuando con el ejemplo, 2 bits/muestra. Conociendo el número de muestras representado en cada segmento de análisis, al multiplicarlo por el número de bits/muestra, se obtiene el número de bits disponible para codificar ese segmento de señal. Para este sistema, se van a estar transmitiendo bloques de 120 muestras cada vez, por lo tanto se dispone de 240 bits para cada marco de análisis. Con este número de bits se debe

codificar todo lo necesario para que el receptor pueda recontruir el segmento en cuestión.

EXPANSION DE LA SEÑAL MUESTREADA.

Es la primera operación que se realiza sobre la señal, esta operación se incluyó debido a que los datos provienen de un convertidor analógico-digital no lineal, el cual entrega cada muestra representada con 8 bits, en formato conocido como ley μ. Este formato ha sido ampliamente utilizado y existe bastante documentación al respecto [12],[14].

Cada muestra que el convertidor va entregando, se convierte a formato lineal para poder ser procesada.

MULTIPLICACION POR UNA VENTANA

Dos de los algoritmos que se utilizan en el sistema, requieren para obtener un mejor desempeño, que las muestras sobre las cuales van a operar sean previamente multiplicadas por una ventana, permitiendo trabajar con traslapes entre las muestras a analizar.

Para la operación de este sistema se seleccionó una ventana trapezoidal de la forma

$$w[n] = \begin{cases} \dfrac{n}{M+1} & 1 \leq n \leq M \\ 1.0 & M+1 \leq n \leq N-M \\ 1.0 - \dfrac{n-N+M}{M+1} & N-M+1 \leq n \leq N \end{cases} \qquad (3.8)$$

donde M es el número de muestras que forman la pendiente, que en nuestro caso fueron 8, con un traslape de 4 muestras entre ventana y ventana; N el número total de muestras en la ventana. Figura 3.2.

$$xw[n] = x[n] \cdot w[n] , \quad n = 1, \ldots, N \qquad (3.9)$$

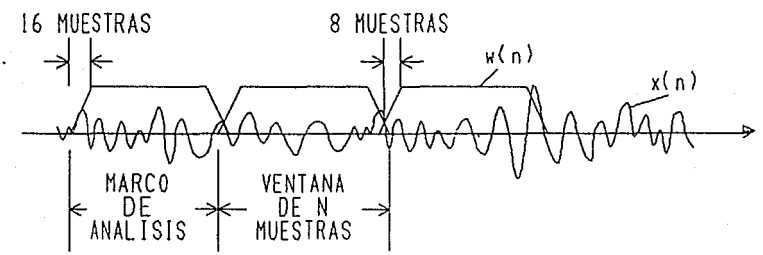

Figura 3.2 Segmentación de la señal

NORMALIZACION DE LA SEÑAL.

Las muestras expandidas a formato lineal con el primer algoritmo, están ahora representadas en 13 bits magnitud signada, lo cual corresponde a un rango dinámico de la señal entre 8031 y -8031, con 256

posibles valores dentro de ese rango. El trabajar con números en ese rango para lenguajes de alto nivel en punto flotante es bastante simple, sin embargo para cumplir con el objetivo de que el sistema pueda ser implantado en un procesador de señales que trabaja en aritmética de punto fijo, es conveniente normalizar la señal con respecto al valor máximo presente en el marco de análisis, con el fin de tener la máxima precisión posible al efectuar las operaciones en esta aritmética.

El valor máximo está dado por :

$$x_{max} = \max \{ | x_w[n] | ;\ n = 1, \ldots, N \} \qquad (3.10)$$

donde x_{w_i} son las componentes del marco de análisis en formato lineal después de haber sido multiplicadas por la ventana y $|a|$ representa al valor absoluto de a.

Y la señal normalizada x_n es

$$x_n[n] = \frac{x_w[n]}{x_{max}} , \quad n = 1, 2, \ldots, N \qquad (3.11)$$

CODIFICACION DEL VALOR MAXIMO

El receptor debe saber el valor que se utilizó para normalizar la señal original. Dado que el valor máximo que se utilizó para normalizar la señal pertenece al conjunto de 256 posibles valores que puede entregar el convertidor, el valor se puede codificar directamente como

su valor en ley μ, ya que como se mencionó, la ley μ representa cada muestra en 8 bits. En este caso sólo se requieren 7 bits, que corresponden a la magnitud, ya que el máximo siempre es positivo.

En este caso se tiene una tabla con los posibles valores que puede tomar el máximo, de donde, utilizando una búsqueda binaria se obtiene el índice del valor máximo utilizado. Este índice debe transmitirse al receptor.

$$\hat{x}_{max} = Q_{max} [x_{max}] \qquad (3.12)$$

donde a $\hat{x}_{max}$ le corresponde la i-ésima posición en el cuantizador I_{max}, que es el índice que se transmite al receptor y Q_{max} es el cuantizador del valor máximo.

CALCULO DE CORRELACIONES

En este algoritmo se calcula la autocorrelación de la señal, pues a partir de ella se obtendrán los coeficientes del sistema de predicción lineal. La autocorrelación se calcula de la siguiente manera :

$$r(i) = \sum_{k=0}^{N-1-1} x_n[k]\, x_n[k+i] \qquad (3.13)$$

Se calculan p = 11 correlaciones para i = 0, 1, ..,p-1 y para cada segmento, debido a que el algoritmo que determina los coeficientes del sistema de predicción lineal requiere una correlación más que el orden del último; el cual generalmente es 10.

CALCULO DE LA TRANSFORMADA DE FOURIER DE LA SEÑAL.

En este caso se emplea el algoritmo de la Transformada Rápida de Fourier para mapear las muestras del dominio del tiempo al dominio de la frecuencia.

En el primer capítulo se mencionó que la transformada óptima es la transformada por componente principal, pero que su desventaja es la dificultad de realización. Después de la transformada por componente principal, la que le sigue en desempeño es la transformada coseno que ya cuenta con algoritmos para su cálculo eficiente y a continuación la transformada de Fourier, que cuenta con el algoritmo de transformada rápida.

La elección de la transformada de Fourier como matriz de transformación se debió a dos situaciones:

- Cuando la matriz de transformación se aplica a secuencias grandes, la transformada de Fourier, al igual que la transformada Coseno se aproximan asintóticamente al desempeño de la transformada óptima, en este sistema se utilizan secuencias de longitud 128 o 256, que de acuerdo con los resultados reportados en [1], corresponden a longitud de secuencia grande.
- Existe mucha información para la implantación de la transformada rápida de Fourier en procesadores de señales [14], lo cual simplifica la realización del sistema en punto fijo.

Se utiliza la técnica de inversión de bits, para ordenar las muestras de entrada, de tal forma que las componentes del espectro que se obtienen después de la transformación están ordenadas. Se seleccionó orden 7 para el cálculo de la TRF, se tienen entonces 65 componentes de amplitud y 65 de fase a codificar (se debe recordar que la transformada es simétrica, así que sólo se necesitan codificar la mitad más uno coeficientes de amplitud y fase).

Para el cálculo de la TRF en lenguaje de alto nivel, se utiliza la rutina que se encuentra en [15], debida a Cooley et.al.; la cual recibe un vector complejo con las muestras de entrada, las ordena de acuerdo con la inversión de bits y calcula la transformada, regresando en el mismo vector complejo las componentes del espectro .

$$X[\omega] = A \cdot x_n[n] \qquad (3.14)$$

donde A es la matriz de transformación de Fourier.

CAMBIO DE FORMATO DE AMPLITUD Y FASE

En esta etapa, del vector resultado de la transformación, que puede estar representado en amplitud y fase o en parte real y parte imaginaria ($X[\omega]_n$), se obtiene la magnitud al cuadrado de cada componente (a) y la parte real (xx) y parte imaginaria (yy) de cada componente.

Se cuantiza escalarmente con 8 bits la magnitud al cuadrado de cada componente, utilizando una tabla en la que se tienen la raíz cuadrada de los valores del cuantizador ($\hat{a}$) y sus inversos ($\hat{a}_{inv}$). La búsqueda para encontrar el valor de reproducción se realiza en forma binaria.

El hecho de que exista en la tabla además del valor cuantizado su inverso, se debe a que en el procesador de señales que opera en punto fijo, la operación división es bastante costosa, se prefiere sacrificar memoria a sacrificar tiempo de proceso; la otra operación que se evita con este cuantizador es el cálculo de la raíz cuadrada, ya que la entrada al cuantizador es la magnitud al cuadrado y la salida es la raíz cuadrada del valor de reproducción.

Una vez obtenida la magnitud cuantizada para cada componente, se obtiene las componentes parte real ($x_r[n]$) y parte imaginaria ($x_i[n]$) referidas al círculo unitario, esto se logra dividiendo parte real y parte imaginaria entre la magnitud.

También se obtiene la suma de las magnitudes del vector en proceso (S), este valor se utilizará posteriormente.

$$xx = Re\{ X[\omega] \}_n$$

$$yy = Im\{ X[\omega] \}_n$$

$$a = xx^2 + yy^2$$

$$\hat{a} = Q_{mag,1}[a]$$

$$\hat{a}_{inv} = \frac{1}{\hat{a}} = Q_{mag,2}[a] \qquad (3.15)$$

$$x_r[n] = xx \cdot \hat{a}_{inv}$$

$$x_i[n] = yy \cdot \hat{a}_{inv}$$

$$y[n] = \hat{a}$$

para n = 1, ..., N

donde Q_{mag} es el cuantizador para la magnitud, que tiene dos salidas, el valor cuantizado $\hat{a}$ y su inverso $\hat{a}_{inv}$.

$$S = \sum_{n=1}^{N} y[n] \qquad (3.16)$$

CUANTIZACION DE LA SUMA DE AMPLITUDES.

Se procede a cuantizar la suma de las amplitudes del vector en proceso (S), utilizando un cuantizador escalar de 7 bits; nuevamente la estrategia de búsqueda en el cuantizador es utilizando búsqueda binaria.

$$\hat{S} = Q_S [S] \qquad (3.17)$$

donde $\hat{S}$ es el valor cuantizado de S, utilizando el cuantizador de la suma de amplitudes Q_S. Se obtiene también el índice I_S que corresponde a la posición de $\hat{S}$ en el cuantizador, que también se transmitirá al

receptor.

CUANTIZACION DE LAS CORRELACIONES.

Para la codificación de las correlaciones se utiliza la cuantización vectorial, el alfabeto que utiliza el cuantizador vectorial de correlaciones se diseña utilizando el algoritmo LBG [4] descrito en el capítulo II.

Para la generación del alfabeto de correlaciones, se obtuvieron las mismas de seis señales de voz (aproximadamente 10 000 conjuntos de 11 correlaciones), las cuales previamente fueron procesadas de acuerdo con el algoritmo aquí propuesto. Al proceso de obtención de los alfabetos se le conoce como entrenamiento. Se generó un cuantizador vectorial de 9 bits, es decir, se dispone de 512 vectores de 11 correlaciones.

Para el alfabeto de correlaciones, el algoritmo se aplica de la siguiente manera :

Se utiliza un cuantizador de 9 bits, es decir, N=512, K=11, pues es el número de correlaciones que tenemos, n=10000, y $\varepsilon = 0.01$. Se utiliza el error cuadrático medio como medida de distorsión para la separación del espacio

$$d(x,\hat{x}) = \sum_{i=0}^{k-1} |x_i - \hat{x}_i|^2 \qquad (3.18)$$

Con los datos anteriores se inicializa el algoritmo, el desarrollo

del algoritmo es el descrito en el capítulo con una modificación, en la etapa de división (splitting), en lugar de perturbar cada componente del vector de correlaciones, que resultó ser el centroide, se perturban los coeficientes de reflexión obtenidos a partir de este vector de correlaciones, utilizando el algoritmo de Levinson-Durbin (para la obtención de los parámetros de predicción lineal).

Una vez que se tienen los nuevos vectores de coeficientes de reflexión k_i, se calculan los coeficientes a_i a partir de los k_i, y a continuación se calcula la correlación de los coeficientes a_i; de esta forma tenemos los dos vectores de correlación para continuar con el algoritmo. [12]

Los coeficientes de reflexión y la ganancia, se obtienen al mismo tiempo que los coeficientes a_i, utilizando el método de Levinson. Para pasar de los coeficientes k_i a los coeficientes a_i se utiliza la siguiente relación : [17]

$$
\begin{aligned}
a_i^{(i)} &= K_i \\
a_j^{(i)} &= a_j^{(i-1)} + K_i\, a_{i-j}^{(i-1)} \qquad \begin{matrix} i = 1,\ldots,p \\ j = 1,\ldots,i-1 \end{matrix} \\
a_0 &= 1
\end{aligned}
\tag{3.18}
$$

Para calcular la correlación de los coeficientes a_i se utiliza la misma rutina que para la correlación de la señal.

Los alfabetos para los cuantizadores vectoriales sólo se tienen que generar una vez, con ellos se procederá a codificar cualquier señal; así pues, es importante que en el entrenamiento se disponga de un gran número de muestras para que el cuantizador sea lo más general posible.

Resumiendo, el proceso de entrenamiento de correlaciones es :

Se utiliza el algoritmo LBG descrito en el capítulo II, sustituyendo el punto 3 por :

3. División (Splitting). Dada $\hat{A} = \{ y_i, i = 1, \ldots, M \}$, se obtienen de cada y_i los coeficientes de predición lineal utilizando Levinson-Durbin. Los coeficientes k[i] (PARCOR) obtenidos, se divide cada vector k en $k + \varepsilon$ y $k - \varepsilon$ donde ε es un vector de perturbación fijo. De los vectores resultamtes $k + \varepsilon$ y $k - \varepsilon$ se calculan los coeficientes a_i con (3.18). A continuación se obtiene la correlación de los coeficientes a_i con (3.19) resultando los coeficientes ra_i

$$ra(i) = \sum_{k=0}^{N-i-1} a[k]\, a[k+i] \qquad (3.19)$$

para i = 0, ..., p-1; donde p es el orden del LPC e

Se define $\hat{A}_0(2M) = \{ ra^+_i, ra^-_i, i = 1, \ldots, M \}$ y se remplaza M por 2M (ra^+ es el vector de correlaciones resultado de utilizar el vector $k + \varepsilon$, y ra^- de utilizar $k - \varepsilon$). El algoritmo continua normalmente.

La codificación de las correlaciones se hace de la siguiente manera :

Para cada vector de correlaciones de orden 11, se busca aquel vector del alfabeto cuya distancia euclidiana al vector a codificar, es la menor.

$$I_{COR} = \{ q : \min \{ \sum_{j=1}^{p} r[j] \cdot ra_q[j] ; q = 1, \ldots, M \} \} \qquad (3.20)$$

donde r[j] es el vector de correlaciones a cuantizar y ra_q es el q-ésimo vector del alfabeto de correlaciones.

CALCULO DEL ESPECTRO APARTIR DE PREDICCION LINEAL.

Utilizando el vector de correlaciones ($\hat{ra}_{ICOR}$) resultado de la cuantización de las correlaciones, se calcula el espectro de cada segmento. Para ello, se utiliza la rutina de TRF que se utilizó para calcular el espectro apartir de las muestras de la señal.

En este caso el espectro se obtiene apartir de las correlaciones, para poder utilizar el mismo algoritmo de TRF y que los espectros sean directamente compatibles, se necesita formar un vector de 128 componentes. El vector a transformar z se forma de la siguiente manera :

$$z[i] = \begin{cases} ra_{ICOR}[i] & \text{si } 1 \le i \le p \\ 0 & \text{si } p+1 \le i \le N-p+1 \\ ra_{ICOR}[N-i+2] & \text{si } N-p+1 \le i \le N \end{cases} \qquad (3.21)$$

Este vector se transforma utilizando (3.14). Del vector resultante

se cuantiza la parte real de cada componente, utilizando un cuantizador escalar de 8 bits, en donde la búsqueda se realiza en forma binaria. Además de forman dos vectores, Z_{LPC} y Z_{LPCI}, con la magnitud del espectro y su inverso respectivamente, y la suma de las componentes de amplitud S_{LPC}.

$$
\begin{aligned}
Z_{\omega}[n] &= A \cdot z[n] \\
H &= Re\{ Z_{\omega}[n] \} \\
\hat{H} &= Q_{LPC}(H) \\
\hat{H}_{inv} &= Q_{LPC}(H) = \frac{1}{\hat{H}} \\
Z_{LPC}[n] &= \hat{H} \\
Z_{LPCI}[n] &= \hat{H}_{inv}
\end{aligned}
\qquad (3.22)
$$

para $n = 1, \ldots, N$

$$
S_{LPC} = \sum_{n=1}^{N} Z_{LPC}[n] \qquad (3.23)
$$

SEÑAL A CODIFICAR.

En este sistema no se codifican directamente las componentes de amplitud de la transformada de Fourier de la señal original, se codifica una relación entre las amplitudes de los espectros obtenidos a partir de la TRF y a partir del sistema de predicción lineal.

Al principio de la descripción del algoritmo, se calculó el número

total de bits de que se dispone para codificar cada marco de análisis, este número de bits se distribuye de la siguiente manera, de acuedo con la información que se tiene que enviar al receptor para que este pueda reconstruir la señal :

- Indice del valor máximo del marco de análisis

  I_{max} ..... 7 bits

- Indice de la sumatoria de amplitudes de la TRF

  Is ..... 7 bits

- Indice de la correlación del marco de análisis

  I_{COR} ..... 9 bits

- Indices de información de amplitudes

  I_{AMP_i} ..... 108.5 bits

- Indice de información de fases

  I_{FAS_i} ..... 108.5 bits

En la tabla anterior se ha considerado que después de codificar valor máxmimo, correlaciones y sumatoria de amplitudes, los bits restantes (217) se destinan 50% a amplitudes y 50% a fases.

La relación a codificar es la siguiente :

$$AMP\,[i] = \frac{y[i]\;\; S_{LPC}}{S\quad Z_{LPC}[i]} \qquad i = 1,\ldots,N \tag{3.24}$$

donde y[i] es el vector de amplitudes del espectro de la TRF y $Z_{LPC}[i]$

es el vector de amplitudes del espectro obtenido por predicción lineal, S· la sumatoria de amplitudes de la TRF y SLPC la equivalente para el espectro obtenido por predicción lineal.

La señal resultante AMP[i], hablando de lo que representan los elementos que la formaron, representa a la fuente que exita al sistema de producción de voz en el ser humano (espectro obtenido apartir de predicción lineal) que genera la señal de voz final (espectro obtenido por TRF).

Esta señal se va a cuantizar utilizando cuantización vectorial. Específicamente se van a utilizar cuatro o cinco cuantizadores vectoriales de tasa variable. En [12] se obtuvieron buenos resultados utilizando está técnica, aunque se tenían deficiencias en la forma de distribución de los bits disponibles para cada marco de análisis..

DISTRIBUCION DE BITS PARA AMPLITUDES.

En el capítulo II de la presente tesis, se realizó una revisión de algunos de los métodos para distribución de bits que se han propuesto en la literatura. Unos de ellos no fueron diseñados para trabajar con diferentes cuantizadores y diferente número de bits por cuantizadores. El último esquema presentado, propuesto por Shoram y Gersho es el que resuelve el problema de asignación óptima de bits para diferentes cuantizadores y diferentes tasas.

El utilizar cuantización vectorial en lugar de cuantización escalar para codificar las componentes de la relación de amplitudes tiene ventajas desde el punto de vista de implantación de algoritmos de distribución de bits como los presentados por Gersho, et.al., Segall y Cuperman. Con cuantizadores vectoriales se puede aproximar a una distribución de bits cercana a la óptima, gracias al eficiente redondeo que puede existir; en el caso vectorial, el redondeo debe realizarse una vez cada vector, en el caso escalar el redondeo es cada muestra. Más aún, los cuantizadores vetoriales permiten asignaciones de fracciones de bit, cosa que los escalares no.

El algoritmo de distribución presentado en [12] se basa en el área que ocupa cada componente del espectro de amplitud obtenido apartir de predicción lineal. De acuedo con las gráficas presentadas, el trabajo realizado en la codificación de la relación de amplitudes es bastante bueno, sin embargo en las fases, se pierde bastante calidad.

El algoritmo presentado por Shoram y Gersho es el mejor de todos los descritos, pues como se mencionó opera para varios cuantizadores, de dimensión variable y tasa variable, sin embargo es un algoritmo costoso computacionalmente, ya que involucra una serie de iteraciones para la obtención de la distribución óptima, debemos recordar que uno de los objetivos del sistema es poder ser implantado en un procesdor de señales; así que debido a que se desea un algoritmo de distribución de bits relativamente rápido, ya que se tienen otros algoritmos que van a necesitar tiempo (como la cuantización vectorial y la transformada de

Fourier) presentamos un algoritmo de distribución que tiene un mejor desempeño que el algoritmo presentado en [12] y que es eficiente en cuanto a cálculo se refiere.

Se parte del hecho de que la distorsión puede expresarse en los siguientes términos :

$$D[i] = k[i]\, 2^{-Ri} \qquad i = 1, 2, \ldots, N \qquad (3.25)$$

donde D[i] es la distorsión en la i-ésima componente, R_i es el número de bits utilizado y k[i] es la variancia del coeficiente i.

Suponemos que las variancias de todos los elementos del vector a codificar es la misma e igual a k.

Para el cálculo de la distorsión en un cuantizador vectorial se han desarrollado algunas expresiones, como la que utiliza Cuperman para su algoritmo de distribución, sin embargo, tratando de simplificar la operación de sistema y observando la similitud de comportamiento que existe entre la función de distorsión y el inverso del espectro obtenido por predicción lineal, en cuanto a la forma de la señal, se propone con fines de obtener una distribución de bits que se aproxime a la óptima, calcular la distorsión como una proporción del inverso del espectro

$$D[i] = \alpha\, Z_{LPC}[i] \qquad (3.26)$$

Susutituyendo las consideraciones anteriores en 3.24 y despejando el número de bits de la componente i

$$R_I = \log_2 \frac{k}{\alpha \, Z_{LPCI}[i]} \qquad (3.27)$$

Por otro lado sabemos que se debe cumplir que la suma de todos los bits asignados a un vector debe ser igual al total de bits disponibles

$$R_T = \sum_{i=1}^{N} R_I$$

$$R_T = \sum_{i=1}^{N} \log_2 \frac{k}{\alpha \, Z_{LPCI}[i]} = \sum_{i=1}^{N} \log_2 \frac{k}{\alpha} - \sum_{i=1}^{N} \log_2 Z_{LPCI}[i]$$

$$R_T = N \log_2 \frac{k}{\alpha} - \sum_{i=1}^{N} \log_2 Z_{LPCI}[i]$$

de donde

$$\log_2 \frac{k}{\alpha} = \frac{R_T + \sum_{i=1}^{N} \log_2 Z_{LPCI}[i]}{N} = cte \qquad (3.28)$$

y sustituyendo en 3.27

$$R_I = \frac{R_T + \sum_{i=1}^{N} \log_2 Z_{LPCI}[i]}{N} + \log_2 \frac{1}{Z_{LPCI}[i]} \qquad (3.29)$$

en términos del espectro se tiene

$$R_i = \frac{R_T - \sum_{i=1}^{N} \log_2 Z_{LPC}[i]}{N} + \log_2 Z_{LPC}[i] \qquad (3.30)$$

En la expresión 3.30 puede ocurrir que R_i sea menor que cero para varias componentes, lo cual provocaría un pobre desempeño en el sistema. Utilizando 3.30 y con a observación anterior se propone el siguiente algoritmo para obtener la distribución de bits :

1. Ordenar las componentes del vector Z_{LPC} de mayor a menor. Con esto se asegura que aquellas componentes de mayor magnitud, que implica menos distorsión dispongan de bits para su codificación. Se inicializan las variables : IND = 1, S_{ESP} = 0.
2. Se repiten los siguientes pasos para n = 1, ..., N

2.1. Calcular el logaritmo base 2 de la componente $Z_{LPC}[n]$. Este paso se realiza en el sistema utilizando una tabla, es un cuantizador escalar de 8 bits, el cual cuando recibe como entrada $Z_{LPC}[n]$ entrega $\log_2 Z_{LPC}[n]$. La búsqueda, al igual que en los otros casos de tablas, se realiza en forma binaria.

$$Z_{LOG}[n] = \log_2 Z_{LPC}[n] \qquad (3.31)$$

2.2. Se actualiza la sumatoria de las componentes del vector de espectro

$$S_{ESP} = S_{ESP} + Z_{LOG}[n] \qquad (3.32)$$

2.3. Se evalúa la expresión 3.30 considerando que sólo existen n componentes en el vector

$$Fac = \frac{R_T - S_{ESP}}{IND} + Z_{LOG}[n]$$

Si Fac > 0 entonces se incrementa IND y n, se repite 2.1.

Si Fac ≤ 0 entonces se elimina esa componente de la sumatoria

$$S_{ESP} = S_{ESP} - Z_{LOG}[n] \qquad (3.33)$$

Se repite 2.1.

3. Se calcula el número de bits no asignados o que faltan. Estos bits se distribuirán equitativamente entre las componentes que participaron en la distribución. El número de bits que se agregará o que se sustraerá a cada componente está dado por

$$Ba = (R_T - S_{ESP}) / IND$$

4. Se realiza la asignación de bits a cada componente

$$B[n] = Ba + Z_{LOG}[n]$$

Si B[n] < 0 entonces B[n] =0

Este es el algoritmo que se utiliza en el sistema para distribuir los bits disponibles entre las componentes de la relación de amplitudes a codificar.

CODIFICACION DE LA RELACION DE AMPLITUDES.

Para codificar la relación de amplitudes, se segmenta la misma en grupos de 8 componentes, para cada grupo se realiza el siguiente proceso: se suma el número de bits que corresponde a las ocho componentes del vector AMP, si la suma de bits para el bloque es menor a 8 se codifican esas componentes vectorialmente utilizando un alfabeto de

8 componentes, si es mayor a 8, se divide en dos grupos de 4 componentes, para cada grupo se suman los bits de sus componentes, si la suma del número de bits de las cuatro componentes es menor a 8, se codifica vectorialmente con un alfabeto de 4 componentes, si es mayor de 8, se divide en grupos de 2 componentes, si la suma del número de bits es menor de 8, se codifica vectorialmente, en caso contrario las componentes se codifican escalarmente, permitiendo como máximo 8 bits por componente, si sobran bits, debido al redondeo, estos se asignan a su grupo adyacente.

Los bits que sobren después de haber codificado el segmento de 8 componentes, en caso de que suceda, se distribuyen en las componentes que aún no se han codificado. Con este procedimiento se codifican todas las componentes del vector AMP. Como se divide el vector en grupos con número par de componentes, la componente que no se incluye en ningún grupo (la última) se codifica escalarmente con los bits que sobraron (máximo 8). Si al terminar de codificar el segmento, sobran bits, estos se agregan a los asignados para codificar la parte real y parte imaginaria. [12]

De lo anterior se deriva que deben existir alfabetos de la relación de amplitudes escalar, vectorial de 2 componentes, vectorial de 4 componentes y vectorial de 8 componentes, para número de bits variable de 0 a 8 bits.

Para la generación de estos alfabetos, se utiliza el mismo

algoritmo descrito en el capítulo II (LBG); en este caso no se hace ninguna modificación. Se aprovecha la característica de ese algoritmo de generar sucesivamente los alfabetos para diferente número de bits, es decir, al calcular los alfabetos para 8 bits se tuvieron que generar los alfabetos de 0 a 7 bits. Se utiliza N = 256, K = 1, 2, 4 y 8; n = 50000, $\varepsilon \approx 0.01$. La medida de distorsión es error cuadrático medio.

El proceso de codificación, es de forma similar a las correlaciones. Se calcula la distancia euclidiana entre el vector a codificar y los existentes en el alfabeto, aquel vector del alfabeto, cuya distancia al vector a codificar es mínima, se define como el vector de reproducción. Se transmite el índice que corresponde al vector seleccionado.

$$I_{AMP_n} = \{ q : \min \{ \sum_{i=1}^{m} (xx[i] - Qm_q[i])^2 ; q = 1, \ldots, M \} \} \quad (3.34)$$

donde

xx es el vector de m componentes a cuantizar vectorialmente

$Qm_q[i]$ es el valor cuantizado de la componente i del vector q

M es el número de vectores de m componentes que forman el cuantizador de $\log_2 M$ bits.

CODIFICACION DE PARTE REAL E IMAGINARIA.

En lugar de codificar la fase de la transformada de Fourier se codifica la parte real y parte imaginaria de la transformada referidas

al círculo unitario. La codificación se lleva a efecto utilizando un cuantizador escalar.

Para la distribución de bits para la parte real e imaginaria, se utiliza el mismo algoritmo que se describió para la relación de amplitudes, con la excepción de que en lugar de utilizar el vector de espectro obtenido por predicción lineal, se utiliza el vector de relación de amplitudes ya cuantizado. A la primera y última componente se les asigna un bit, ya que siempre toman el valor Π o $-\Pi$.

El cuantizador escalar se diseña para b = 0, 1, ..., 8 bits, dividiendo el círculo unitario en 2^b porciones, de cada segmento se calcula la parte real y la imaginaria, para b = 0 se selecciona un valor aleatorio.

El proceso de codificación es como sigue :

Se busca en el alfabeto del cuantizador aquella pareja de parte real y parte imaginaria que es la más cercana a la pareja a cuantizar, obteniéndose el índice de la más cercana I_{FAS}, valor que se va a transmitir al receptor.

$$I_{FAS_n} = \{ q : \min \{ |\phi m_q[q] - \phi| ; q = 1, \ldots, M \} \} \qquad (3.35)$$

donde

$\phi_{m_q}[q]$ es el q-ésimo ángulo formado por la parte real e imaginaria cuantizadas que pertenecen al cuantizador de m bits.

ϕ es le ángulo formado por la parte real e imaginaria a cuantizar.

M es el número de niveles del cuantizador, igual a 2^m.

Después de utilizar los algoritmos descritos anteriormente, se tiene la información a transmitir :

- Indice de valor máximo del marco de análisis.
- Indice del valor de la suma de amplitudes de la TRF.
- Indice de la correlación del marco de análisis.
- Indices de relación de amplitudes.
- Indices de parte real y parte imaginaria.

Con esta información, el receptor debe poder reconstruir la señal, y lo más importante, ésta debe tener una calidad aceptable.

A continuación se describe el proceso en el receptor.

III.3.2 RECEPCION.

El receptor dispone de los indice mencionados al final del inciso anterior, con los cuales debe reconstruir la señal de voz.

DISTRIBUCION DE BITS PARA RELACION DE AMPLITUDES.

Apartir del indice de correlaciones, el receptor puede calcular la distribución de bits que se utilizó para codificar la relación de amplitudes en el transmisor, de la misma manera que se describió en la sección anterior. Ya que conociendo la distribución de bits empleada en la codificación así como el algoritmo de codificación, sabe como

vienen codificadas las componentes de la relación de amplitudes.

$$\hat{ra} = Q_{COR}[I_{COR}] \qquad (3.36)$$

DECODIFICACION DE RELACION DE AMPLITUDES

El receptor debe calcular la distribución de bits y ejecutar el algoritmo de agrupamiento de componentes mencionado en el transmisor, para proceder a decodificar las componentes de la relación de amplitudes.

La decodificación consiste en buscar en el cuantizador vectorial de m bits, aquel vector que corresponda al índice recibido

$$\hat{AMP} = Q_{AMP}^{-1}[I_{AMP}] \qquad (3.37)$$

DISTRIBUCION DE BITS PARA FASES.

Una vez reconstruído el vector de relación de amplitudes, se calcula la distribución de bits que se utilizó para la codificación de la parte real e imaginaria de las fases en el transmisor. EL algoritmo de distribución es el mismo que para el transmisor. El objetivo es conocer la asignación de bits empleada para proceder a la decodificación de la parte real e imaginaria de las fases de la TRF.

DECODIFICACION DE FASES.

Una vez conocida la asignación de bits de las fases, con los índices recibidos, es fácil reconstruir el vector de parte real e imaginaria referidas al círculo unitario que se codificó en el transmisor.

$$\hat{xx} = Q_{FAS}^{-1}[I_{FAS}] \qquad (3.38)$$

Con el paso anterior se obtienen las componentes parte real e imaginaria de cada fase, apartir de ellas se calcula la fase correspondiente.

OBTENCION DE LA TRANSFORMADA DE FOURIER DE LA SEÑAL ORIGINAL.

Como se ha venido mencionando, se codifica una relación de amplitudes normalizadas entre el espectro obtenido por predicción lineal y el obtenido utilizando la TRF. Para recuperar la señal original, se debe primero determinar la transformada de Fourier de la señal. Esto es fácil ya que se cuenta con el espectro por predicción lineal y con el índice de sumatoria de amplitudes de la TRF. La obtención de la transformada se realiza de la siguiente manera

$$\hat{S} = Q_s[\ I_s\ ] \qquad (3.39)$$

$$y[i] = \frac{\hat{AMP}[i]\ \hat{S}\ Z_{LPC}[i]}{S_{LPC}} \qquad (3.40)$$

para i = 1, 2, ..., N

TRANSFORMADA INVERSA DE FOURIER.

Con las amplitudes y fases decodificadas, se puede calcular la transformada inversa de Fourier, de tal forma de obtener la señal original. Para calcular la transformada inversa, se utiliza nuevamente la rutina de TRF. La forma de hacerlo es la siguiente :

-Se forma un vector complejo con las componentes de amplitud y fase decodificadas. En el sistema son 65 componentes de amplitud y 65 de fase.

-Las primeras 65 componentes son los valores que se han decodificado, las componentes 66 a 128 es la parte simétrica del espectro, la componente 66 es igual a la 64, la 67 a la 63 y así sucesivamente.

-Finalmente, se obtiene el conjugado de las componentes, formando así un vector complejo de 128 componentes, con el cual la rutina de TRF obtiene la señal discreta original.

$$\hat{x} = A^{-1} y \qquad (3.41)$$

Debido a que para el cálculo de la transformada de Fourier de la señal se multiplicó la señal por una ventana de tipo trapezoidal, el resultado de la transformada debe procesarse para eliminar el efecto del traslape de bloques. Esto se realiza promediando las componentes que están en involucradas en el traslape.

RECONSTRUCCION DE LA SEÑAL ORIGINAL.

Al obtener la transformación inversa de las componentes de amplitud y fase, se obtiene la señal normalizada original. Con el índice del valor máximo del marco, se puede recuperar la señal original en formato lineal.

$$\hat{x}_{max} = Q_{max}^{-1} [I_{max}] \qquad (3.42)$$

$$\hat{x}_r = \frac{\hat{x}}{\hat{x}_{max}} \qquad (3.43)$$

COMPRESION DE LA SEÑAL.

Recordando que el tipo de convertidor que se utiliza para la conversión A/D y D/A es un convertidor no lineal (ley μ), antes de enviar la señal al convertidor, se debe comprimir ésta utilizando el algoritmo de ley μ.

Una vez efectuada la compresión, se envía al convertidor D/A para que se reproduzca la señal análogica recuperada y se pueda establecer en forma subjetiva la calidad de la señal de voz que ha sido reconstruída.

Con el inciso anterior, termina la descripción del algoritmo de Codificación por Transformada de Señales de Voz Utilizando Cuantización Vectorial, objeto de esta tesis.

IV. OPERACION DEL SISTEMA

IV.1 DESCRIPCION DE LA OPERACION DEL SISTEMA

El sistema fué desarrollado inicialmente en lenguaje de alto nivel, Fortran 77, en una computadora Vax. Cada uno de los algoritmos que se describieron en el capítulo anterior, está programado en forma independiente, el hecho de tener un programa por cada algoritmo beneficia la depuración del sistema, ya que si se desea probar una variante en alguno de los algoritmos, sólo se modifica ese programa, y no hay necesidad de compilar todo el sistema. De esta forma, el primer algoritmo se aplica a toda la señal, el resultado de éste es entrada para el siguiente algoritmo, que procesa toda la señal, y así sucesivamente.

La segunda versión del sistema, está codificada en lenguaje de alto nivel, Fortran 77, pero para una computadora personal, en este caso es

Fortran 77 Microsoft V4.1. El program simula la operación en linea del sistema, esto es, toma un segmento de señal (128 muestras) y efectúa el proceso de compresión - expansión, y regresa a procesar otro segmento de señal.

La tercera versión del sistema, es en punto fijo, aunque no se dispone del sistema completo, es decir, sólo se tienen codificados varios de los algoritmos, operando en línea.

Todas las versiones reciben los mismos parámetros, que permiten configurar al sistema. Estos parámetros son los siguientes (se utilizarán como ejemplo los valores para codificar una señal de 16 segundos de duración) :

Número de muestras de análisis : 120

Número de muestras de ventana : 128

Número de bloques a procesar (120 muestras) : 1093

Orden de la Transformada de Fourier : 7

Orden de Predicción Lineal : 10

Ancho de banda del canal : 16000

Frecuencia de muestreo : 8000

Número máximo de bits por componente a cuantizar : 8*

* puede seleccionarse el mismo valor para todos los cuantizadores, o fijar el valor para cada uno.

Además de los parámetros anteriores, también debe proporcionarse el nombre del archivo a procesar, y en caso de cambiar, por ejemplo, el orden de las correlaciones, se debe dar el nombre del archivo donde se encuentran los alfabetos correspondientes para los cuantizadores vectoriales.

Todos los sistemas entregan a la salida la señal comprimida en formato ley μ, para poder ser escuchados directamente utilizando una tarjeta de conversión A/D-D/A con un convertidor no lineal (que utiliza ley μ).

La señales de voz de que se dispone, han sido muestreadas a 8000 muestras por segundo, y son de duración aproximada de 14 segundos y 40 segundos, lo que corresponde a 131072 bytes y 393216 bytes respectivamente.

IV.2 PRUEBAS REALIZADAS Y RESULTADOS OBTENIDOS

Para llegar al esquema final de compresión que ha sido descrito detalladamente, se realizaron un gran número de pruebas tanto subjetivas (escuchar la señal reconstruida) como cuantitativas (medición relación señal a ruido SNR), comparando así los desempeños obtenidos para cada variante propuesta.

A continuación, mencionaremos algunos de los resultados cuantitativos que se fueron obteniendo, básicamente los referentes a las

aportaciones importantes que se realizaron tanto en asignación de bits para amplitudes y fases, como forma de codificación de fases.

Utilizando el sistema descrito, con asignación de bits de acuerdo con el área que ocupa cada componente del área total del espectro de amplitud por predicción lineal, tanto para amplitudes como para fases y utilizando cuantización vectorial para amplitudes y fases de acuerdo a lo propuesto en [12] se obtuvieron los siguientes desempeños:

7 200 bps	6.34 dB
8 000 bps	7.08 dB
9 600 bps	8.39 dB
12 000 bps	10.03 dB
16 000 bps	12.46 dB

Una observación interesante, fué que al modificar el tipo de cuantización de las fases, de vectorial [12] a escalar, el desempeño del sistema "mejoró" a 12.52 dB en la señal a 16 kbps. Este aumento sugirió ver el efecto de la cuantización en las fases, para lo cual, se reconstruyó la señal utilizando las fases originales, el desempeño se incrementó a 19.36 dB. Una diferencia de casi 7 dB, lo cual provocó la búsqueda de otro algoritmo tanto para la distribución de bits a las fases como para su codificación.

Fué en este momento que se investigó la mejor forma de efectuar la asignación de bits. De donde se originó el algoritmo que se describió en el capítulo anterior.

A continuación se incluyó la nueva forma de asignación de bits para las amplitudes, manteniendo la asignación de bits para las fases en forma escalar, donde el desempeño aumentó de 12.52 dB a 12.65 dB. Lo que indica que el trabajo realizado en amplitudes mejoró ligeramente.

El siguiente paso fué el diseño del algoritmo de codificación para fases, pues de acuerdo con las pruebas realizadas, al utilizar cuantización vectorial con la distribución de bits original, no se obtenía un mejor desempeño en el sistema, comparado con cuantización escalar.

Al mismo tiempo que se modificó para las fases el algoritmo de distribución de bits, se modificó también el algoritmo de codificación. El nuevo cuantizador, es escalar, como se detalló en el capítulo anterior, donde se codifican las componentes parte real e imaginaria referidas al círculo unitario de la fase en cuestión. La principal ventaja que se obtiene con este algoritmo es velocidad, ya que el alfabeto del cuantizador se puede ordenar para aplicar una búsqueda binaria y encontrar rápidamente la fase cuantizada; lo cual es muy útil para la implantación en un procesador de señales del sistema.

El resultado obtenido al incluir la modificación anterior fué un aumento en el desempeño, de 12.65 dB a 13.37 dB. Este fué el mejor incremento en desempeño para sistema. Recordemos que se supera el desempeño obtenido con cuantización vectorial anterior y se reduce notablemente la complejidad total del algoritmo.

Se realizaron algunas pruebas adicionales teniendo en mente la posibilidad de que el algoritmo actual no pudiera implantarse en un procesador de punto fijo, desde el punto de vista de tiempo para codificación y decodificación. Las pruebas consistieron en evaluar la importancia de que el espectro obtenido por predicción lineal fuera del mismo orden que el de la transformada de Fourier. Las idea es calcular un espectro por predicción lineal de orden menor al de la TRF y luego interpolar para obtener un espectro de igual número de componentes, los resultados fueron los siguientes :

Espectro LPC orden 7 y TRF orden 7	13.37 dB
Espectro LPC orden 6 y TRF orden 7	13.24 dB
Espectro LPC orden 5 y TRF orden 7	13.10 dB

En caso de que no alcance el tiempo disponible para la codificación y la decodificación con el algoritmo propuesto, se puede modificar el cálculo del espectro reduciendo el orden del mismo, la ventaja que se obtendría es menos tiempo requerido para el cálculo de la TRF, menor tiempo para cuantización, con la disminución consecuente en desempeño.

Hemos mencionado el incremento en desempeño obtenido con el nuevo esquema, es necesario también comentar el incremento en velocidad de cálculo que resultó.

Este incremento en velocidad se debió fundamentalmente, al cambio

de algoritmo para codificación de fases y a que se sustituyeron operaciones "costosas" en términos de velocidad de cálculo, como el logaritmo y la raíz cuadrada por tablas que contienen valores típicos (cuantizador), que se accesan en forma binaria.

El sistema trabajando en la VAX tarda aproximadamente 30 minutos en codificar una señal de 131072 muestras, mientras que el esquema en una PC con procesador 80386 y coprocesador 80387 tarda 11 minutos para la misma señal. El proceso en PC es de aproximadamente 0.66 seg for marco de análisis.

Otra ventaja del esquema es que no se trasmite la distribución de bits al receptor, ésta puede calcularse en el receptor, reduciendo así la información adicional a trasmitir.

A continuación se presentan unas gráficas que muestran para varias tasas de transmisión el desempeño en decibeles que se obtiene al codificar una señal. Las tres señales procesadas fueron grabadas de un teléfono, digitalizadas a 8000 muestras por segundo. La figura 4.1 corresponde a una señal de voz masculina, la 4.2 una voz femenina y 4.3 una combinación. Con fines ilustrativos se muestran en las gráficas el desempeño del sistema al inicio del desarrollo.

Desempeño del Sistema

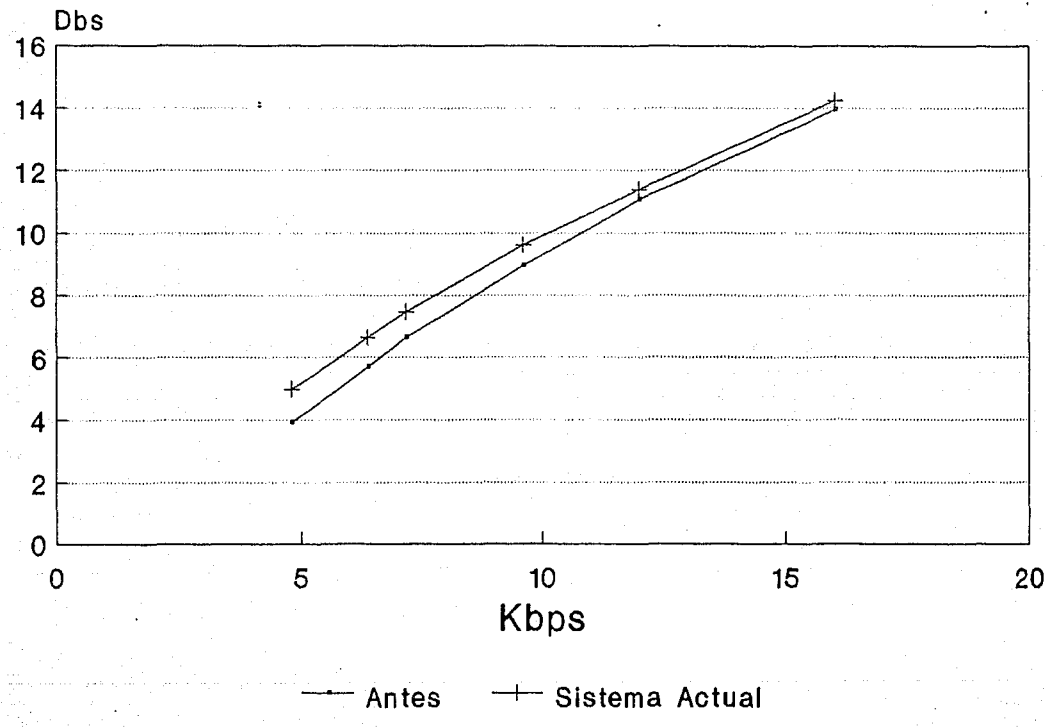

Figura 4.1

Desempeño del Sistema

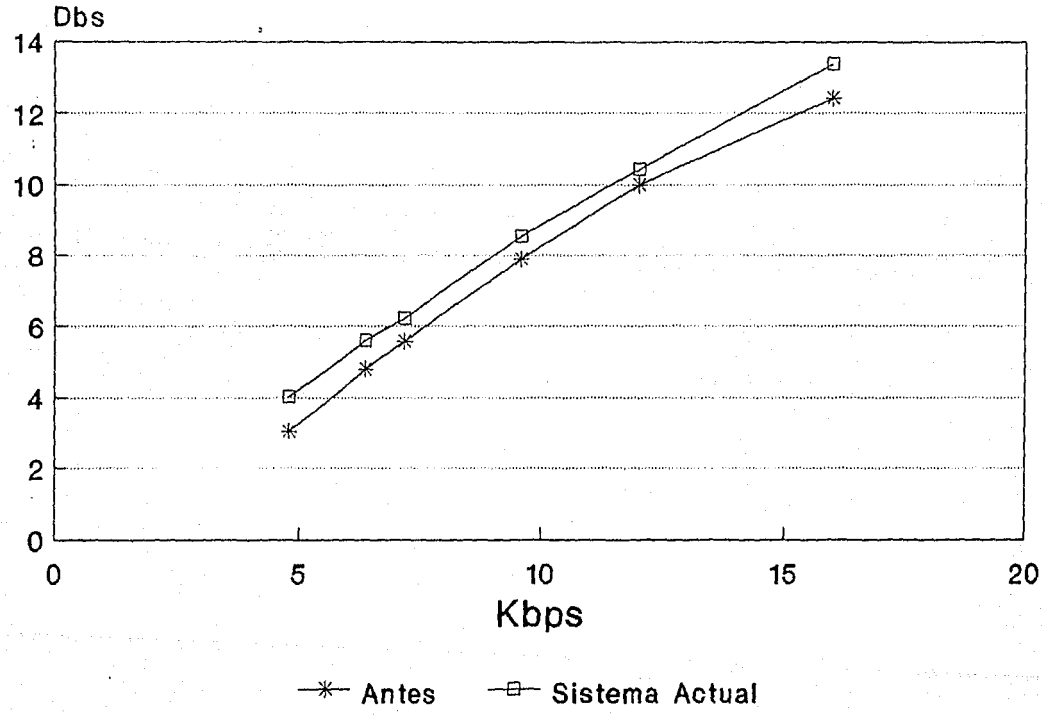

Figura 4.2

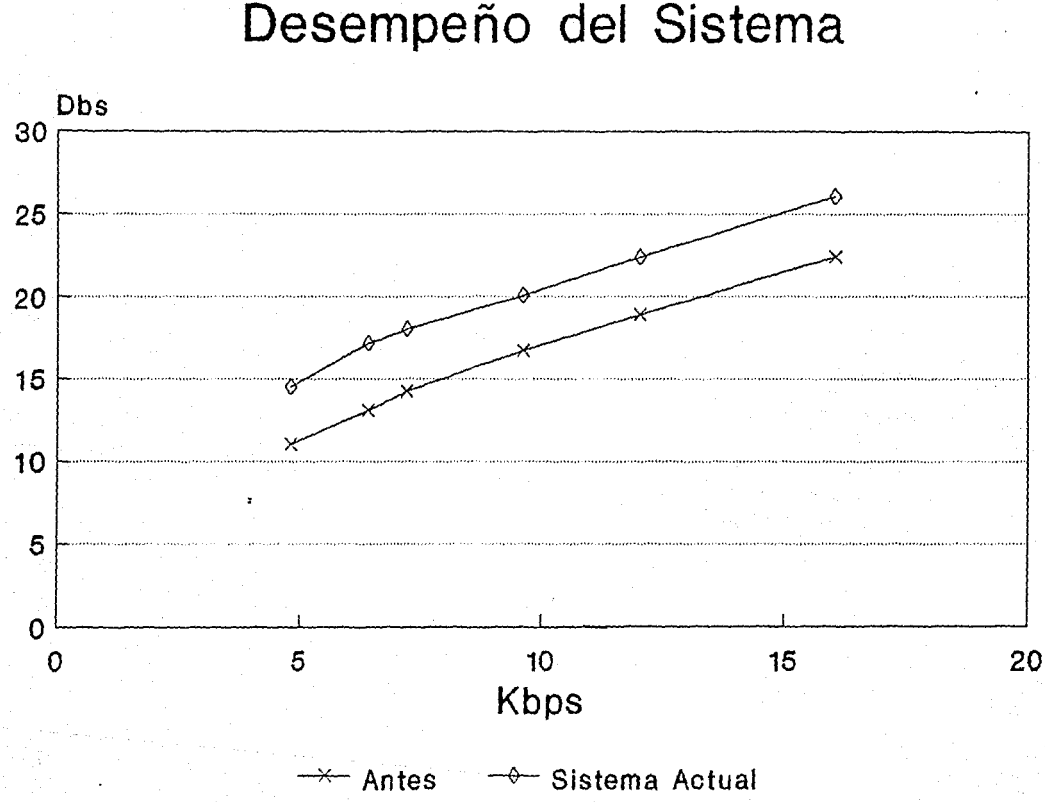
Desempeño del Sistema
Dbs
30
25
20
15
10
5
0
0
5
10
15
20
Kbps
Antes
Sistema Actual

Figura 4.3

IV.3 TRANSMISION DE VARIAS SEÑALES POR UN CANAL

Una aplicación interesante de la trasmisión de señales en forma digital, es la posibilidad de trasmitir ya sea varias señales del mismo tipo o señales de diferente fuente por un mismo canal.

Para esta aplicación, es fundamental saber como asignar del ancho de banda total del canal, que parte corresponde a cada señal a trasmitir.

En el caso de señales de voz, sabemos que durante una conversación, existen muchos espacios en que uno de los interlocutores permanece callado, mientras el otro individuo habla. El objetivo del algoritmo siguiente es aprovechar esos espacios de silencio que ocurren en una de las señales para que se asignen más bits a las señales en las que existe conversación.

La idea es distribuir el total de bits disponibles para trasmitir información por el canal, de una forma adaptable, de tal forma que si sólo se trasmite una señal, se le asignan todos los bits disponibles, pero si se desean trasmitir 2 o 3 señales, se le deben asignar bits a cada señal de acuerdo con alguna medida de si la señal es ruido o es voz.

En este caso, se sugiere utilizar como medida para tomar la decisión del tipo de señal, la energía de la misma, ya que si el valor

es pequeño es muy probable que sea ruido, en caso contrario se trata de voz.

Para el cálculo del indicador se utilizan los coeficientes k[i] obtenidos apartir de las correlaciones con el algoritmo de Levinson-Durbin :

$$E_i = \prod_{j=1}^{p} (1 - k[j]^2) \qquad i = 1, 2, \ldots, n \qquad (4.1)$$

donde p es el orden de las correlaciones, y n el número de señales a trasmitir por el canal.

$$\phi_i = 120 \; / \; E_i, \; i = 1, 2, \ldots, n \qquad (4.3)$$

$$S_\phi = \sum_{i=1}^{n} \phi_i \qquad (4.4)$$

$$\text{No.Bits} = \frac{\frac{120\ n}{S_\phi} * \text{Tasa}}{n\ f_m} - n \text{ Bits asignados} \qquad (4.5)$$

donde los Bits asignados son los que se ocupan para codificar los índices de correlación, valor máximo y sumatoria de amplitudes; f_m es la frecuencia de muestreo utilizada para digitalizar las señales. Se utiliza 120 porque es el valor del marco de análisis utilizado en el

esquema.

El algoritmo anterior se aplicó a tres señales de voz, resultando un desempeño de 4.5 dB, 5.2 dB y 4.4 dB para cada señal respectivamente.

IV.4 COMENTARIOS OPERACION EN PUNTO FIJO.

Se ha mencionado que uno de los objetivos del sistema, es que se pueda implantar en un procesador de señales. Durante la descripción del sistema, se mencionaron algunas consideraciones que se hicieron para tratar de cumplir con ese objetivo.

Algunas de ellas son las siguientes :

- Normalización de la señal en formato lineal con respecto al valor máximo presente en el marco de análisis.

- Utilización de tablas para el cálculo de algunas operaciones como el logaritmo y la raíz cuadrada, que invoucran un gran número de operaciones, además de algunas divisiones como el inverso del espectro por predicción lineal y el inverso de la suma de amplitudes.

- Búsqueda binaria para la cuantización de valores en las tablas mencionadas en el inciso anterior y para la codificación de fases.

Como se mencionó al inicio de este capítulo, se tienen programados en punto fijo (ensamblador TMS 320C25) algunos de los algoritmos que forman el sistema, ellos son :

- Expansión de la señal de ley μ a formato lineal.

Se utiliza el algoritmo publicado en [14]. Existen varias implantaciones de este algoritmo para procesadores de la familia TMS320. Se pasa de una muestra en 8 bits a una muestra en formato lineal representada en 13 bits más signo. (Q13).

- Multiplicación por la ventana.

Los factores de la ventana se almacenan en memoria, para este sistema se almacenan 8 valores en formado Q15.

- Cuantización del valor máximo.

Se dispone de un cuantizador con los valores máximo posibles, como salida del cuantizador se tiene el inverso del valor máximo cuantizado en formato Q15.

- Normalización con respecto al valor máximo del marco de análisis.

Se normaliza la señal multiplicando por el inverso del valor máximo. El resultado se expresa en formato Q15.

- Cálculo de correlaciones.

Se utilizan dos palabras para representar las correlaciones, con formato Q15.

- Cálculo de Transformada de Fourier.

El algoritmo utilizado es el publicado en [14], el cual requiere que la señal de entrada esté en formato Q15 y entrega los coeficientes transformados en parte real y parte imaginaria en formato Q15. En cada etapa de la transformada divide la señal entre dos para evitar

Cuantizador parte imaginaria	1024 localidades

El proceso de tomar cada muestra y expanderla para 256 muestras 11 897 ciclos de reloj.

Cuantización del valor máximo	178 ciclos de reloj.
Normalización de la señal	7 308 ciclos de reloj.
Cálculo de correlaciones	55 539 ciclos de reloj.
Cálculo de la TRF	39 550 ciclos de reloj.
Cambio de parte real e imaginaria	74 607 ciclos de reloj.
Cuantización vectorial de correlaciones	129 160 ciclos de reloj.
Obtención del espectro (LPC)	42 220 ciclos de reloj.

De donde se observa que el algoritmo más pesado es la cuantización vectorial de las correlaciones, ya que en la cosdificación de las amplitudes, en el peor caso se cuantizará un vector de 8 componentes con 7 bits máximo. Se han utilizado 0.018 s de los 0.032 s disponibles.

V. CONCLUSIONES Y RECOMENDACIONES.

1. Se presentó un sistema de compresión de señales de voz, utilizando codificación por transformada y cuantización vectorial, que permite obtener una calidad bastante aceptable a una tasa de transmisión de 16 000 bits por segundo.

2. Se propone un algoritmo para la distribución de bits a utilizar en el proceso de codificación de los coeficientes de la transformada, partiendo de la consideración que la función de distorsión es proporcional al inverso del espectro obtenido por predicción lineal.

3. Para la codificación de las fases de la transformada, se propone utilizar un cuantizador escalar cuyo alfabeto consiste en dividir el círculo unitario en 2^b secciones (b = No. bits), con el cual se incrementa el desempeño del sistema.

4. Para la realización física del sistema de compresión se modificó el cálculo de algunas operaciones "costosas" por tablas de acceso rápido.

5. El sistema está listo para ser implantado en una arquitectura basada en un procesador de señales, para su operación en tiempo real.

6. Se propone una técnica de asignación de recursos de un canal, para trasmitir varias señales de voz por él.

7. Se recomienda realizar los siguientes puntos para continuar con el desarrollo del sistema :

- Proponer una arquitectura para la operación en tiempo real del esquema presentado.

- Concluir con la implantación del algoritmo en punto fijo.

B I B L I O G R A F I A

1. Rainer Zelinski and Peter Noll, " Adaptive Transform Coding of Speech Signals ", *IEEE Transaction on Acoustics, Speech and Signal Processing*, VOL ASSP 25, NO.4, AGOSTO 1977, 299-309.
2. T.J.Lynch, *Data Compression Techniques and Applications*. Lifetime Learning Publications Belmont California, 1985.
3. N.Ahmed, T.Natarajan and K.R.Rao, " Discrete Cosine Transform ", *IEEE Transaction on Computers*, ENERO 1974, 90-93.
4. Y.Linde, A.Buzo and R. M. Gray, " An algorithm for vector quantizer design ", *IEEE Transaction on Communications*, VOL COMM 28, NO.1, ENERO 1980, 84-95.
5. R.M.Gray, " Vector quantization ", *IEEE ASSP Magazine*, ABRIL 1984, 4-29.
6. J.Makhoul, S.Roucos and H.Gish, " Vector quantization in speech coding ", *Proceedings of the IEEE*, VOL 73, NO.11, NOVIEMBRE 1982, 1551-1558.
7. Adrian Segall, "Bit Allocation and Encoding for Vector Sources", *IEEE Transactions on Information Theory*, VOL IT 22,NO.2, MARZO 1976, 162-169
8. A.H.Gray,Jr, J.D.Markel, " Quantization and Bit Allocation in Speech Processing ", *IEEE Transactions on Acoustics, Speech and Signal Processing*, VOL.ASSP 24,NO.6, DICIEMBRE 1976, 459-473.
9. T.Moriya and M.Honda, " Transform Coding of Speech Using a Weighted Vector Quantizer ", *IEEE Journal on Selected Areas in*

Communications, VOL.6, NO.2., FEBRERO 1988, 425-431.

10. Y.Shoham and A.Gersho, " Efficient Bit Allocation for an Arbitrary Set of Quantizers ", *IEEE Transactions on Acoustics, Speech and Signal Processing*, VOL. 36, NO.9, SEPTIEMBRE 1988, 1445-1453.
11. V. Cuperman, " On Adaptive Vector Transform Quantization for Speech Coding ", *IEEE Transactions on Communications*, VOL.37, NO.3, MARZO 1989, 261-267.
12. P.Valle. Tesis de Licenciatura. Diseño de un Sistema de Compresión de Señales de Voz para Velocidades de Transmisión menores a 9600 bits por segundo. 1988.
13. A.Gersho, " Asymptotically Optimal Block Quantization ", *IEEE Transactions on Information Theory*, VOL IT-25, NO.4, JULIO 1979, 373-380.
14. Texas Instrument, *Digital Signal Processing Applications with the TMS320 Family*, Volumen 1. Englewood Cliffs, NJ : Prentice Hall 1987.
15. L.R.Rabiner and B.Gold, *Theory and Applications of Digital Signal Processing* . Englewood Cliffs, NJ : Prentice Hall, 1975.
16. L.R.Rabiner and R.W.Schafer, *Digital Processing of Speech Signals*. Englewood Cliffs, NJ : Prentice Hall 1978.
17. P.E.Papamichalis, *Practical Approaches to Speech Coding*. Englewood Cliffs, NJ : Prentice Hall, 1987.