VNIVERSIDAD NACIONAL AVÏNºMA DE MEXICO

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

TÉCNICA DE COMPRESIÓN EN IMÁGENES GEOGRÁFICAS BASADA EN LA TRANSFORMADA WAVELET PARA COMUNICACIÓN VÍA TCP/IP

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN

P R E S E N T A:
RAFAEL MARCOS MONTANTE LÓPEZ

Director de Tesis :
M.C. Marco Antonio Vigueras Villaseñor

Ciudad Universitaria, 2001

Dedicatorias y Agradecimientos

Quiero dedicar este trabajo a:

A mi Madre, por haberse preocupado tanto en que no me faltara amor, cariño, comprensión, principios, educación y bienestar.

A mi Padre, por ser mi mejor amigo, por apoyarme, estar presente y compartir conmigo cada logro conseguido.

A mi tía Nena, por hacerme sentir afortunado de no carecer de la persona más importante en mi vida y por todo el cariño y apoyo que me has dado.

A Gloria, por cada esfuerzo y empeño puesto en mi educación, por ser un ejemplo e impulsar cada deseo de superación.

A Jaime y a mis hermanos: Any, Jaime, Susana y Manuel, con mucho cariño.

A la familia Montante, a la cual estoy muy orgulloso de pertenecer y que sin su cariño, enseñanzas y todo el apoyo, hubiera sido muy díficil conseguir todo lo que con orgullo he obtenido.

Agradezco en primer instancia a Dios por permitirme vivir intensamente cada momento.

Agradezco enormemente al M.C. Marco Antonio Vigueras V. por haberme tenido confianza y paciencia al dirijir esta tesis y a mis sinodales por aceptar ser parte del jurado.

De una forma muy especial, quiero agradecer al Biol. José Luis Villarreal, por haber planteado este proyecto y por toda la ayuda y apoyo incondicional que me ha brindado.

A mi Alma Mater y a la Facultad de Ingeniería por dejarme pertenecer a la mejor Universidad del País.

Quiero también agradecer a todo el personal del Dpto. de Supercómputo de la D.G.S.C.A. por haberme permitido pertenecer a una de las generaciones del plan de becarios.

Agradezco la valiosa aportación de Roque Alarcón Guerrero en la corrección del estilo de este trabajo.

A mis amigos, agradezco todas las palabras de aliento, la compañía y por hacer que prevalezca una de las cosas más importantes de esta vida.

A mis compañeros del Departamento de Visualización, por todos sus comentarios, consejos y sugerencias, gracias.

"El día más díficil fue ayer"
Anónimo.

Prefacio

La necesidad de métodos alternativos para agilizar el transporte de datos en las redes de computadoras, hace que año con año surjan nuevas propuestas o métodos alternativos, con el único propósito de solventar ciertos factores que limitan el transporte de los datos. En muchas ocasiones estos factores pueden ser las mismas condiciones físicas de los dispositivos o la forma en la que se representan los datos en una computadora. En particular, se eligió atacar el problema de representación de los datos de acuerdo a las características y objetivos de este proyecto. En la actualidad, existe un conjunto amplio de propuestas para solventar ese problema, en donde unas presentan ventajas y desventajas con respecto a otras, pero lo importante es que cada una realiza una propuesta diferente y hace que se adapte mejor a ciertos problemas y requerimientos que otros, que sin lugar a dudas, contribuye a mejorar o aportar ideas para futuras propuestas. El presente trabajo de investigación fue elaborado para probar una de esas tantas propuestas, con el propósito de estudiar la manera en la que se adecua a cierto tipo de imágenes.

Este proyecto trata de agilizar el transporte principalmente de imágenes en una red de computadoras, mediante el uso de una *técnica de compresión.* El principal logro de esta técnica elegida es que fue una de las primeras en considerar el uso de una transformada, que en esos momentos estaba tomando gran auge, y que fue y sigue siendo estudiada por muchos investigadores relacionados con el campo del análisis de señales. Esta transformada es ejecutada con base en un tipo de señal llamada *wavelet.*

Muchos investigadores se dieron cuenta de que una de las etapas típicas de un proceso de compresión podía ser realizado con esta transformada, de la cual observaron muchas ventajas en comparación con la utilizada en esos momentos por el estándar de compresión (JPEG). A partir de ese momento se empezaron a idear diversas formas con las cuales dicha transformada podía interactuar con las demas partes que forman una técnica de compresión, una de ellas era precisamente la *cuantificación vectorial.* A raíz de esto fue formalizado el uso de estas partes en dicho proceso y a mediados del año 1992, salió publicado un artículo que mencionaba cuales eran las bases teóricas que reafirmaban tales hechos (Marc Antonini, Michel Barlaud, Ingrid Daubechies y Pierre Mathieu, 1990).

La técnica que se propone en este proyecto se basó en ese artículo y se planteó

desde un inicio para ser aplicada a imágenes de tipo geográfico, con la finalidad de mostrar su desempeño en diversos aspectos como, la calidad visual de la imagen descomprimida, el tiempo del proceso de compresión y descompresión y su uso en una aplicación que involucrara el transporte de éstas en una red de computadoras.

La tesis está dividida en cinco capítulos, siendo el primero utilizado para mencionar aspectos introductorios, tanto de los conceptos que involucra una técnica de compresión, como el mostrar ventajas y desventajas de éstas, dando lugar también a mencionar la problemática principal del uso de este tipo de imágenes.

El capítulo dos describe conceptos importantes para conocer qué son las redes de computadoras y como es que viajan los datos a través de éstas. De la misma forma, se da una introducción de la teoría y conceptos que involucra la transformada wavelet.

El capítulo tres fue dedicado para establecer cómo se construye la técnica de compresión, se mencionan conceptos importantes que explican en qué se basa un método de compresión aplicado a imágenes, para que en la parte final de este capítulo se detallen de forma clara y concisa, las partes que forman esta técnica y la manera en la que interactúan.

El capítulo cuatro fue dedicado a presentar los resultados obtenidos de aplicar el proceso de compresión a las imágenes. Estos pueden ser observados de manera visual por medio de las imágenes finales y de manera cuantitativa, por medio de ciertas medidas de error utilizadas para medir el rendimiento ("*performance*") del método de compresión. También en este capítulo se presenta cómo el proceso de compresión afecta un análisis realizado con cierto tipo de imágenes, para lo cual es mostrado el resultado obtenido de un proceso conocido como *clasificación de pixeles*.

En el capítulo cinco se muestra el desarrollo de una aplicación práctica que fue sugerida para trabajar principalmente con imágenes y con la técnica de compresión en un sistema cliente/servidor sencillo. Se muestra a grandes razgos cómo fue desarrollada por medio del lenguaje Java.

Para finalizar este trabajo, en el apéndice A se muestra una pequeña descripción de cómo puede ser utilizada la aplicación.

A este trabajo se le ha añadido un disco flexible con los programas fuentes que se desarrollaron en este proyecto y algunas imágenes de evaluación.

Índice General

Índice de Figuras

Índice de Cuadros

Capítulo 1

Introducción

Actualmente la compresión de imágenes forma parte importante en el área de investigación del procesamiento digital de imágenes, ya que la cantidad de datos para crear una imagen que se obtiene desde un dispositivo que convierte una señal analógica en digital es muy grande, así como el número de imágenes [6]. También es fundamental en el área de transmisión y recepción de datos a través de una red de computadoras, por ejemplo Internet, o una red de telecomunicaciones, como la transmisión de señal de televisión digital (HDTV).

Para tener una idea general de la importancia de la compresión de imágenes en las áreas antes mencionadas se presentan a continuación algunos ejemplos.

En el caso de Internet, es fácil percibir cuando una página contiene algunas o muchas imágenes, ya que regularmente la página tarda algunos momentos en desplegar su contenido total. Aunque en muchas ocasiones el retraso al desplegarla, pueda ser a causa de las imágenes, también dependerá mucho de varios factores que componen la red, como su velocidad y ancho de banda.

En un sistema de recepción y transmisión de datos vía satélite, en [14] se menciona que, "la información se transmite codificada, modulada y en muchos casos comprimida, principalmente cuando se transmite vídeo (secuencia de imágenes), ya que esto aumenta la tasa de transmisión de los datos". Por ejemplo, si un cuadro ("frame") de una señal de vídeo se digitaliza, es decir, se crea una imagen digital, se observará que el tamaño en bits de la imagen resultante será igual a la cantidad de pixeles (8 bits/pixel) que se tengan horizontalmente por la cantidad de pixeles que se tengan verticalmente, por el número de bits que se ocupen para representar los colores de la imagen; por ejemplo si la imagen es de 512×512 pixeles y 256 colores (8 bits), el tamaño que ocupa la imagen en algún dispositivo de almacenamiento es de 2,097,152 bits, que en bytes son 262,144 (1 byte = 8 bits). Hay que notar que esto es sólo por un cuadro de la señal de video, si entonces se quisiera almacenar un segundo de video (30 cuadros), el tamaño que ocuparía en bytes sería 7,864,320

$(262,144 \times 30 = 7.5$ MB) con lo que podemos notar que el aumento en la cantidad en bytes es ya considerable sólo para un segundo de video. La importancia que tiene en este caso la compresión de imágenes es la de reducir la cantidad de datos para representar las imágenes, con el propósito de que los datos a transmitir sean el menor número posible, con la garantía de que cuando se quiera reproducir el video, éste se pueda observar tal y como era originalmente.

El término compresión de datos es parecido en concepto a una codificación de datos[1], donde la codificación en términos generales es acompañada de alguna representación especial de los datos, los cuales satisfacen una necesidad. El ejemplo más claro de una codificación es el mundialmente conocido "Código Morse", en donde para representar las letras del alfabeto se usan puntos y líneas, es decir, cada letra del alfabeto tiene asociada alguna representación por medio de puntos y líneas, por ejemplo para la letra A es ". - ", para la letra B "-...". La garantía de la codificación es que en la decodificación (proceso inverso a la codificación) no existe error o cambio alguno en el dato que se codificó. Si por ejemplo, se enviara por algún medio una señal de S.O.S., lo lógico es que la parte que la decodifique, entienda que es un S.O.S. En cambio en la compresión de datos puede o no haber errores en los datos finales, por lo que aquí es aceptable cierta pérdida de información.

La compresión de datos tiene una aplicación importante en las áreas de transmisión y almacenamiento de datos y actualmente es crucial para el crecimiento de la informática multimedia. Muchas de las aplicaciones para su procesamiento requieren almacenamiento de grandes volúmenes de datos y el número de tales aplicaciones se incrementa constantemente con el uso de las computadoras [10]. Al mismo tiempo, la proliferación de las comunicaciones en las redes de computadoras, hace que la transferencia masiva de datos sobre canales de comunicaciones sea más demandante. La compresión de datos para ser almacenada o transmitida reduce costos de almacenamiento o comunicación. Cuando la cantidad de datos a ser transmitida se reduce, se incrementa la capacidad de los canales de comunicaciones, por ejemplo, comprimir un archivo a la mitad de su tamaño original, es equivalente a duplicar la capacidad del medio de almacenamiento.

En el caso de las imágenes, siempre que se digitaliza una señal para formar una imagen digital, se produce una gran cantidad de datos. Esta cantidad de datos generados puede ser tan grande que su almacenamiento, procesamiento y comunicación puede llegar a causar problemas en aplicaciones prácticas, en estos casos es necesario utilizar ciertas formas de representación que permitan codificar la misma cantidad de información sin tener que utilizar el total de los datos generados [10]. Por lo que es deseable poder representar la información en una imagen con la menor cantidad de bits posibles y al mismo tiempo poder reconstruirla lo más cercana o semejante a la

[1] En casi todas las referencias citadas, codificación de datos y compresión de datos, resulta ser lo mismo, pero no lo es, al ser traducido al castellano y en concepto.

imagen original [6]. La compresión de imágenes afronta el problema de la reducción en la cantidad de datos necesarios para representar una imagen digital.

La demanda del manejo de imágenes en forma digital se ha incrementado en años recientes. Gracias al mejoramiento y reducción significante en el costo de exploradores (scanners), las fotografías, texto impreso y otras aplicaciones de multimedia pueden ahora ser convertidas fácilmente a información digital (digitalización de imágenes). La adquisición directa de imágenes digitales (digitalización de escenas), es realizada comúnmente por un sensor o algún tipo de electrónica asociada. Ésta se ha mejorado gracias al advenimiento de la electrónica en cámaras fijas, las cuales se pueden encontrar actualmente en el mercado. Muchas modalidades de imágenes en medicina, como una tomografía computarizada o imágenes de resonancia magnética, generan imágenes directamente en forma digital. El uso de gráficas por computadora en publicidad y entretenimiento es demasiado amplio y su uso en visualización científica y aplicaciones en ingeniería está creciendo rápidamente. La razón para el interés de imágenes digitales es clara: la representación de imágenes en forma digital permite información visual para ser manipulada fácilmente de diversas formas útiles y originales. Este factor, combinado con el crecimiento exponencial en poder de cómputo sobre la década pasada, ha resultado en el uso de sistemas de imágenes digitales en diversos campos como visualización científica, astronomía, sensado remoto, medicina, fotografía, etc. [25].

Desgraciadamente el costo que se tiene que pagar al utilizar imágenes digitales es su almacenamiento y contar con algún medio (canal) capaz de transportar grandes volúmenes de datos. En el cuadro 1.1 se muestran algunos ejemplos de la cantidad de bytes que son necesarios para representar imágenes en algunas áreas que se mencionaron anteriormente [15]:

Volúmenes de datos en imágenes (en megabytes MB)	MB
Imágenes de satélite (LANDSAT)	1.7×10^9
1 hora de televisión a color	28×10^3
Enciclopedia británica	1.5×10^3
200 páginas de caracteres de texto	1.3
Una página vista como imágen	0.13

Cuadro 1.1: Almacenamiento de imágenes

En el tema siguiente se mencionará, a grandes rasgos, que es la compresión de datos.

1.1 Compresión de datos e imágenes

La compresión de datos puede ser vista como una rama de la teoría de la información en la cual el objetivo primario es minimizar la cantidad de datos para ser almacenados o transmitidos en un sistema.

El término compresión de datos se refiere al proceso de reducción del volumen de datos necesarios para representar una determinada cantidad de información. Debe hacerse una clara distinción entre lo que son los datos e información. No son sinónimos. De hecho, "los datos son los medios a través de los cuales se conduce la información" [10]. Con esto es posible utilizar distintas cantidades de datos para describir la misma cantidad de información.

Como ejemplo, podemos mencionar algunos tipos de codificación como el ASCII y el EBCDIC: Si en ASCII se codifica, una letra "A", por ejemplo, puede darse el caso que ésta fuera igual a 100 0001; en cambio si se codifica de igual forma la letra "A" en EBCDIC, puede ser que ésta sea igual a 1100 0001. En el ejemplo anterior se puede notar que el código ASCII utiliza 7 bits para codificar cualquier caracter y que el código EBCDIC utiliza 8 bits, con esto se concluye que se puede representar la misma información (es decir, la letra A), con más o menos bits, dependiendo del código que se esté utilizando.

Para el caso de las imágenes, existe un problema potencial, y éste es el gran número de bits requeridos para representarlas. Afortunadamente las imágenes digitales contienen una cantidad significativa de redundancia en su información. La compresión de imágenes se define en [25] como "el arte/ciencia de codificar eficientemente los datos de la imagen, intentando tomar ventaja de la redundancia, para disminuir el número de bits requeridos en la representación de una imagen". Para obtener este propósito muchas técnicas de compresión aplican una transformación lineal reversible a una imagen[2], para de esa forma obtener una imagen en la cual los valores de los pixeles no esten correlacionados, de manera que puedan ser representados por un número menor de bits, en comparación con la imagen original [6]. Esto puede resultar en una mejora significativa para el almacenamiento o la transmisión de la imagen en un sistema de comunicaciones.

1.2 Formación de la imagen digital

Todos tenemos la idea de lo que es una imagen, ésta puede ser una escena en tres dimensiones (3-D) o puede ser una fotografía; de hecho toda representación que podamos crear en la mente será una imagen [27].

[2] En el contexto de la compresión de imágenes, una imagen se dice que es un arreglo correlacionado de valores (pixeles).

Técnicamente una imagen de dos dimensiones (2D) es obtenida por un sistema óptico a través de rayos de luz que son reflejados en cada punto de la escena, captados por lentes. Esta imagen puede ser convertida en señal eléctrica o puede ser grabada como una fotografía [27].

Matemáticamente, una imagen es definida por una función $f(x, y)$ de dos variables (que son las coordenadas en el plano de la imagen que describen las direcciones espaciales), los valores de la función corresponden al brillo o k-tuplas de los valores del brillo en varias bandas espectrales (para color) o en el caso de blanco y negro, los valores corresponden a niveles de gris [27].

En [25], se menciona que: "para representar matemáticamente una imagen digital, podemos aproximarla mediante una función $f(x, y)$, en donde a la digitalización de cada punto (x, y) se le conoce como *muestreo de la imagen*, y a la digitalización de cada valor de un punto (x, y) en la función $f(x, y)$ (amplitud), se le conoce como *cuantificación del nivel de gris* (en el caso de imágenes en escalas de grises)".

Entonces una función $f(x, y)$ es descrita por una serie de muestras igualmente espaciadas en forma de una matriz de $N \times M$, como se muestra en la ecuación (1.1).

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,M-1) \\ f(1,0) & f(1,1) & \cdots & f(1,M-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{bmatrix} \tag{1.1}$$

El término de la derecha de la ecuación (1.1) representa lo que comúnmente se denomina *imagen digital*, en donde cada elemento de la matriz corresponde a un pixel de la imagen.

Siempre que se realice el proceso de digitalización se tienen que fijar valores para N y M y el número de grises permitidos para cada pixel; es común que estos valores sean potencias enteras de 2, es decir:

$$N = 2^n, M = 2^k \tag{1.2}$$

$$G = 2^m \tag{1.3}$$

en este caso $n = log_2 N$, $k = log_2 M$, si nos referimos a imágenes cuadradas, entonces $n = k$. G indica el número de niveles de gris. Aquí m indica el número máximo de bits para representar los niveles de gris. Empleando las ecuaciones (1.2) y (1.3) se obtiene el número b de bits necesarios para almacenar una imagen digitalizada:

$$b = N \times M \times m \tag{1.4}$$

Por ejemplo, una imagen de 128×128 pixeles con 64 niveles de gris ($m = 6$) necesita 98,304 bits de memoria. En las imágenes existen dos factores principales que hay que tomar en cuenta al momento de su creación, el primero lo forman los valores de N y M, que representan la resolución de la imagen y el segundo es el valor de m, el cual permite ajustar los niveles de grises. La elección entre los valores de uno y otro dependerá de que tan buena aproximación se quiere tener a una imagen continua. Es verdad que cuanto más se incrementan estos valores se aproxima más la imagen digitalizada a la imagen original, pero hay que tomar en cuenta que, como indica la ecuación (1.4), cuanto más crezcan estos valores, desafortunadamente también crece el número de bits para almacenarla.

El proceso de compresión de imágenes involucra cierta pérdida de información de la imagen original, más adelante se explicará en más detalle como es que sucede esto, en este momento las figuras que se muestran a continuación ejemplificarán como afectan a la calidad de la imagen la variación de los parámetros N, M y m (durante el capítulo se tomará $M = N$).

La imagen de la figura 1.1(a) muestra una imagen digital con 256 niveles de gris y 256×256 pixeles. Las imágenes de la figura 1.1(b) a 1.1(d) muestran los resultados de reducir la resolución espacial de $N = 256$ a $N = 128$, 64 y 32 pixeles respectivamente. Es lógico que, cuando la resolución espacial disminuye, la imagen resultante tendría que obervarse más chica que la original, pero para mostrar este ejemplo, se le aplicó un operador de acercamiento ("zoom") a las imágenes reducidas para poder ser apreciadas del mismo tamaño que la original, de esta forma se puede notar el efecto que se produce al tener imágenes de resolución espacial pequeña. Este efecto es conocido como "replicación de bits" [10], el cual se aprecia al observar las últimas imágenes, en donde se nota un patrón de tipo "mosaico" o "mosaiqueo". De esta forma se puede afirmar que la disminución de la resolución espacial no es un método para disminuir el número de datos de la imagen, por lo cual no se logra compresión.

Las imágenes de la figura 1.2, muestran el efecto producido al reducir el número de bits empleados para representar el número de niveles de gris de una imagen. La imagen de la figura 1.2(a) es de 256×256 pixeles por 8 bits (256 niveles de gris). Las imágenes de las figuras 1.2(b) a la 1.2(d) han sido obtenidas al reducir el número de bits a m=6, m=4 y $m = 1$, respectivamente, manteniendo constante la resolución espacial. Se puede notar que las imágenes con 256 y 64 niveles de gris son visiblemente idénticas. Sin embargo, al reducir los niveles de gris de la imagen a 16, figura 1.2(c), se nota que la calidad de la imagen no es la misma, ya que los detalles de sombreado, o cambios de luminosidad en la imagen, se van perdiendo debido a la carencia de niveles de gris. La reducción de bits usados para tener los niveles de gris, si bien ayuda a reducir el espacio de almacenamiento de la imagen (de acuerdo a la ecuación 1.4), afecta el nivel de detalle de la imagen. Por tanto, la reducción del número de bits para representar la tonalidad de un pixel, tampoco es un método

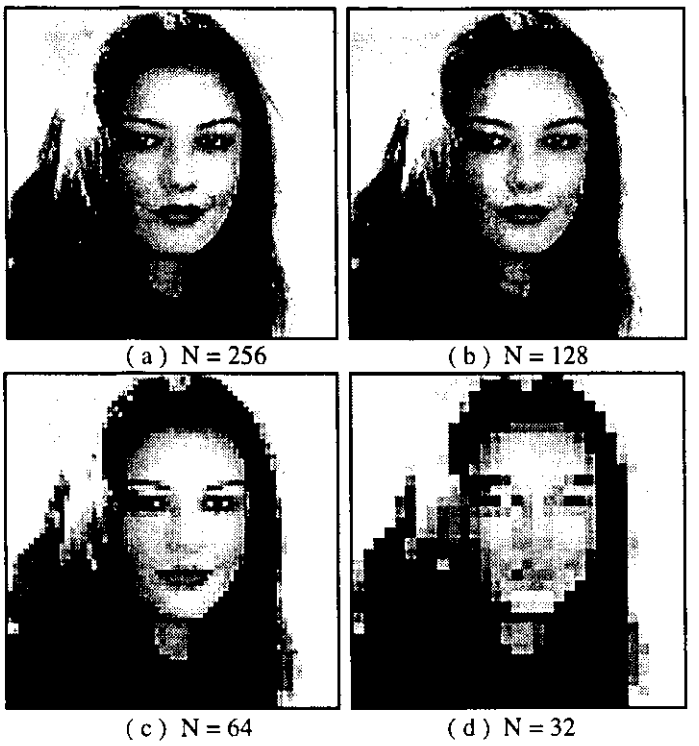

Figura 1.1: Reducción de la resolución espacial de una imagen.

para lograr compresión en la imagen.

Las imágenes anteriores fueron el resultado de variar los parámetros N y m, aún así no se puede determinar cuales son los valores ideales de N y m para tener una imagen de buena calidad sin que su almacenamiento sea excesivo, estos factores dependerán de la aplicación práctica y de la calidad subjetiva que se quiera obtener.

1.3 Clasificación de las técnicas de compresión

Los archivos de imágenes comúnmente contienen un porcentaje alto de información que es redundante y mucha que es irrelevante. Estos factores, redundancia e irrelevancia, son los que se aprovechan para las diversas técnicas de compresión. La redundancia la podemos notar en ciertas regiones de una imagen que son del mismo tono. Por esto, sería innecesario codificar cada uno de estos pixeles por separado, ya que contienen el mismo valor y de alguna forma se puede hacer que se codifiquen en conjunto. En el caso de la irrelevancia se verá posteriormente que hay ciertos componentes en una imagen que pueden ser eliminados sin que la calidad visible de la imagen se afecte. El objetivo de las técnicas de compresión es poder reconstruir con exactitud (caso ideal) o aproximadamente una imagen compactada [5].

Figura 1.2: Reducción de los niveles de gris de una imagen.

Existen tres tipos de redundancia en imágenes digitales, las cuales son:

- redundancia espacial, la cual es debida a la correlación (o dependencia) entre valores de pixeles adjuntos, la cual es explotada por varias técnicas de compresión, como JPEG,
- redundancia espectral, la cual se debe a la correlación entre diferentes planos de color (por ejemplo, en una imagen de color RGB) o banda espectral (por ejemplo, fotografías aéreas tomadas con sensores remotos), en muchas ocasiones los valores RGB de una imagen a color son transformados a otras componentes (YUV), en donde el proceso de codificación resulta ser más eficiente,
- redundancia temporal, la cual es debida a la correlación entre diferentes cuadros (frames) en una secuencia de imágenes, este tipo de redundacia es tomada comúnmente en la codificación de video [25].

La investigación en compresión de imágenes está dirigida a la reducción del número de bits requeridos para representar una imagen quitando estas redundancias. Para lograr esto es importante tener conocimientos de los conceptos de la teoría de la información y de la ejecución de cualquier esquema de compresión. Los algoritmos

o esquemas de compresión de imágenes pueden estar divididos en dos grupos fundamentales: algoritmos sin pérdida (*lossless*) y algoritmos con pérdida[3] (*lossy*) [25].

En la compresión sin pérdida (también conocida como compresión reversible o de preservación de bits), la imagen reconstruida después de la compresión es numéricamente idéntica a la imagen original pixel por pixel. Obviamente la compresión sin pérdida es idealmente deseable, ya que la información no se compromete, ni se pierde. Sin embargo sólo es posible una modesta cantidad de compresión [25].

En la compresión con pérdida (también conocida como compresión irreversible), la reconstrucción de la imagen no es numéricamente idéntica a la original, sino que la imagen resultante va a contener degradaciones con respecto a la original. Como resultado, es posible lograr mucha compresión en comparación a la compresión sin pérdida. Es importante notar que estas degradaciones pueden o no, ser visibles aparentemente, característica que es importante recordar. De hecho, el término "visualmente sin pérdida", es aveces usado para caracterizar esquemas de compresión con pérdidas que resultan no visibles bajo condiciones visuales normales [25].

1.4 Necesidades y ventajas de la compresión

La necesidad y el interés por comprimir los datos de una imagen se remonta a hace más de veinticinco años, en donde los investigadores se preocupaban ya por reducir los anchos de banda en los canales de transmisión de video. La compresión en aquel entonces se realizaba de forma analógica, pero con el surgimiento de las computadoras y circuitos digitales esas técnicas pasaron de ser analógicas a digitales.

El ancho de banda de un canal de transmisión se refiere a la cantidad de datos que puede transportar el medio o canal de comunicación. Esta cantidad de datos muchas veces es limitada por el material con el que está construido dicho medio.

La compresión de imágenes forma una parte importante en la producción de video digital, ya que las técnicas empleadas en compresión de video, utilizan las mismas que se usan en imágenes fijas, considerando además otros aspectos, como la redundancia temporal.

Cuando la transmisión o almacenamiento de video digital es vista en términos de tasas de bits y requerimientos de anchos de banda analógicos, los números son tan grandes que muchas tareas serían imprácticas. Es por eso que se tiene que considerar la compresión de video, logrando con ello factores de compresión por arriba de 100:1 en algunos casos [14].

[3] También conocidos como: visualmente sin pérdidas (*visually lossless*)

Una de las ventajas de la compresión de datos en el campo de la multimedia es que ha contribuido al crecimiento del mismo, pudiendo así tener herramientas para manipulación de imágenes, video y audio. Además la compresión de imágenes es un punto fundamental en muchas aplicaciones, tales como la videoconferencia, la transmisión de televisión digitaly de fax. También juega un papel muy importante en la tecnología de los sensores de imágenes en los satélites espaciales, en la creación de imágenes médicas e inclusive en imágenes de huellas digitales utilizadas por el FBI.

Es importante mencionar que, en cualquier aplicación que involucre compresión de imágenes, debe tomarse en cuenta que la imagen no se reconstruirá de manera exacta, es decir, después de aplicar un proceso de compresión a una imagen, se genera un conjunto de datos que solo puede ser interpretado por la técnica de compresión, así, para poder observar la imagen, es necesario aplicar un proceso de descompresión, en el cual a partir del conjunto de datos obtenido en la etapa anterior, se pueda reconstruir la imagen. Este es un aspecto clave, por ejemplo en la visualización científica, ya que en el área de análisis de datos, éstos están relacionados comúnmente con valores de los pixeles de una imagen, ya sea por haber sido aplicado un mapeo de colores y generar una imagen o tomar un conjunto de datos como tales y generar con éstos una imagen; por lo que es importante establecer un compromiso entre la reconstrucción de una imagen comprimida y la obtención de los datos a partir de esa imagen, ya que como se mencionó anteriormente, éstos seguramente no serán los originales. En el siguiente tema se explica un poco más acerca de la compresión de imágenes en la visualización científica.

1.5 La compresión de imágenes en la Visualización

Gracias al avance en la tecnología de los microprocesadores, las tarjetas gráficas y de las técnicas de programación orientada a objetos, la computación ha formado parte importante en el campo de la visualización. Actualmente la visualización se puede dividir en dos áreas: la visualización comercial y la visualización científica. La visualización comercial se puede observar comúnmente en anuncios de televisión y ha tenido mucho éxito en el mundo del cine; este tipo de visualización usa gráficas sólo como entretenimiento. La visualización científica está enfocada al campo de la simulación numérica; con el empleo de técnicas de visualización, los investigadores pueden observar el resultado de simulaciones numéricas usando representaciones de gráficas complejas. Con la visualización es posible ver lo que comúnmente es imposible, por ejemplo, la temperatura en un sólido, las fuerzas de torsión en un cuerpo, moléculas, plasmas. La visualización es de gran ayuda a físicos, químicos, ingenieros, médicos, porque gracias a ésta, es posible observar con gráficas y animación los resultados de sus modelos.

De manera más formal, la visualización científica se define como la técnica que explora datos numéricos por un significado visual en objetos gráficos. Con métodos

iterativos, por ejemplo, ver objetos en tres dimensiones (3D) le permite al investigador observar todos los aspectos relevantes de un conjunto de datos [21].

El conjunto de datos que se obtienen a través de dispositivos de medición o por resultados de simulaciones han causado un serio problema para el proceso de visualización, ya que muchas veces este conjunto de datos es demasiado extenso y no se tiene la capacidad para almacenarlos y procesarlos. Como ejemplo se pueden mencionar las enormes cantidades de datos que un satélite espacial manda cada año a una estación terrestre. Actualmente no se tiene completamente una tecnología para recibir, analizar y presentar la información de tal forma que sea usada por los científicos. Es por eso que la compresión de datos ha venido a ser un elemento importante en los sistemas de visualización.

Un factor importante que se debe tomar en cuenta en los sistemas que involucran a la visualización científica es reducir el espacio de almacenamiento y los tiempos de acceso a los conjuntos de datos, con el propósito de acelarar la visualización de los mismos.

Los científicos en diferentes disciplinas tratan de obtener mejores resultados en sus estudios y para esto es necesario tener la mayor cantidad de datos posible de algún fenómeno; a algunos quizá sólo les interese estudiar pequeños subconjuntos de datos o solo una región de interés en la máxima resolución que se pueda obtener, es decir, que los datos se aproximen lo mayor posible a los datos obtenidos en mediciones o simulaciones. Para ellos sería de gran importancia poder examinar el conjunto completo de datos en una baja resolución y ser capaces después de identificar y analizar sólo un subconjunto de los datos con más detalle. Una técnica que permite manipular la información de esta forma es conocida como *Transmisión Progresiva*, la cual puede usar un modelo de codificación jerárquica.

En la siguiente sección se explica de manera breve, cual es el problema a enfrentar y sobre que imágenes es aplicada la técnica de compresión de este proyecto de investigación.

1.6 Problemática en imágenes geográficas

Este proyecto de investigación tiene la finalidad de trabajar principalmente con dos tipos de imágenes geográficas. Por un lado la técnica será aplicada a imágenes generadas a partir de modelos digitales de terreno[4], también conocidos como planos digitales de elevación (DEM), de la República Mexicana. Como caso particular se tienen datos provenientes de fuentes como el INEGI (Instituto Nacional de Estadística Geografía e Informática), los cuales representan el modelo digital de terreno de la República Mexicana; estos datos se agrupan en matrices de números enteros que

[4]Del inglés DEM (Data Elevation Model) o Modelo Digital de Elevación

tienen una dimensión de 1201 × 1201. Para representar el modelo digital de terreno de toda la República Mexicana se necesitan 255 matrices de este tipo. Después de realizar determinado procesamiento a las matrices de datos se puede tener una imagen como la mostrada en la figura 1.3, la cual representa el modelo digital de terreno de la República Mexicana.

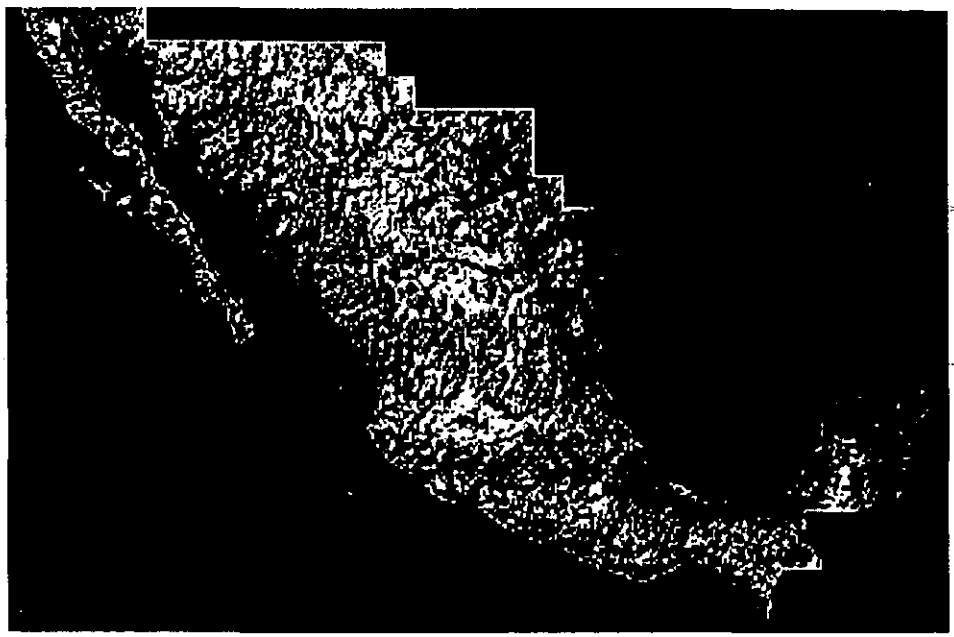

Figura 1.3: Representación del modelo digital de terreno de la República Mexicana.

El problema del almacenamiento de este conjunto de datos es un factor importante, ya que cada archivo ocupa 1201 × 1201 × 2 = 2.89 MB (la multiplicación por 2 en la igualdad anterior es porque los datos estan almacenados con un tipo de dato *short int* o entero corto, es decir, 2 bytes por cada dato) y que por los 255 archivos suma un total de 735.62 MB. Aquí cabe aclarar, que toda la zona en la que no hay datos (la que representa al mar en este caso), no se considera en la cantidad mencionada anteriormente, es decir, no se contabilizan las matrices, pero debe ser tomada en cuenta cuando se genera la imagen mostrada en la figura 1.3. Esta imagen es formada por un conjunto de matrices de las dimensiones mencionadas anteriormente y se necesitan 32 matrices a lo ancho y 19 a lo largo, lo cual suma un total de 608, de las cuales 255 son de datos del modelo y 353 son sólo para representar al mar. Entonces, la cantidad total de espacio de almacenamiento de la imagen completa es de 1.7 GB, la cual es ya considerable, tanto para el almacenamiento como para su transmisión.

Por otro lado existe también la problemática de manejar imágenes multiespectrales, estas imágenes son captadas por el satélite Landsat 4 y están formadas por

7 bandas. Cada una de las imágenes tiene una resolución de 4000 × 3600 pixeles a 8 bits/pixel. El tamaño de cada pixel representa un área de 25 × 25 metros. El espacio en almacenamiento requerido por estás imágenes es de 14.4 MB por banda y por las 7 es un total de 100.8 MB, lo cual resulta también una cantidad considerable para el manejo en cualquier aplicación práctica. Un ejemplo de una de las imágenes generadas por 3 de las bandas (para formar una imagen RGB) es mostrado en la figura 1.4. Esta imagen es formada por las bandas 7, 4 y 2, (7:4:2).

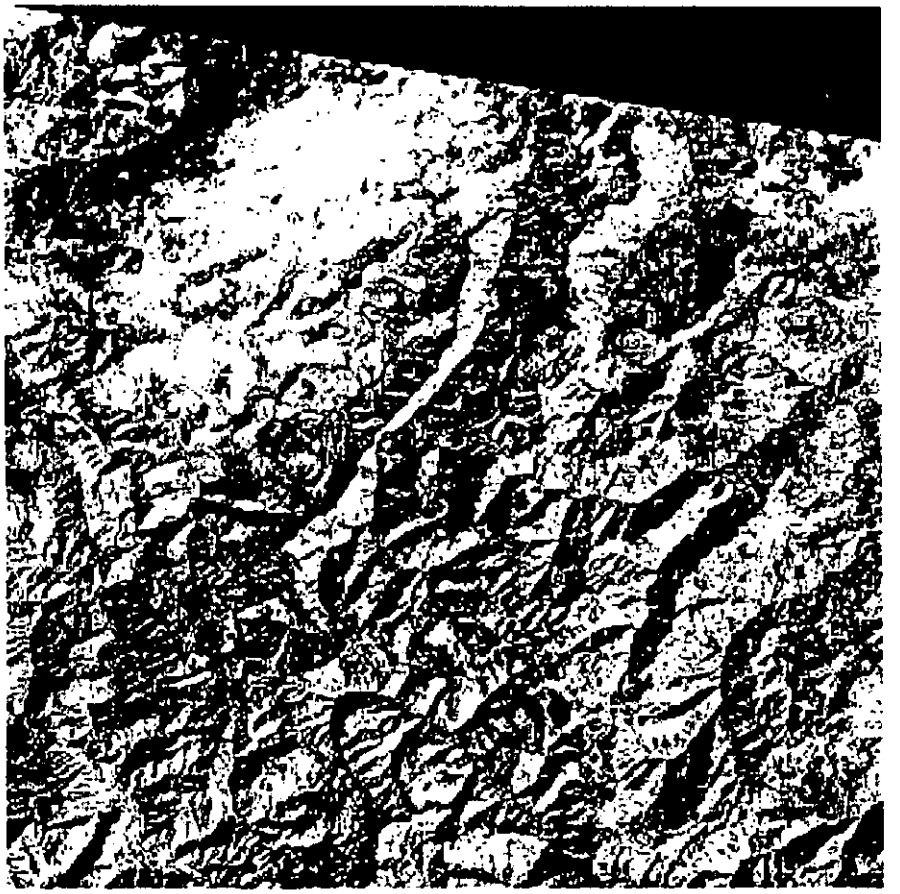

Figura 1.4: Imagen Multiespectral de la zona de Molango en Hidalgo, Mex.

Se ha mostrado hasta aquí una breve introducción de la problemática existente al utilizar imágenes en un sistema que depende de ellas. En este proyecto de investigación la problemática a resolver es básicamente el transporte de imágenes de gran tamaño en un sistema visual, el cual permita tener cierto "control" sobre la cantidad de pérdida de los datos originales, ayudando de igual forma a reducir el espacio de almacenamiento de la imagen.

Se pretende que la técnica de compresión sea utilizada en un sistema que permita visualizar los modelos digitales de terreno, en una geometría en 3 dimensiones (3D) y en su representación en 2 dimensiones (2D, imágenes). Este sistema forma parte de un proyecto que incluirá la visualización de datos en 2D y 3D. La parte de generación de datos 3D es realizada independientemente de la 2D y para lo cual hasta estos momentos se lleva gran avance en esa parte. Se requiere que este sistema sea interactivo, para lo cual el usuario podrá "navegar" dentro de la geometría en 3D o poder hacer acercamientos a una región de interés en la imagen ("zoom"). Cuando en una imagen (2D) sea seleccionada una región de interés, automáticamente se generará su geometría en 3D, en donde se podrá hacer un análisis de la zona a mayor detalle. El sistema plantea la posibilidad de tener un servidor de imágenes y de datos, por tanto se necesitará que dicho sistema deba estar corriendo en una máquina cliente, por lo cual es importante contar con un método de compresión que realice de manera más eficiente el transporte de los datos, ya que al hacer una petición al servidor, éste aplicará la técnica de compresión sobre los datos. Después de ser recibidos, el programa cliente será el encargado de realizar el proceso de descompresión, para que, de manera local éstos puedan ser trabajados y visualizados.

De acuerdo con los requerimientos del sistema anterior es necesario tener conocimientos básicos de la teoría de redes de computadoras en general, esto va a permitir comprender como se forma una red, su modelo básico de conexión, la estructura cliente/servidor, como se realiza el transporte de datos en una red y la importancia de tener un método para disminuir el flujo de datos en el transporte de los mismos, por lo que en el siguiente capítulo se describirá una breve introducción a estos conceptos.

Uno de los problemas del manejo de datos del modelo digital se puede solventar con la aplicación de la técnica en el sistema mencionado anteriormente. De la misma forma se pretende que la técnica sea utilizada también, para imágenes multiespectrales. Este tipo de imágenes son utilizadas principalmente para hacer el análisis de la zona que muestra la imagen, mediante un proceso llamado "clasificación de pixeles" y otro llamado "componentes principales". La clasificación de pixeles principalmente se refiere a separar ciertas regiones de la imagen como vegetación, bosque, desierto, agua, etc, para su análisis posterior. El estudio de los componentes principales se realiza para obtener datos estadísticos importantes de la imagen.

El punto importante en esta parte es que si al aplicar la técnica de compresión a este tipo de imágenes, se pueden obtener resultados similares o que no varíen mucho con los que se obtienen al hacer el análisis con las imágenes originales, entonces será viable realizar el proceso de compresión a éstas, para lo cual solo bastará determinar, cual será la tasa de compresión máxima que se puede permitir para que el análisis posterior no difiera en gran medida de los resultados obtenidos con los datos originales, ayudando de esta forma a reducir el espacio de almacenamiento que requieren

éstas.

En el siguiente capítulo se dará una breve introducción a la teoría que forma la base de este proyecto de investigación. Esta teoría es sobre la transformada wavelet. Se mencionarán los conceptos que la forman y como se aplica a imágenes, para después en el capítulo 3, describir como será aplicada esta teoría en la construcción de la técnica de compresión.

Capítulo 2

Fundamentos

2.1 Introducción

Detrás de cualquier técnica o algoritmo o simplemente algo que se desarrolle, hay una teoría que describe el comportamiento de las partes que lo componen. Este capítulo está enfocado a revisar de manera general los fundamentos teóricos en los que está basada la técnica de compresión y transmisión (a grandes rasgos), estos son, las redes de computadoras y la teoría sobre wavelets.

La técnica de compresión de imágenes propuesta tiene como enfoque principal trabajar en un ambiente cliente/servidor, por lo que es importante conocer las bases de cómo funciona este ambiente; así como tener un fundamento general de las redes de computadoras, y de igual forma tener claro como se transportan los datos a través de una red.

Particularmente, esta técnica utiliza un método de compresión basado en una transformación. La aplicación de esta transformación es uno de los puntos importantes en el método. Esta transformada ha sido estudiada desde principios de los años 80's y actualmente ha tomado mucho auge en el campo del procesamiento de señales e imágenes, particularmente en el área de compresión. Esta transformada es realizada por medio de funciones que son llamadas "wavelets", por lo cual es conocida como la transformada wavelet.

Este capítulo pretende mostrar de manera general qué es una wavelet, qué es una transformada wavelet, bancos de filtros y el concepto de multiresolución. La parte teórica de estos conceptos es demasiado extensa, por lo que resultaría muy difícil describirla en detalle en este trabajo de investigación; tampoco se pretende demostrar por qué es preferible el uso de éstas y no de otras transformadas, como la DCT (transformada coseno discreta) o algunas de uso frecuente en el procesamiento de imágenes, sino más bien, especificar cómo se emplea en un proceso de compresión de imágenes y mencionar cuáles son las ventajas que se obtienen al usarla.

2.2 Redes de computadoras

En la actualidad las redes de computadoras son el mecanismo más importante de comunicación a nivel mundial. Hoy en día, la necesidad de comunicación entre las personas hace que los investigadores y gente dedicada al área de las comunicaciones digitales, busquen nuevas formas o nuevos métodos para lograr de manera rápida y eficiente este propósito. Con el crecimiento de Internet, se ha logrado que éste sea el mecanismo más popular de comunicación a grandes distancias, por lo cual, muchos proyectos están orientados hacia esta forma de comunicación.

Este trabajo pretende evaluar el desempeño de un algoritmo que comprima datos (imágenes) y que estos sean transportados a través de una red. Para tener una idea general de cómo se transportan los datos en una red, los siguientes temas están enfocados en describir los conceptos básicos de una red y la forma en que se llevan a cabo las comunicaciones.

2.2.1 Topologías

Una red se define regularmente como un conjunto de computadoras y periféricos (impresoras, módems, escáners, etc.) que se interconectan por algún medio. La conexión puede ser directa (a través de un cable) o indirecta (a través de un módem, o por microondas). Los distintos dispositivos en la red se comunican entre sí, utilizando un conjunto predefinido de reglas, conocido como protocolos [24].

En una red los dispositivos pueden estar: en la misma habitación dispersos en un edificio, separados por muchos kilómetros utilizando líneas de teléfono para comunicarse o por microondas o algún dispositivo similar; incluso pueden estar separados intercontinentalmente, lo importante es que una red no necesariamente debe estar conectada físicamente en una área determinada. Una topología de red se define de acuerdo a como se distribuyen los dispositivos que se van a interconectar, básicamente puede ser en bus, anillo, estrella o alguna combinación de éstas. El diseño de la red es uno de los puntos importantes antes de conectar los dispositivos; de esto depende su buen desempeño de acuerdo a los requerimientos establecidos, además de que el elegir uno u otro dependerá incluso de la cantidad de dinero a ser invertido para construir la red. La topología de la red puede ser lógica o física, la lógica es la forma en la que opera la red y la física es la forma en que está tendido el cableado [7]. Las topologías lógicas más comúnes son: bus, anillo y estrella.

Después de tener una forma lógica de interconexión de los dispositivos que conforman una red, es necesario que se establezcan una serie de reglas para que éstos se puedan comunicar, es por eso que se tiene que definir un protocolo de comunicación entre los dispositivos.

2.2.2 Protocolos

Un protocolo es un conjunto de reglas que deben seguirse para que exista una comunicación entre dos dispositivos, las cuales definen la forma como ocurren las comunicaciones. Si por ejemplo, una computadora envía un mensaje a otra y ambas siguen correctamente el protocolo, el mensaje pasará independientemente del tipo de máquina y del sistema operativo que tengan [22].

Lógicamente, una red puede estar dividida en varias capas o bloques, estas capas serán construidas de manera que cada capa preceda a otra y el propósito es que cada capa proporcione un servicio a otras capas. Un protocolo provee un servicio de comunicaciones para el intercambio de mensajes [31].

En un modelo de red con capas, una capa n de una red se comunicará con la capa n de otra red por medio de un protocolo. La comunicación entre capas adyacentes se realiza por medio de una interfaz, que es la que define la clase de servicios que una capa ofrece a otra. Entonces, para que una capa n de una red pueda enviar un mensaje a otra capa n en otra red, tendrá que utilizarse un protocolo que defina las reglas de comunicación entre dichas capas. El mensaje tendrá que pasar por cada capa hasta llegar a una capa de nivel cero o de enlace físico por medio de la interfaz. Esta interfaz realizará operaciones para que el mensaje pueda ser transportado por cada capa, hasta llegar a la capa destino.

Los protocolos se desarrollan a partir de procesos muy sencillos con el objetivo de elaborar mecanismos complejos que cubran todos los problemas posibles y todas las condiciones de transferencia. Si, por ejemplo, se quisiera enviar un mensaje de una computadora a otra que está a unos cientos de kilómetros, la tarea podría ser muy compleja, considerando la forma como se transmite, si solo existiera un solo protocolo, la tarea sería demasiado grande y el protocolo tendría que ser muy especializado para poder cubrir todos los aspectos de la transferencia [22]. Debido a esto, se han desarrollado numerosos protocolos, en donde cada uno realiza una tarea específica.

Como se mencionará más adelante la mayoría de las capas en una arquitectura o modelo de comunicaciones tiene cada una su propio protocolo para comunicarse entre capas adyacentes, pero debido a la aplicación de este proyecto, el texto se enfocará más al protocolo TCP, que es uno de los protocolos utilizado comúnmente en las comunicaciones de datos.

2.2.3 Modelo OSI

La Organización Internacional de estándares, ISO (International Standar Organization), fue una de las primeras organizaciones en definir formalmente una manera común para la conexión de computadoras. Su modelo, Interconexión de sistemas abiertos, mejor conocido como OSI (por sus siglas en inglés, *Open Systems Inter-*

connection), define una estructura para la funcionalidad de una red en siete capas, donde uno o más protocolos implementan la funcionalidad asignada a cada capa [24]. La figura 2.1 muestra el esquema del modelo OSI:

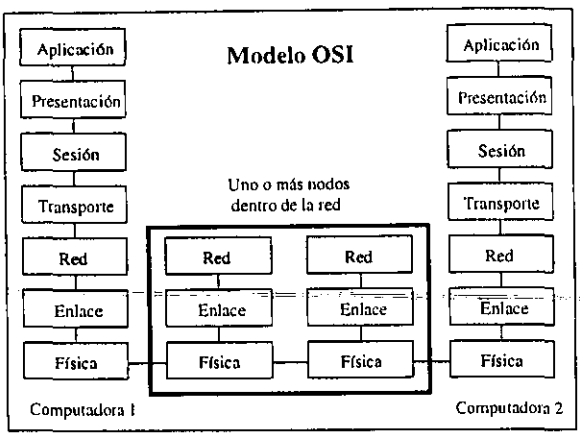

Figura 2.1: Modelo OSI.

Las capas de aplicación, presentación y sesión están orientadas a una aplicación final (por ejemplo, correo electrónico), en la que son responsables de la presentación de la interfaz del usuario. Las tres capas son independientes de las capas inferiores y son totalmente ajenas a los medios por los cuales los datos llegan a la aplicación [22].

Las cuatro capas inferiores tienen que ver con la transmisión de los datos y se ocupan del empaque, enrutamiento, verificación y transmisión de cada grupo de datos. Estas capas no se preocupan por los tipos de datos que reciben o envían a la aplicación, sino solamente se ocupan de la tarea de enviarlos [22].

En el cuadro 2.1 se muestra un resumen de las funciones de las capas que conforman el modelo OSI:

2.2.4 Modelo de Internet

El modelo de Internet es a veces llamado TCP/IP, debido a sus dos protocolos principales, TCP e IP. El modelo de Internet evolucionó de las experiencias de una de las redes tipo switcheo de paquetes o "packet-switched" más antiguas llamada ARPANET. Tanto la Internet como ARPANET fueron fundadas por la agencia de proyectos de investigaciones avanzadas (Advanced Research Projects Agency - ARPA), una de las agencias del departamento de defensa de los Estados Unidos y las dos fueron construídas un poco antes de que se propusiera el modelo OSI, ocasionando de esta manera, que se tuviera mucha influencia de estas arquitecturas de

Capa	Función Principal
Aplicación	En un ambiente cliente/servidor, en esta capa reside el programa cliente, proporcionando una interfaz de la aplicación con el usuario
Presentación	Convierte los datos a un formato común para que los pueda entender la capa de aplicación. Se encarga de la sintaxis y semántica de los datos que se transmiten
Sesión	Permite establecer sesiones entre el emisor y el receptor, sincronizando los datos transmitidos y permitiendo a cada una el conocimiento del estado de la comunicación entre éstos
Transporte	Comprueba que los datos enviados concuerden con los recibidos, determinando orden y prioridad, si existiera un error hace una petición de reenvío de datos
Red	Selecciona la ruta que deben tomar los mensajes entre el origen y el destino, utilizando tablas de ruteo estáticas o dinámicas
Enlace de datos	Proporciona una transmisión confiable entre dos puntos en una red. Requiere de un mecanismo que detecte y corrija errores
Física	Se ocupa de la transmisión de los datos a través de medio físicos, ya sean eléctrico u ópticos y se refiere basicamente al cableado u otro medio de transmisión

Cuadro 2.1: Modelo OSI y sus capas.

redes en la construcción del modelo de referencia [24].

En lugar de usar un modelo de siete capas como en el modelo OSI, el modelo de Internet está basado en un modelo de cuatro capas. En la figura 2.2 se muestra un esquema con protocolos del modelo de Internet:

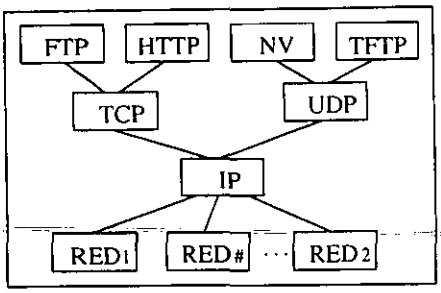

Figura 2.2: Esquema del modelo de Internet.

De acuerdo a la figura anterior, en el nivel más bajo hay una amplia variedad de protocolos de redes, denotados por RED1, RED2, y así sucesivamente. En la práctica estos protocolos son implementados por una combinación de hardware (por ejemplo, un adaptador de red) y software (quizá algún driver de un dispositivo de red). Por ejemplo, en esta capa se pueden encontrar la tecnología Ethernet o una interfaz de datos distribuida por fibra (FDDI). La segunda capa consiste de IP o protocolo de Internet (*Internet Protocol*), este es el protocolo que soporta la interconexión de múltiples tecnologías de redes en una sencilla interred lógica. La tercera capa consiste de dos protocolos principales, TCP, protocolo de control de transmisiones (*Transmission Control Protocol*) y UDP, protocolo datagrama de usuario (*User Datagram Protocol*). TCP y UDP proveen canales alternativos lógicos para programas de aplicación, TCP provee un canal fiable de flujos de bytes y UDP provee un canal de entrega de datagramas[1] inestables. En el lenguaje de Internet, TCP y UDP son en ocasiones llamados protocolos punto a punto, aunque también es correcto llamarles o referirse a ellos como protocolos de transporte [24]. La diferencia entre TCP y UDP es que TCP es orientado a conexión, es decir, TCP se ocupa de establecer una conexión virtual punto a punto con su destino con el propósito de controlar el flujo de los mensajes y retransmitirlos en caso de que sea necesario, en cambio UDP es un protocolo sin conexión, es decir, permite la transmisión de mensajes sin el establecimiento de una conexión, con lo cual no se garantiza que los mensajes lleguen correctamente [31].

Corriendo sobre la cuarta capa hay un rango de aplicación de protocolos, tales como FTP, protocolo de transferencia de archivos (*File Transport Protocol*), protocolo de transferencia de archivos trivial, TFTP (*Trivial File Transport Protocol*), Telnet (login remoto) y protocolo de transferencia de correos simple o correo electrónico

[1] Un datagrama es sinónimo de mensaje.

SMTP (*Simple Mail Transfer Protocol*), todos estos permiten la interoperabilidad con diversas aplicaciones. Para entender la diferencia entre la capa de aplicación y una aplicación, podemos pensar en la gran variedad de navegadores de Internet que existen, por ejemplo, Mosaic, Netscape, Explorer, etc. La razón por la cual uno puede usar cualquiera de estos programas de aplicación para poder acceder a un sitio en la Web es porque todos ellos utilizan el mismo protocolo en la capa de aplicación: el protocolo de transporte de hipertexto HTTP (*HyperText Transport Protocol*) [24].

Internet no es una sola red, sino un conjunto de éstas que se comunican entre sí a través de compuertas (a veces también conocidas como enrutadores). Una compuerta es un sistema que realiza funciones de relevo entre redes. Las distintas redes conectadas entre sí a través de compuertas a menudo se conocen como subredes, ya que éstas forman una pequeña parte de la red general más grande [24]. Las subredes son redes completas y están conectadas a una interred a través de una compuerta o en su caso a Internet. La figura 2.3 describe esta idea:

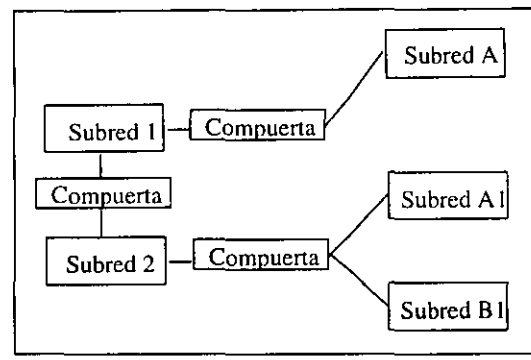

Figura 2.3: Composición de Internet a través de subredes.

Como se había mencionado anteriormente la mayor parte de las interredes, incluyendo a Internet, se pueden considerar como una arquitectura en capas. Estas capas son solo conceptuales y no son capas físicas ni de software como en el caso del modelo OSI. Internet está formado básicamente por cuatro capas, las cuales se muestran en la figura 2.4.

Estas capas no se deben confundir con la arquitectura de cada máquina, descrita en el modelo de siete capas de OSI, sino que se usan para ver como funciona la interred, la red local TCP/IP y cada una de las máquinas individuales. En el cuadro 2.2 se muestra un resumen de la función que desempeña cada capa del modelo de Internet [24].

Hasta esta parte, se ha mostrado como se establecen las comunicaciones en una red con base en protocolos. A continuación se describirá de forma breve qué quiere

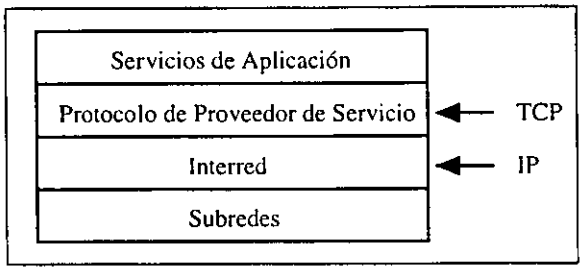

Figura 2.4: Arquitectura de Internet.

Capa	Función principal
Subred	En esta capa residen las máquinas independientes, las cuales están conectadas en una red de área local (LAN), la cual es identificada como una subred
Interred	Proporciona la funcionalidad para las comunicaciones de las subredes, a través de compuertas. En esta capa los datos son transferidosde una compuerta a otra, hasta que llegan a su destino. Aquí es donde corre el protocolo IP
Protocolo proveedor de servicios	En esta capa corre el Protocolo de Control de la Transmisión (TCP), el cual es el responsable de manejar el flujo o tráfico de datos y asegura la confiabilidad de la transferencia del mensaje
Servicios de aplicación	Esta capa soporta la interfaz de las aplicaciones de usuario, como el correo electrónico o la transferencia remota de archivos, entre otras.

Cuadro 2.2: Capas del modelo de Internet.

decir IP y TCP, ya que estos nombres están muy relacionados con las comunicaciones a través de Internet.

2.2.5 Interconexión simple (IP)

Aunque la palabra Internet es parte del nombre del protocolo, no se restringe su uso exclusivamente a éste, si bien en Internet todas las máquinas utilizan o entienden IP, éste se puede utilizar en redes dedicadas que no tengan ninguna relación con Internet. IP define un protocolo, no una conexión [22].

El Protocolo Internet (IP) es la herramienta principal usada actualmente para construir inter-conexión de redes escalables[2] y heterogéneas[3]. IP corre sobre todos los nodos en una colección de redes y define la infraestructura que permite que estos nodos y redes funcionen como una red lógica sencilla [24].

El modelo de servicio de IP tiene dos partes: un esquema de direccionamiento, el cual provee una manera de identificar todas las máquinas (hosts) en la interconexión y un modelo de datos de entrega de datagramas [22].

Las tareas del protocolo Internet son el direccionamiento de los datagramas de información entre computadoras y la administración del proceso de fragmentación de dichos datagramas. El protocolo tiene una definición formal del diseño de un datagrama de información y de la información de un encabezado compuesto de información relativa al datagrama. IP es responsable del enrutamiento de los datagramas, determinando a donde se enviarán, así como las rutas alternas en caso de problemas [22].

Cuando se envía un datagrama, no se garantiza que llegue a su destino, quizá por problemas en el ruteo o de alguna mutilación de un datagrama. IP no es el encargado de resolver estos problemas, no verifica que un mensaje enviado se reciba correctamente, este proceso es responsabilidad de otras capas [22].

Otra de las características de IP, es que a éste no le importa por qué nodos viaje el datagrama a lo largo de la ruta, ni siquiera en donde se originó el datagrama y cual es su destino, por eso es que se le conoce como "sin conexión". Esta información se encuentra en el encabezado, pero el proceso de analizar y pasar el datagrama no tiene nada que ver con el análisis por parte de IP del envío y recepción de las direcciones IP.

Después de tener una idea general del protocolo IP, a continuación se describirán las características del protocolo TCP.

[2]La escalabilidad está relacionada con la cantidad de nodos que puede soportar una red.

[3]La heterogeneidad se refiere a la manera como se establecen las comunicaciones entre diferentes tipos de redes.

2.2.6 TCP

Es el Protocolo de Control de Transmisión que proporciona a las capas superiores un protocolo orientado a conexión, que permite a una aplicación asegurarse de que un datagrama enviado sobre una red se reciba totalmente. TCP opera como protocolo de validación de mensajes proporcionando comunicaciones confiables ya que si algún datagrama se corrompe o se pierde, TCP es el encargado de manejar la retransmisión. También maneja el flujo de datagramas provenientes de las capas superiores, así como los datagramas de llegada provenientes de la capa IP. Tiene que asegurarse de que la prioridad y la seguridad se respeten. Debe mantener una tabla de estado, de todos los flujos de datos hacia dentro y afuera de la capa TCP [22].

Este protocolo reside en la capa de transporte, colocado encima de IP, pero debajo de las capas superiores y sus aplicaciones.

Debido a que TCP es un protocolo orientado a conexión, debe recibir mensajes de la máquina destino para poder enterarse de que recibió el datagrama. Por lo general se utiliza el término "circuito virtual" para referirse a la comunicación que hay entre dos máquinas terminales [22].

A grandes rasgos un mensaje se origina a partir de una aplicación y se pasa al TCP desde la capa superior siguiente de la arquitectura, a través de algún protocolo. El mensaje es pasado como un flujo, es decir, como una secuencia de caracteres individuales enviados en forma asíncrona, TCP entonces los ensambla en segmentos, en donde se va a añadir el encabezado del protocolo. Un segmento es un conjunto de datos, que está formado por un encabezado y puede contener o no, los datos que el usuario quiere transmitir. En el encabezado se incluye la suma de verificación y el número de secuencia si es que existe más de un segmento. Si se requiere una comunicación de dos vías, se establece una conexión o "circuito virtual" entre la máquina emisora y receptora [22].

Después de establecer el circuito virtual, se envía el segmento a IP, el que a su vez envía el mensaje a la red como datagrama. Una vez que se haya destinado el camino a través de la red, el IP de la máquina receptora pasa el segmento recibido a la capa TCP, donde se procesa y pasa a las aplicaciones superiores, mediante el uso de un protocolo de capa superior [22].

Todas las aplicaciones de capa superior que utilizan TCP tienen un número de puerto[4] que las identifica. El número de puerto, identifica un tipo de servicio que el sistema TCP está solicitando a otro.

Los números de puerto mayores a 255 se reservan para el uso de la máquina local,

[4]Un puerto es definido como un dispositivo que permite la entrada y salida de datos en una máquina, a través de un canal.

mientras que los menores a 255 se utilizan para procesos de uso frecuente. La Autoridad de Números de Internet Asignados (*Internet Assigned Numbers Authority*) publica una lista de los números de puerto de uso frecuente.

Cada circuito de comunicación dentro y fuera de la capa TCP se identifica en forma única mediante la combinación de dos números, los cuales en conjunto se conocen como socket. El socket se compone de la dirección IP de la máquina y del número de puerto utilizado. Ya que existe sólo una dirección IP por cada máquina y un número de puerto único para cada aplicación, entonces los números de socket serán también únicos. Esto permite que un proceso se comunique con otro a través de la red basándose en el número de socket [22].

En el capítulo 5 se describe como una aplicación de transmisión de imágenes es usada, se explica como es que se realiza la conexión entre el servidor y el cliente. Para realizar esta conexión, fue utilizado el lenguaje de programación Java, ya que permite con relativa facilidad hacer este tipo de conexiones y manejo de datos a través de una red.

2.2.7 Transmisión de datos en una red

Al transportar datos en una red es importante considerar algunas transformaciones que se le hacen a los datos originales, para tener un transporte confiable y en muchas ocasiones reducir la tasa de bits en la transmisión sobre los canales. Estas transformaciones pueden ser: preformateo, compresión y encriptamiento de los datos [24].

El preformateo se refiere al cambio en los datos que realiza un programa de una aplicación. Básicamente consiste en la representación de éstos en una computadora, o dicho de otra manera, la forma en la que son codificados, por ejemplo, como enteros, números de punto flotante, cadenas de caracteres o alguna estructura de datos compleja, con la finalidad de ser transportados por la red [24].

La compresión de datos se refiere al hecho de reducir la tasa de transmisión de bits que representan una porción de datos en particular, este proceso es realizado antes de pasarle datos al protocolo de transporte en el lado donde se envían los datos y al recibirlos la máquina destino tendrá que descomprimirlos antes de pasarle los datos a la aplicación para que los procese.

El encriptamiento se realiza para que los datos estén protegidos de posibles personas que se dediquen a interceptar mensajes y con ello, poder obtener información de cualquier usuario de una red. Además, el encriptamiento también es utilizado para asegurar la integridad de los datos [24].

En una red de computadoras, el flujo de datos o tráfico, como se le conoce comúnmente, es un aspecto a considerar desde el diseño de la red, ya que de éste dependerá el buen desempeño de ésta. Como se mencionó en el capítulo 1, las imágenes se adquieren digitalmente y generan regularmente una gran cantidad de datos, por tal motivo es importante poder contar con alguna técnica de compresión de datos para tratar de reducir el tráfico de la red y que el ancho de banda de los canales de comunicaciones no se sature.

Hasta aquí se ha mencionado un panorama genérico de las redes de computadoras. Se ha mencionado el modelo OSI con las capas que lo conforman y el modelo de conexión de internet indicando de manera resumida cómo se realiza una conexión entre dos máquinas en una red y las funciones del protocolo TCP/IP. El propósito de este capítulo es mencionar aspectos importantes de cómo se construye una red y poder resaltar las dificultades que existen en la transmisión de datos a través de ésta. En el siguiente capítulo se mencionarán algunos de los inconvenientes o ciertos procesos que se tendrán que aplicar a los datos de una imagen a transmitir para que se puedan transportar de forma óptima.

La técnica de compresión de imágenes que se propone en este trabajo de investigación, tiene como base una teoría desarrollada y estudiada desde los años 80's y que formará parte del proceso de transformación de la imagen, por lo que a continuación se describirán algunos fundamentos de esta teoría.

2.3 Wavelets

2.3.1 Introducción

Las "*wavelets*" en la actualidad han llegado a ser muy populares en las áreas de física y procesamiento de señales y se han extendido a otras como procesamiento de imágenes y de gráficas por computadora. El estudio de este tipo de señales surgió como una alternativa para estudiar y analizar algunos casos, en donde otro tipo de análisis, como el de Fourier, tiene algunas limitantes. Esto no quiere decir que el análisis con wavelets desplace al análisis con Fourier, sino que solamente es utilizado como una alternativa. El propósito de esta parte, es mostrar un panorama general de la utilización de la teoría de wavelets en el área de la compresión de imágenes, se describirán de manera breve y concisa, aquellos conceptos que son necesarios para entender como se realiza una transformación con wavelets, desde mencionar sus propiedades hasta ver como se aplica la Transformada Wavelet Discreta a una imagen. Como primer punto se mencionarán una serie de sucesos históricos que contribuyeron al estudio de esta teoría [19]:

- Hace más de un siglo, en 1873, Karl Weirstrass describió una familia de funciones que son construidas superponiendo copias escaladas de una función base dada.

- En 1909, Alfred Haar construyó el primer sistema ortonormal de funciones con soporte compacto.
- Los avances siguieron y en 1940 se usó por primera vez el término wavelet en el campo de la sismología, la cual fue inventada por Ricker y la usó para describir las perturbaciones que afectan a la superficie, cuando se origina un impulso sísmico o una carga explosiva.
- En 1946, Dennis Gabor descubrió unas bases no ortogonales de wavelets sin soporte compacto basadas en traslaciones de funciones Gaussianas.
- En 1982 Morlet mostró como las wavelets sísmicas de Ricker pueden ser modeladas con funciones matemáticas que Gabor había definido. Después Grossman y Morlet mostraron como estas señales arbitrarias podían ser analizadas en términos de escalas y traslaciones de una *Wavelet prototipo* (también llamada *Wavelet Madre*).
- Ya para 1988, Yves Meyer y Staphane Mallat desarrollaron esta noción en una teoría llamada *Análisis en Multiresolución*, posteriormente en 1989, Mallat mostró como el análisis en multiresolución puede ser visto como otra alternativa a un algoritmo piramidal, usado comúnmente en procesamiento de imágenes, junto con filtros cuadrados espejo (*QMF, Quadrature Mirror Filters*), usados regularmente en procesamiento de señales.

Nota: cabe aclarar que los siguientes temas muestran solo una introducción básica a la teoría del análisis de señales con wavelets, si el lector esta interesado en el tema y quiere profundizar, puede consultar las referencias citadas.

2.3.2 Wavelets

Las wavelets son funciones que deben satisfacer ciertos requerimientos. El nombre wavelet se deriva de uno de esos requerimientos, el cual indica que debe "ondear" u oscilar cruzando el eje de las abscisas o del tiempo, según sea el caso, y una de sus propiedades establece que, el área bajo la curva de esa señal debe ser igual a cero. El uso de las wavelets introdujo un término importante en el análisis de una señal, éste se refiere a la *escala*, en donde para un análisis con wavelets, lo importante es la *escala de la señal*. Esto quiere decir que, cuando se analiza una señal en una escala grande (o una ventana grande), no se podrán analizar detalles finos de la señal, mientras que cuando se analiza una señal en una escala pequeña, la señal puede ser analizada con mayor detalle. Más adelante en el texto se retomará esta idea de analizar una señal a diferentes escalas. Por lo pronto en la figura 2.5 se muestran ejemplos de wavelets [32].

Existen diversos tipos de wavelets, por lo que se puede escoger entre wavelets suaves ("*smooth*"), wavelets con soporte compacto, wavelets con filtros asociados o wavelets con expresiones matemáticas simples. Dependiendo del propósito para las cuales serán usadas se escogerá entre una u otra. Más adelante en el texto se hará

mención de las características de estas wavelets.

En un análisis de señales con Fourier, primeramente se parte de tratar de representar una señal por medio de sumas de funciones seno y coseno, por tanto este tipo de funciones forman la base del análisis. De igual forma en un análisis con wavelets, éstas son las funciones base utilizadas para representar una señal. Cabe recordar que una función base es aquella que es utilizada para representar cualquier función en un espacio [32].

Las wavelets son funciones generadas a partir de una sencilla función $\psi(x)$ (llamada wavelet *madre o prototipo*), una vez que es fijada esta función se pueden hacer dilataciones y traslaciones de esta función para crear otras bases, es decir, teniendo una función prototipo, se pueden crear distintas bases para representar a la señal, multiplicando ésta por un factor y haciendo desplazamientos en el eje de las abscisas. Por ejemplo, una ecuación para crear un conjunto de bases a partir de una wavelet se muestra a continuación:

$$\psi^{a,b}(x) = |a|^{-1/2}\psi\left(\frac{x-b}{a}\right) \tag{2.1}$$

en donde a y b es conveniente que tomen valores especiales para definir las bases, así estos pueden estar dados por: $a = 2^{-j}$ y $b = k2^{j}$, donde k y j son enteros [32]. En la ecuación 2.1 la variable a refleja la escala de una función base en particular, mientras que la variable b especifica la posición de traslado a lo largo del eje de las abscisas [3].

La definición de las bases wavelet como traslaciones se refiere a recorrer éstas a lo largo del eje de las abscisas y las dilataciones significan que se varía la frecuencia de la señal. De esta forma, para la ecuación 2.1, las wavelets de alta frecuencia corresponden cuando $a < 1$, o sea que son cortas en el tiempo, mientras que las wavelets de baja frecuencia corresponden cuando $a > 1$, esto es que son más largas en el tiempo [2]. Esto se puede notar en la figura 2.5.

2.3.3 Propiedades de las wavelets

La clase de funciones que se analizarán para representar una transformada wavelet son aquellas que son cuadrablemente integrables (*square integrable functions*) sobre la línea real (por ejemplo, el conjunto de todos los números reales), esta clase es denotada por $L^2(R)$ [30].

La propiedad más importante de las wavelets es que éstas forman una base ortogonal (ecuación 2.2), para el espacio $L^2(R)$. Esta propiedad hace que las wavelets tengan buena localización, característica que en ocasiones se dice que están bien localizadas, esta propiedad se debe a que muchas wavelets tienen soporte compacto, lo cual hace que se puede hacer un buen análisis de una señal tanto en frecuencia como

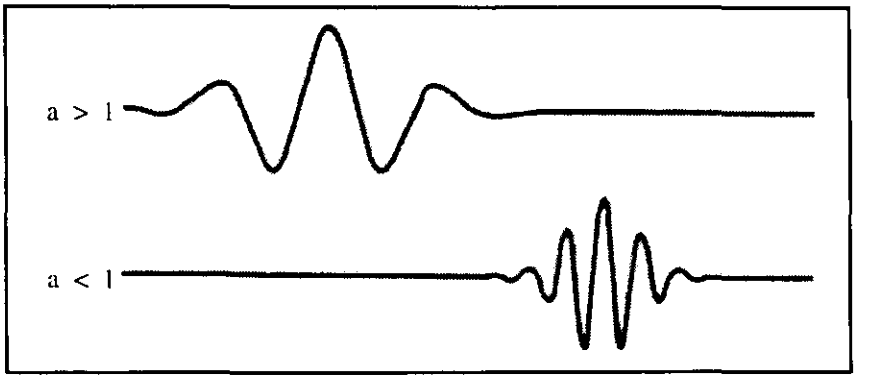

Figura 2.5: Wavelets de alta y baja frecuencia.

en el tiempo. Como se mencionó anteriormente la transformada de Fourier usa como sus funciones base a las funciones seno y coseno, las cuales son ortogonales, pero no tienen soporte compacto, lo cual hace que las funciones no esten localizadas [30].

$$\int_{-\infty}^{\infty} \psi^{j,k}(x)\psi^{J,K}(x)dx = \quad \text{producto interno de} \quad \psi^{j,k} \quad y \quad \psi^{J,K} = 0 \tag{2.2}$$

Otra forma de escribir la ecuación 2.2 y que es muy utilizada en la literatura es:

$$\left\langle \psi^{j,k}, \psi^{J,K} \right\rangle = 0 \tag{2.3}$$

El soporte compacto se refiere a un intervalo en donde la función no es cero sobre el dominio entero. Por ejemplo, puede ser que una función sea cero desde $-\infty$ (menos infinito) hasta cierto intervalo y después de éste y hasta el ∞ (infinito), la función seguirá siendo cero. Regresando al caso de Fourier sus funciones base no tienen un valor constante igual a cero durante un intervalo en el dominio entero, por eso estas funciones no tienen soporte compacto [30].

Normalmente la wavelet prototipo es centrada en el origen, por lo que $\psi^{a,b}(x)$ es centrada en $x = b$ y a partir de ésta se empiezan a hacer traslaciones y dilataciones para formar las otras wavelets base [28]. En la figura 2.6 se muestra un ejemplo de la siguiente wavelet:

En la figura 2.7 se muestran otro tipo de wavelets (de la familia Daubechies):

Hasta este momento podemos ver a las wavelets como simples funciones que satisfacen ciertos requerimientos, cuentan con ciertas propiedades y sirven para representar a una función. A continuación se explica como estas funciones se utilizan para definir una transformación.

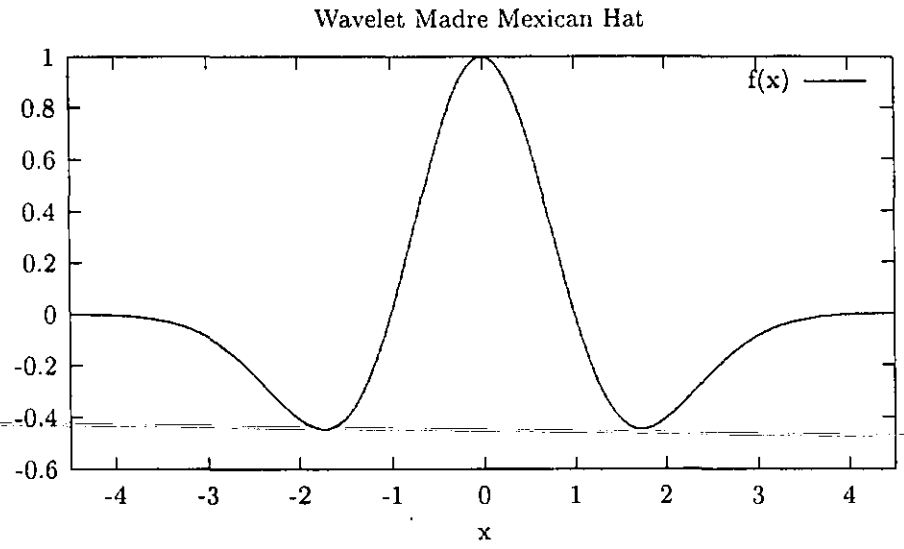

Figura 2.6: Wavelet Mexican Hat

Regresando a la analogía con Fourier, cuando se explica éste análisis, en muchas ocasiones se describe primero la serie de Fourier, después y a partir de ésta puede ser definida la "transformada continua de Fourier" y partir de esta última, las consecuentes "transformada discreta" y "transformada rápida", aunque no necesariamente tenga que ser así. De la misma forma con las wavelets, se puede primero explicar como se forma la "serie de expansión con wavelets", luego mencionar como se calcula la "transformada continua wavelet" y luego la "transformada discreta wavelet (DWT, por sus siglas en inglés)", aunque para ésta última primero hay que puntualizar algunos conceptos importantes, como bancos de filtros, codificación en subbandas y análisis en multiresolución, ya que ayudarán a entender su interpretación y la manera en la que se obtiene.

2.3.4 La serie de expansión con wavelets

Como se mencionó anteriormente, un conjunto de funciones wavelet pueden ser utilizadas para representar una señal. De nueva cuenta una wavelet prototipo es escalada y trasladada para formar un conjunto de funciones base, solo que para este caso los escalamientos se harán de manera "*binaria*", es decir, por factores de dos, y las traslaciones serán del tipo "*dyadic*". Una traslación "*dyadic*" es un corrimiento por la cantidad de $k/2^j$, la cual es un entero múltiplo del factor de escalamiento.

Una wavelet típica y $\psi^{j,k}$ que es escalada j veces y recorrida k veces, es mostrada

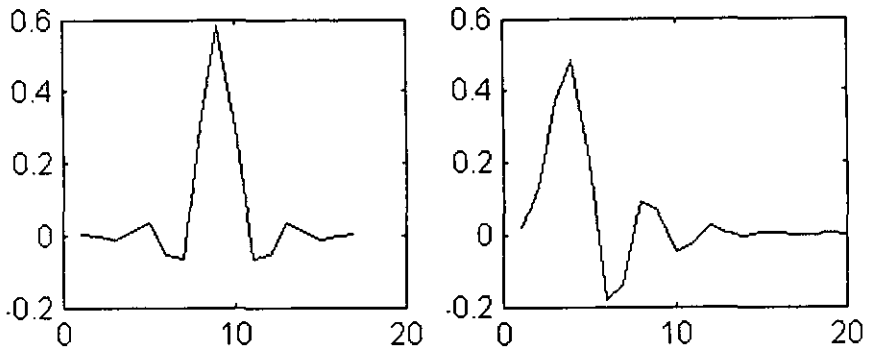

Figura 2.7: Ejemplos de wavelets.

en la siguiente ecuación:

$$\psi^{j,k}(x) = 2^{J/2}\psi\left(2^j x - k\right) \tag{2.4}$$

Se puede definir una base como un conjunto de funciones linealmente independientes que pueden ser usadas para producir todas las funciones admisibles $f(x)$, así la serie de expansión con wavelets está definida por [30]:

$$f(x) = \text{combinación de funciones base} = \sum_j \sum_k c_{j,k}\psi^{j,k}(x) \tag{2.5}$$

la ecuación anterior quiere decir que una función $f(x)$ puede ser representada por la suma de los coeficientes de peso $c_{j,k}$ multiplicados por el conjunto de funciones wavelet donde los $c_{j,k}$ están dados por:

$$c_{j,k} = 2^{j/2}\int_{-\infty}^{\infty} f(x)\psi\left(2^j x - k\right)dx \tag{2.6}$$

Para describir o enunciar la transformada de Fourier a partir de la serie de Fourier, se hacen algunas suposiciones, como decir que el periodo de la función va a tender a infinito, y a raíz de esto, es posible deducir la transformada continua de Fourier[12]. En el caso de las wavelets no se puede hacer la misma suposición, ya que para aplicar la serie de expansión con wavelets a una función, no se restringe que ésta sea periódica, por lo que para describir la transformada continua, se puede solo suponer que los coeficientes que multiplican a las bases wavelets, están dados por una transformada wavelet. En el siguiente tema se observará esta característica.

2.3.5 Transformada Wavelet Contínua (CWT)

Como se ha estado mencionando anteriormente la técnica de compresión de este proyecto de investigación utiliza a la transformada wavelet como herramienta principal para lograr dicho propósito. Para poder describir y comprender como se realiza

una transformación con wavelets a veces es necesario hacer una comparación de ésta con la Transformada de Fourier, ya que la idea de la transformación es la misma y utiliza los mismos principios. Como una computadora regularmente trabaja con funciones discretas, no se enfatizará demasiado este tema, sino que el propósito es meramente el formalismo que debe haber para poder describir la transformada wavelet discreta, ya que la discreta se basa en la continua.

En una transformación, cada uno de los coeficientes son determinados por medio del producto interno entre una función de entrada y una de las funciones base. Esto representa en algún sentido el grado de similitud que existe entre la función de entrada y una función base particular, por esto mismo, el producto interno entre dos funciones base es igual a cero, ya que entre ellas no debe existir ninguna similitud. Entonces si una señal o imagen esta formada de componentes que son similares a algunos componentes de una o varias funciones base, entonces, al realizar el producto interno entre ellas, lo que se obtendrá son algunos coeficientes que son muy pequeños o muy cercanos a cero. De esta forma, muchos de los términos de una transformación pueden ser ignorados y la señal o imagen se puede representar de una forma más compacta [3].

La idea básica de la transformada wavelet es representar cualquier función como una superposición de wavelets. Tal superposición descompone a la función en diferentes niveles de escala, donde cada nivel de escala es además descompuesto en una resolución adaptada al nivel [30].

La *transformada wavelet contínua* de una función $f(x)$ con respecto a la wavelet prototipo $\psi(x)$, está dada por:

$$W_f(a,b) = \left\langle f, \psi^{a,b} \right\rangle = \int_{-\infty}^{\infty} f(x)\psi^{a,b}(x)dx \tag{2.7}$$

donde $\psi^{a,b}(x)$ puede estar definida por la ecuación (2.1).

Se puede notar que también la transformada wavelet como la transformada de Fourier, es obtenida a partir de la integral del producto de la función que se quiere transformar con las funciones base pertenecientes al espacio de análisis.

Los coeficientes de la transformada wavelet son dados por los productos internos de la función que está siendo transformada por cada una de las funciones base [3]. Estos coeficientes son muy parecidos a los obtenidos en la serie de expansión y la única diferencia es el factor de escala.

La transformada wavelet inversa continua es:

$$f(x) = \frac{1}{C_\psi} \int_0^{\infty} \int_{-\infty}^{\infty} W_f(a,b)\psi^{a,b}(x)db\frac{da}{a^2} \tag{2.8}$$

Se puede notar que la transformada wavelet continua $W_f(a,b)$ de una función $f(x)$ de una dimensión es una función de dos variables, una más que la función $f(x)$. De esta forma la CWT se dice que está sobrecompletada y puede interpretarse como si mantuviera más información, lo cual puede producir que se incremente el espacio de almacenamiento [3].

Extendiendo esta definición a dos dimensiones, la transformada wavelet continua de una función $f(x,y)$ está dada por:

$$W_f(a,b_x,b_y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\psi_{a,b_x,b_y}(x,y)dxdy \tag{2.9}$$

donde b_x y b_y especifican las traslaciones en las dos dimensiones.

Uno de los aspectos importantes de la transformada wavelet es que puede interpretarse como un banco de filtros, por ejemplo si se define:

$$\bar{\psi}_a(x) = \psi_a^*(-x) \tag{2.10}$$

la cual es el complejo conjugado de la wavelet escalada. Entonces la CWT puede ser reescrita como:

$$W_f(a,b) = \int_{-\infty}^{\infty} f(x)\bar{\psi}_a(b-x)dx = f * \bar{\psi}_a \tag{2.11}$$

la cual expresa la convolución de $f(x)$ con la wavelet conjugada en la escala a.

En la figura 2.8 se observa como la CWT se puede interpretar como un banco de filtros lineal actuando sobre la función $f(x)$. Cada valor de a define la respuesta impulso de un filtro pasobanda, por lo que entonces todas las salidas estarán filtradas y al tomarlas juntas se obtiene la CWT de la función de entrada [3].

De lo anterior puede notarse como existe la relación de otros conceptos, como el de bancos de filtros, para interpretar y entender como funciona una transformada wavelet, de hecho es común que en gran parte de la literatura se utilice este ejemplo para describir a la transformada, por tanto a continuación se explica de manera breve que es un banco de filtros.

2.3.6 Bancos de Filtros

Esta herramienta ha sido utilizada más que nada en el procesamiento de señales acústicas y análisis de voz. El principio básico es descomponer una señal en diferentes bandas de frecuencias y procesar cada banda separadamente ya que la implementación de este tipo de filtrado es relativamente sencilla [3].

Un filtro, es un operador lineal invariante con el tiempo. En el caso de señales discretas, éste actúa sobre vectores de entrada. El vector h por lo regular contiene los

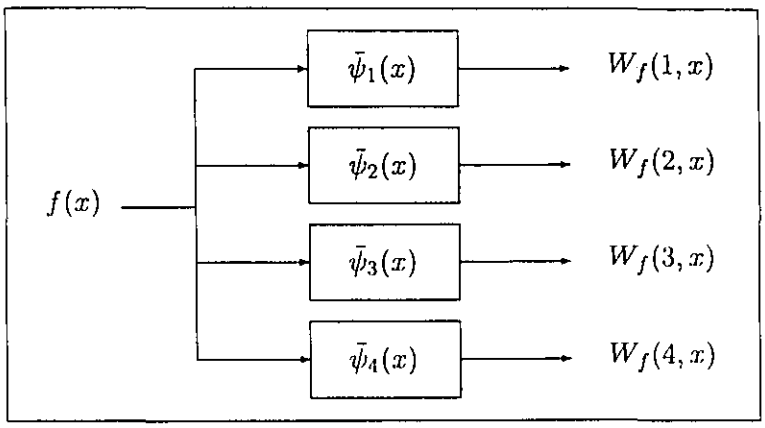

Figura 2.8: Interpretación de la CWT como un banco de filtros

coeficientes del filtro, $h(0), h(1), ..., h(n)$, así una salida filtrada $y(n)$ será entonces [30]:

$$y(n) = \sum_k h(k)x(n-k) \rightarrow \text{convolución } h * x \text{ en el dominio del tiempo} \qquad (2.12)$$

Este método es precursor del análisis en *tiempo-frecuencia* en donde las componentes son desplegadas en un espacio de dos dimensiones, las cuales son tiempo de ocurrencia y frecuencia de oscilación [3].

En un análisis en frecuencia, si se particiona el eje de frecuencia en un conjunto de intervalos de frecuencia (adyacentes y sin traslaparse) y se usa esta partición para definir un conjunto de filtros pasobanda ideales, se obtendrá lo que se llama un **Banco de Filtros** [3]. En un caso ideal se pretende que las funciones de transferencia de los filtros sean como los que se muestran en la figura 2.9, en ésta se puede notar como para distintos intervalos de frecuencia se tiene una función que corresponde a un filtro. En la figura 2.10 se muestra un diagrama de un banco de filtros pasobanda.

Se puede notar en la figura 2.10 como la señal de entrada es alimentada a cada uno de los filtros. Una de las características del banco de filtros es que las funciones de transferencia de los filtros $H_i(s)$ son construidas de manera que la suma sea igual a 1 para todas las frecuencias, así las salidas $g_i(x)$ (que son formadas por las sumas de convolución de los filtros y las señales de entrada) se sumarán para formar $f(x)$ y reconstruir la señal [3]. Esto es:

$$\sum_{i=1}^{\infty} H_i(s) = 1 \Rightarrow \sum_{i=1}^{\infty} g_i(x) = f(x) \qquad (2.13)$$

En principio un banco de filtros es el conjunto de filtros que se utilizan para descomponer a una señal en un número fijo de bandas de frecuencia, después se

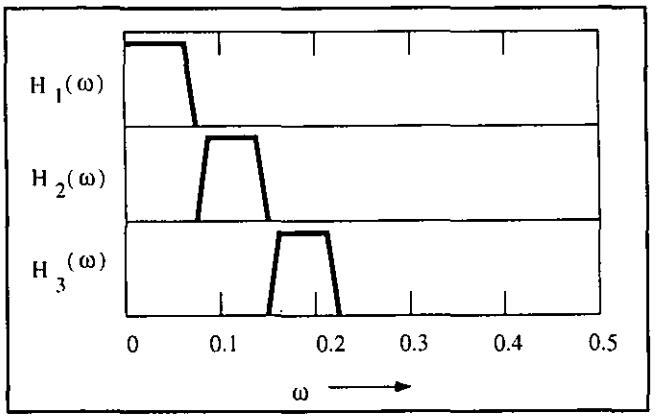

Figura 2.9: Funciones de transferencia de un banco de filtros.

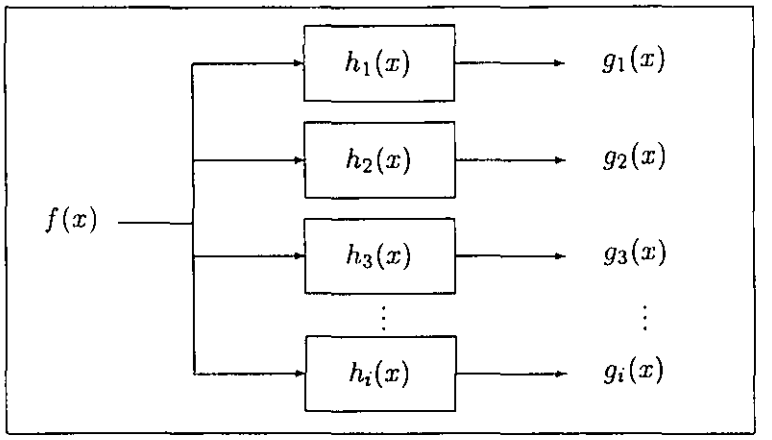

Figura 2.10: Banco de filtros paso banda

analizarán éstas, pero siempre es importante poder reconstruir la señal original. Para esto también son utilizados ciertos filtros que tienen que cumplir con ciertas características y que garantizan la perfecta reconstrucción de la señal. A todo este proceso de descomponer la señal y reconstruirla, en gran parte de la literatura citada se le conoce como el "*Análisis y Síntesis por medio de bancos de filtros*". Entre el proceso de descomposición y reconstrucción puede existir aparte, otro tipo de procesamiento que dependerá del tipo de aplicación para la cual se utiliza el banco de filtros. Una de ellas es conocida como "*codificación en subbandas*", que utiliza un banco de filtros dentro de su proceso, pero cabe aclarar que no se deben confundir ya que en muchas ocasiones se menciona como la misma cosa. La codificación en subbandas se describirá más adelante.

Lo anterior muestra una idea general de un banco de filtros, lo interesante resulta ser, como se conectan los bancos de filtros con las wavelets y como puede ser aplicada

esta conexión a imágenes para formar una técnica de compresión.

Junto con la idea de los bancos de filtros se ha mencionado que la transformada wavelet es utilizada en varias escalas y que de hecho, el análisis con wavelets se hace en el espacio del tiempo–escala. El concepto de escala tiene que ver con el concepto de "resolución", ya que al decir que una señal es analizada es determinada escala, es lo mismo a decir que es analizada en una resolución especícifica. Por tanto en el siguiente tema se mostrará el concepto de Análisis en Multiresolución, el cual es importante conocer ya que ayuda a entender el objetivo de la transformada wavelet discreta, además de que esta herramienta fue la primera que utilizó las wavelets para procesar imágenes y que se adapta muy bien para tener una técnica de compresión de imágenes.

2.3.7 Análisis en Multiresolución

Muchos de los desarrollos en el análisis con wavelets entraron en un campo que se llama "*análisis en multiresolución*" [3]. Una idea muy general del análisis en multiresolución, puede ser el hecho de analizar una señal o una imagen en diferentes escalas y diferente resolución. Un ejemplo muy común para ilustrar esta idea es la Cartografía, en este campo se pueden tener mapas dibujados a diferentes escalas, en este caso, la escala es la razón del tamaño actual de un territorio a aquella que se representa en el mapa. En un mapa con una escala grande, los objetos grandes como océanos, mares, continentes, son mas visibles, mientras los detalles, como la calle de una ciudad por ejemplo, no lo son. En un mapa con una escala muy pequeña este tipo de detalles son más visibles, mientras que los objetos grandes señalados anteriormente se van perdiendo [3].

Para poder describir el concepto de Multiresolución, se presenta a continuación un ejemplo que muestra de manera sencilla la idea del análisis en multiresolución, cabe mencionar que este ejemplo es con base en una de las señales wavelets más viejas (alrededor de 1909) y más sencilla, la cual es llamada "Wavelet Haar".

2.3.7.1 Wavelet Haar

Se había comentado anteriormente que una base en un espacio puede describir cualquier función que pertenezca a ese espacio. La base Haar es la base wavelet más simple. Para ver las características de ésta, primero se discutirá como una función de una dimensión puede ser descompuesta usando wavelets Haar, describiendo para esto sus funciones base [28].

Suponiendo que se tiene una función discreta de una dimensión con solo cuatro valores:

$$f(x) = \begin{bmatrix} 9 & 7 & 3 & 5 \end{bmatrix} \tag{2.14}$$

Resolución	Promedios	Coeficientes detalle
4	[9 7 3 5]	
2	[8 4]	[1 −1]

Cuadro 2.3: Promedios y coeficientes detalle del análisis.

Resolución	Promedios	Coeficientes detalle
4	[9 7 3 5]	
2	[8 4]	[1 −1]
1	[6]	[2]

Cuadro 2.4: Promedios y coefientes detalle de las tres resoluciones.

podemos representar esta función en la base Haar, calculando una transformada wavelet como sigue:

primero se promedian los elementos por pares, obteniendo así una nueva función:

$$f(x) = [\ 8 \quad 4\ ] \tag{2.15}$$

se puede notar claramente que parte de la información ha sido perdida en el proceso de promediar. Para poder recuperar los cuatro valores originales de la función a partir de los elementos promediados, es necesario almacenar algunos "coeficientes detalle", los cuales capturan la pérdida de información. Para el ejemplo anterior, se escogerá 1 para el primer coeficiente detalle, ya que para el promedio calculado es 1 menor que 9 y 1 más que 7. Este procedimiento permitirá recuperar los dos primeros pixeles de la imagen original de cuatro pixeles. Similarmente el segundo coeficiente detalle es -1, ya que 4 + (-1) = 3 y 4 - (-1) = 5.

Así se ha descompuesto la función original en dos funciones, una con los elementos promediados y la otra que contiene a los coeficientes detalle.

De acuerdo a este análisis se puede decir que la resolución de la función original es igual a 4, definiendo en este caso a la resolución como el número de elementos de la función. De esta forma la resolución de la función de los coeficientes promediados será igual a 2. El cuadro 2.3 muestra la salida de este primer paso de la transformada.

Repitiendo este proceso de manera recursiva sobre los promedios, se obtiene una descomposición completa de la función original, los coeficientes se muestran en el cuadro 2.4.

Se puede definir la transformada wavelet de este análisis (en ocasiones llamada descomposición wavelet) de la función original de cuatro elementos, como la representación con un solo coeficiente del promedio global de la función original, seguido

por los coeficientes detalle [28]. Así, para la base Haar, la transformada wavelet de la señal original está dada por:

$$W_f(p) = \begin{bmatrix} 6 & 2 & 1 & -1 \end{bmatrix} \tag{2.16}$$

Cabe aclarar que en este proceso no se ganó ni se perdió información, la función original tiene cuatro coeficientes, los mismos que su transformada, también se puede notar que, dada la transformada se puede reconstruir la imagen original desde cualquier resolución, sumando y restando recursivamente los coeficientes detalle desde sus versiones de baja resolución, de esta forma se puede ir desde la resolución 1 a la 2 y de la 2 a la 4.

Una ventaja al almacenar la transformada wavelet de la función original, más que la función misma, es que la transformada y en particular los coeficientes detalle tienden a ser muy pequeños en magnitud, por lo que muchas veces podemos truncar o simplemente quitar estos coeficientes, llevando solamente esto a pequeños errores en la reconstrucción de la función.

Como se mencionó anteriormente la wavelet Haar es una de las wavelets más sencillas, por lo cual, es muy utilizada para ejemplificar como funciona la transformada wavelet y para entender el concepto de análisis en multiresolución. A continuación se explicará de manera poco formal, como a partir de la wavelet Haar, se puede interpretar el concepto de la descomposición de una señal en varias resoluciones (multiresolución).

En el capítulo 1 se mostró que una imagen puede ser manejada matemáticamente como una función discreta de dos dimensiones. Pero a manera de ejemplificar podemos decir que una imagen puede ser una función discreta de una dimensión, en donde los elementos serán los valores de los pixeles y la cantidad de estos determinarán la resolución.

Considerando ahora solamente un intervalo semi-abierto [0,1) y estableciendo que una imagen puede estar formada por funciones constantes, es decir, una imagen de un pixel puede estar formada por una función constante en el intervalo de [0,1); y al espacio vectorial que ocupa esta función se le puede denominar V^0. De la misma manera se puede considerar una imagen de dos pixeles, en donde ahora estará formada por dos funciones constantes, cada una en el intervalo en [0,1/2) y [1/2,1), de esta forma el espacio vectorial que forman esas funciones puede ser llamado V^1. Si se continua de esa manera, el espacio V^j incluirá todas las funciones constantes definidas sobre cada uno de los 2^j intervalos igualmente espaciados. En el cuadro 2.5 se muestra esta idea de manera clara.

Se puede notar que cada vector en V^j es también contenido en V^{j+1}. De esta manera, se puede describir una función constante con dos intervalos como una función

Niveles	Pixeles	Intervalo	Espacio
0	$2^0 = 1$	$[0, 1)$	V^0
1	$2^1 = 2$	$[0, 1/2), [1/2, 1)$	V^1
2	$2^2 = 4$	$[0, 1/4), [1/4, 1/2), [1/2, 3/4), [3/4, 1)$	V^2
⋮	⋮	⋮	⋮
J	2^J pixeles	2^j intervalos	V^j

Cuadro 2.5: Intervalos en un espacio vectorial conforme al número de pixeles.

constante con cuatro intervalos, con cada intervalo de la primera función correspondiendo a un par de intervalos de la segunda función, como se muestra en la figura 2.11:

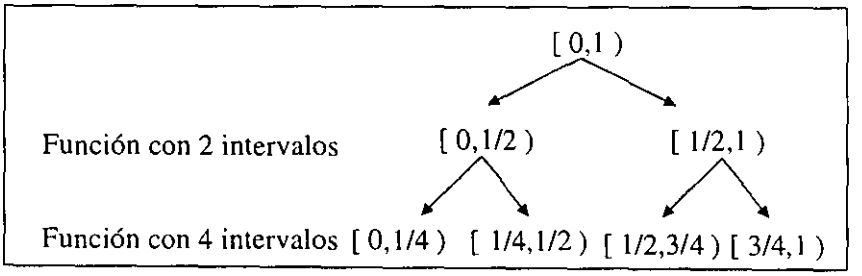

Figura 2.11: Asignación de intervalos a funciones.

Lo que se pretende con la idea anterior, es mostrar como existe una especie de anidamiento entre los espacios, ya que el espacio V^{j+1} contiene al V^j, es decir:

$$V^0 \subset V^1 \subset V^2 \subset \cdots$$

la teoría del análisis en multiresolución establece que estos espacios tienen que estar anidados [18].

Ahora se necesita definir una base para cada espacio vectorial V^j. Estas bases son llamadas funciones de escalamiento (*scaling functions*), se denotan usualmente por el símbolo ϕ. Una base simple para V^j es el conjunto de funciones "*box*" (escalón) escaladas y trasladadas por 2^j y k respectivamente:

$$\phi^{j,k}(x) = \phi(2^j x - k) \quad k = 0, \ldots, 2^j - 1 \tag{2.17}$$

donde:

$$\phi(x) = \begin{cases} 1 & \text{para } 0 \leq x \leq 1 \\ 0 & \text{cualquier otro caso} \end{cases}$$

En la figura 2.13 se muestran las cuatro funciones que forman la base para el espacio V^2:

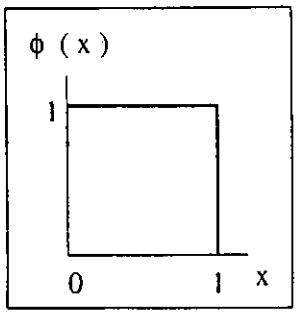

Figura 2.12: Función "box"

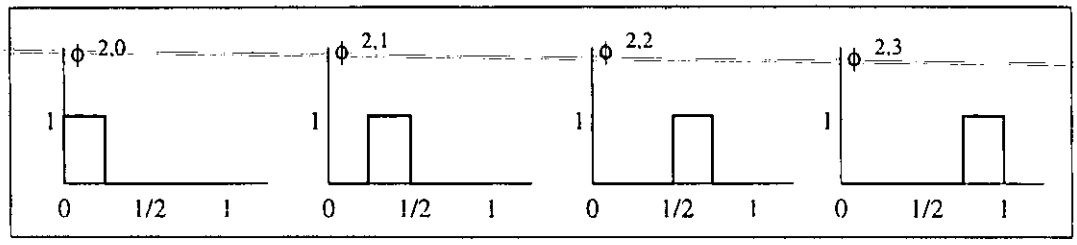

Figura 2.13: Funciones base para el espacio V^2

El primer paso para hacer un análisis en multiresolución, es definir las bases de los espacios vectoriales.

Ahora se define W^j como un nuevo espacio vectorial, el cual tiene la característica de ser el complemento ortogonal de V^j en V^{j+1}, es decir, W^j es el conjunto de todas las funciones en V^{j+1} que son ortogonales a todas las funciones en V^j bajo un producto interno.

Lo interesante es que para el espacio W^j, el conjunto de bases que lo extienden son las wavelets $\psi(x)$. Estas funciones base tienen dos propiedades importantes:

1. Las funciones base de W^j, junto con las funciones base de V^j, forman una base para V^{j+1}.

2. Cada función base de W^j es ortogonal a cada una de las funciones base de V^j bajo el producto interno elegido.

Informalmente podemos pensar que las wavelets en W^j representan las partes de las funciones en V^{j+1} que no pueden ser representadas en V^j.

Las wavelets correspondientes a las bases "box" (o escalón) y que forman el complemento ortogonal a éstas, son conocidas como wavelets Haar, y son descritas

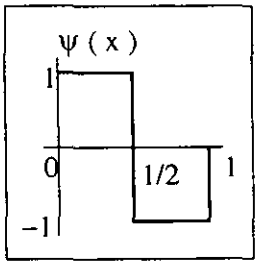

Figura 2.14: Función wavelet Haar

por:

$$\psi^{j,k}(x) = \psi(2^j x - k) \qquad k = 0, \ldots, 2^j - 1 \tag{2.18}$$

donde:

$$\psi(x) = \begin{cases} 1 & \text{para } 0 \leq x \leq 1/2 \\ -1 & \text{para } 1/2 \leq x \leq 1 \\ 0 & \text{cualquier otro caso} \end{cases}$$

En la figura 2.15 se muestran las dos wavelets Haar que extienden al espacio W^1:

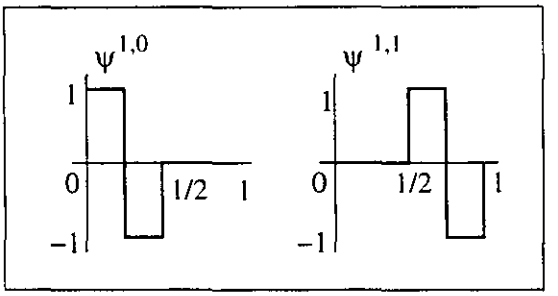

Figura 2.15: Bases Haar para el espacio W^1

Retomando el ejemplo de la función discreta de cuatro elementos mostrado anteriormente, se aplicarán estas ideas para representar a la función original $f(x)$ como una combinación lineal de funciones base Haar del espacio V^2 (resolución 4):

$$f(x) = c_{2,0}\phi^{2,0}(x) + c_{2,1}\phi^{2,1}(x) + c_{2,2}\phi^{2,2}(x) + c_{2,3}\phi^{2,3}(x) \tag{2.19}$$

su representación gráfica es (figura 2.16):

cabe notar que los coeficientes $C_{2,0}, \ldots C_{2,3}$ son los valores de los pixeles de la imagen original ([9 7 3 5]).

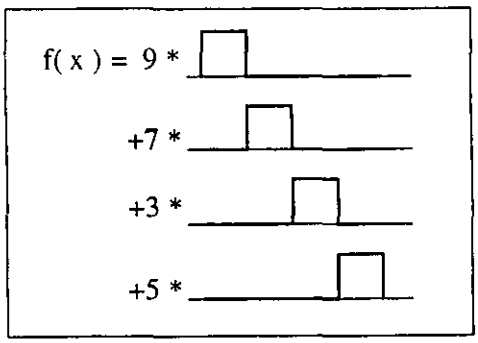

Figura 2.16: Representación de la función con las bases en la resolución 4

Se puede reescribir la función $f(x)$ en términos de las funciones base en V^1 y W^1 (resolucion 2):

$$f(x) = c_{1,0}\phi^{1,0}(x) + c_{1,1}\phi^{1,1}(x) + d_{1,0}\psi^{1,0}(x) + d_{1,1}\psi^{1,1}(x) \tag{2.20}$$

En la figura 2.17 se muestra su representación gráfica:

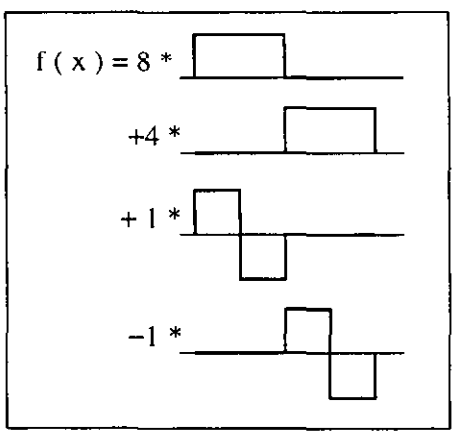

Figura 2.17: Representación de la función con las bases en la resolución 2

Finalmente se puede volver a escribir la función $f(x)$ como la suma de funciones base en V^0, W^0 y W^1 (resolución 1):

$$f(x) = c_{0,0}\phi^{0,0}(x) + d_{0,0}\psi^{0,0}(x) + d_{1,0}\psi^{1,0}(x) + d_{1,1}\psi^{1,1}(x) \tag{2.21}$$

y su representación gráfica se muestra en la figura 2.18:

Otra vez, estos cuatro coeficientes son la transformada wavelet Haar de la imagen original [28].

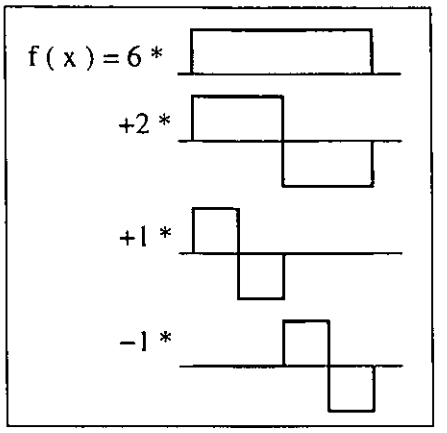

Figura 2.18: Representación de la función con las bases en la resolución 1

El ejemplo anterior mostró en una forma muy básica, lo que se pretende hacer con un análisis en multiresolución, es decir, se mencionó como una función sencilla puede ser representada por una sumatoria de distintos coeficentes calculados mediante una regla determinada, multiplicando a un conjunto de funciones base de algún espacio; la teoría del análisis en multiresolución establece la relación que debe existir entre estas funciones base.

Resumiendo, en un análisis de multiresolución se tienen funciones de escalamiento $\phi(2^j t - k)$ que son las que permiten representar una función a un nivel j (funciones base). Los nuevos detalles en ese nivel, son representados por las wavelets $\psi(2^j t - k)$. Entonces las funciones ϕ y las ψ se combinan en una multiresolución en el nivel más fino $j + 1$ (también conocido como de mayor detalle). Los promedios de la señal vienen de las funciones de escala (ϕ) y los detalles vienen de las wavelets (ψ).

La idea de la multiresolución es muy intuitiva. Si se tiene la aproximación a una señal en una resolución correspondiendo a V_0, una mejor aproximación puede ser obtenida sumándole los detalles correspondientes a W_0. Esto es, sumar los coeficientes de peso de las wavelets en esa escala. Por lo que entonces si se sigue esta idea, cualquier función o señal dentro del espacio puede ser vista como la suma con coeficientes de peso de funciones wavelets en escalas más y más finas.

La explicación y definición formal de esta teoría, está descrita en el artículo publicado por Stephane Mallat en 1989 [18].

Los conceptos anteriores están enfocados a describir como se obtiene la transformada wavelet discreta, aunque si bien, esta transformada parte de la transformada continua, existe una serie de conceptos independientes que pueden ser utilizados para ver como se interpreta la transformada wavelet discreta, estos conceptos son,

una transformación jerárquica, la cual es conocida como "la pirámide laplaciana" y la codificación en subbandas, por lo cual, a continuación se explican de manera general.

2.3.8 La Pirámide Laplaciana

Un análisis en multiresolución es parecido a lo que se denomina algoritmos en forma de pirámide, en donde la idea principal es tener un conjunto de imágenes de modo que puedan ser clasificadas de forma jerárquica. La Pirámide Laplaciana es un esquema de codificación que aportó las ideas iniciales para realizar un análisis en multiresolución.

La Pirámide Laplaciana es un trabajo realizado por Peter Burton y Edward Adelson [4], en donde su esquema de codificación está basado en una función Gaussiana. Una imagen en este esquema es primero filtrada con un filtro paso bajas y la imagen resultante es restada de la imagen original. Reteniendo con este proceso los detalles de alta frecuencia de la imagen [3].

Se mencionará a grandes rasgos como se realiza el proceso de codificación basado en este esquema. Posteriormente se podrá observar que este proceso es muy parecido al análisis en multiresolución basado en la transformada wavelet.

Sea $f_0(i,j)$ la imagen original y $g(i,j)$ un filtro paso bajas Gaussiano. Entonces en cada paso del proceso de codificación, la imagen es descompuesta en componentes $f_1(i,j)$ de baja frecuencia con la mitad de resolución de la imagen y componentes $h_1(i,j)$ de alta frecuencia con la misma resolución que la imagen, es decir:

$$f_1(i,j) = [f_0 * g](2i, 2j) \quad \text{y} \quad h_1(i,j) = f_0(i,j) - [f_0 * g](i,j) \tag{2.22}$$

Después f_1 será descompuesta de la misma manera y así sucesivamente hasta que f_n sea un solo punto. Después de n iteraciones de una imagen de $N \times N$, donde $N = 2^n$, $f_n(i,j)$ es la imagen resultante final (de 1×1 pixel) o el punto que se había mencionado. El proceso de codificación consiste en mantener esta imagen final y todas las $h_k(i,j)$ obtenidas.

La decodificación de la imagen es realizada en orden inverso. Tomando la última imagen en el proceso de codificación se insertarán una columna y un renglón de ceros entre cada una de los pixeles en la imagen, en el caso de la imagen de 1×1 se insertarán solo una columna y un renglón de ceros, obteniendo así una imagen de 2×2, después a esta imagen se le aplica el operador de convolución con el filtro $g(i,j)$. Después el resultado es sumado a $h_k(i,j)$ para obtener $f_{n-1}(i,j)$. Repitiendo el proceso para cada imagen resultante se reconstruirá la imagen original sin error [3]. En la figura 2.19 se muestra el proceso de codificación.

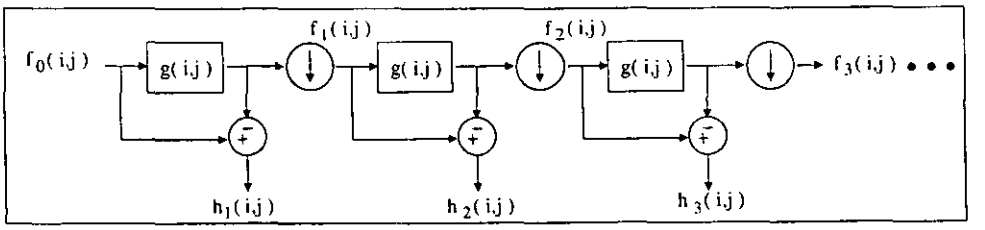

Figura 2.19: Esquema de codificación de la Pirámide Laplaciana

Con un poco de imaginación se puede ver este esquema de codificación como si fuera en forma de pirámide, es decir, se puede colocar a la imagen original como la base de la pirámide y las imágenes filtradas obtenidas, como los siguientes escalones de la pirámide. El objetivo de la estructura piramidal es que a partir de la imagen que se encuentre en la parte más alta de la pirámide, se pueda obtener la imagen previa a ésta, realizando el proceso de decodificación , en este proceso solamente va a ser requerida esta imagen, la imagen con detalles de alta frecuencia y el filtro Gaussiano, figura 2.20

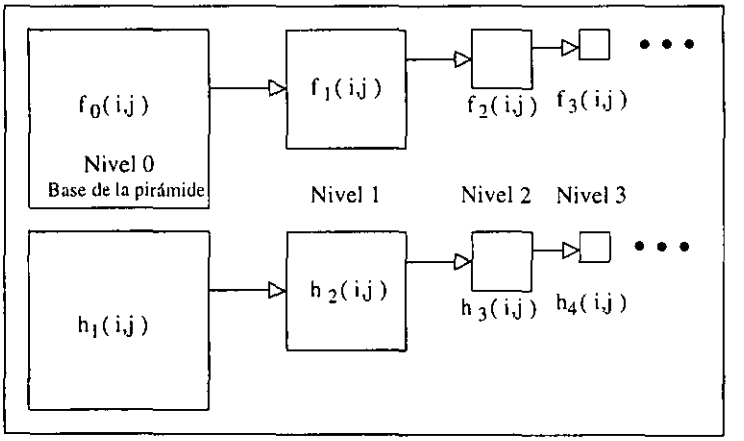

Figura 2.20: Conjunto de imágenes de la Pirámide Laplaciana

Mientras que en la teoría de los bancos de filtros se plantea el análisis en las bandas de frecuencias, dos de los conceptos importantes que se mostraron para el análisis con wavelets son: "escala y resolución". La *escala* está relacionada con el tamaño de la señal, mientras que la *resolución* está ligada a la cantidad de los detalles presentes en la señal [1].

En el esquema mostrado anteriormente, el cambio de resolución es obtenido cuando se filtra la señal de entrada (los detalles de alta frecuencia se pierden), mientras que el cambio de escala se debe al proceso de eliminar muestras (*downsampling*).

Por último y para definir a la DWT (transformada wavelet discreta), se muestra a continuación lo que se conoce como codificación en subbandas.

2.3.8.1 Codificación en subbandas

La codificación en subbandas es una técnica que describe un análisis en tiempo-frecuencia. Originalmente esta técnica fue desarrollada para codificación de señales de audio digital y de voz [1].

La codificación en subbandas utiliza un banco de filtros básico que está formado por un par de filtros, un filtro paso bajas y un filtro paso altas. Cuando se definió un banco de filtros, se mencionó, que una señal de entrada es filtrada tanto por el filtro paso bajas, como por el paso altas, a estos filtros en el banco se les conoce como filtros de análisis [3].

Este proceso tiene dos etapas básicas, la etapa de análisis (también conocido como banco de análisis) y la etapa de síntesis (banco de síntesis). El banco de análisis contiene los filtros de análisis y son los encargados de separar a la señal de entrada en bandas de frecuencias, con el objetivo de que la señal filtrada pueda ser comprimida más eficientemente que la señal original. [30].

Al salir la señal del banco de análisis no es necesario que se preserve la señal tal cual, normalmente se realiza un proceso que se le conoce como *downsampling*, en donde se utilizan sólo las componentes pares de la señal a la salida de los filtros paso bajas y paso altas [30], proceso que también se le llama decimación. El hecho de poder realizar lo anterior es porque los filtros tienen una propiedad conocida como "reconstrucción perfecta" [1], estos filtros son también conocidos como QMF (Quadrature Mirror Filter). Después de la decimación, la señal puede ser codificada, por ejemplo, para su transmisión. Cuando sea requerida la señal se efectuará el proceso de reconstrucción.

La etapa de síntesis o de reconstrucción de la señal, empieza por realizar un proceso conocido como *upsampling*, que se refiere a insertar ceros en las componentes impares de la señal, después de esto la señal es filtrada por unos filtros que guardan cierta relación con los utilizados en la etapa de análisis, estos filtros son conocidos como filtros síntesis [3] y que por la misma característica que los anteriores permitirán la reconstrucción perfecta de la señal original.

En la figura 2.21 se muestra un diagrama de la codificación en subbandas.

El banco análisis se encuentra a la izquierda, éste tiene un filtro paso bajas $h(n)$ y un filtro paso altas $g(n)$, estos van acompañados del proceso de "*downsampling*" ($\downarrow 2$) (o decimación). El banco síntesis de la derecha empieza por hacer el proceso

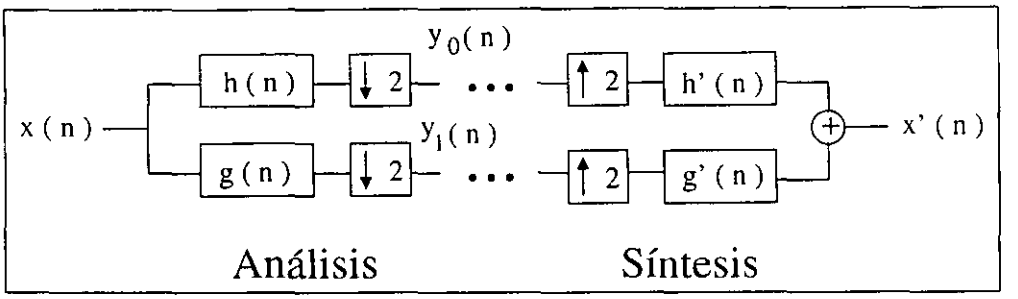

Figura 2.21: Esquema de un banco de filtros

de "*upsampling*" ($\uparrow 2$) , a las señales $y_0(n)$ e $y_1(n)$, después la señal es filtrada de nuevo por los filtros $g'(n)$ y $h'(n)$, así sumando las señales filtradas se obtendrá la señal reconstruida [30].

Como se mencionó anteriormente, la parte importante de los bancos de filtros es la elección de los filtros que tengan la propiedad de reconstruir perfectamente la señal. Estos filtros son los llamados filtros con cuadratura espejo (QMF) y corresponden a filtros con respuesta al impulso finita (FIR). La teoría de estos filtros muestra que las subbandas (señales de salida de los filtros) correspondientes al análisis y síntesis del banco de filtros, corresponden a una descomposición sobre un conjunto de bases ortonormales, seguida en la etapa de reconstrucción por la suma de sus proyecciones ortogonales [1].

Después de haber descrito estos conceptos de manera general, se describirá a continuación como se realiza la transformada wavelet discreta.

2.3.9 Transformada Wavelet Discreta

En [18] se define un algoritmo para calcular la transformada wavelet discreta. En este algoritmo se utiliza una codificación por subbandas con dos bandas (formado por un filtro paso bajas y un filtro paso altas) con ciertas modificaciones que producen la DWT.

Para realizar la transformada directa con este algoritmo, después de la primera etapa de análisis de la codificación en subbandas, la señal de salida del filtro paso bajas puede ser otra vez sujeta a una etapa de análisis, observando la figura 2.21 la señal de salida del filtro paso bajas $y_0(n)$ se convierte en la entrada de otra etapa de análisis. Este mismo proceso puede otra vez ser realizado para la señal de salida del filtro paso bajas, de esta manera este proceso puede ser continuado hasta incluso obtener un punto en la señal de salida del filtro paso bajas (como puede ser el caso del algoritmo de la pirámide laplaciana). En la práctica esto no es necesario y con realizar el proceso de 2 a 5 veces es suficiente. Posteriormente se mencionará en que

beneficia o afecta el realizar el proceso un cierto número de veces.

En la figura 2.22 se muestra un diagrama de descomposición de la transformada wavelet discreta directa para una señal discreta de una dimensión.

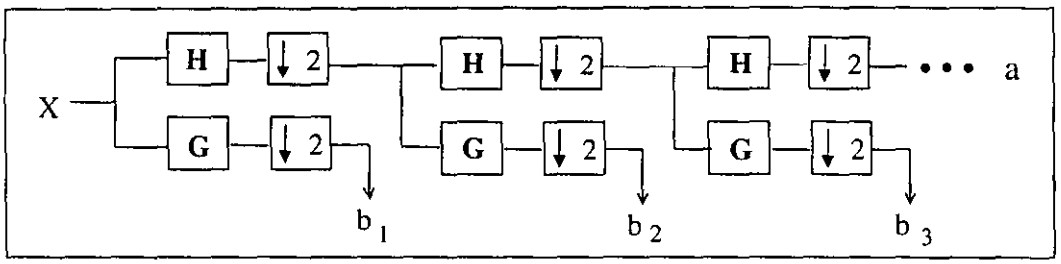

Figura 2.22: Diagrama de la DWT directa de una señal

De acuerdo a la figura 2.22 los coeficientes **a** son los coeficientes obtenidos cuando se realiza la etapa de análisis de manera recursiva sobre los coeficientes de la salida de los filtros paso bajas (también son conocidos como coeficientes promedio), los coeficientes $\mathbf{b_n}$ son las salidas de los filtros paso altas (los cuales también son conocidos como coeficientes detalle). Cabe mencionar que para reconstruir la señal original, es necesario tener todos estos datos de salida.

Para poder ver la conexión de todos los conceptos mencionados anteriormente con la DWT, se puede notar en el diagrama de la figura 2.22 que el filtro **H** de la etapa de análisis de un banco de filtros de una codificación en subbandas y que corresponde a un filtro paso bajas en una transformada wavelet discreta directa, le corresponde una función de escalamiento $\phi(t)$ y al filtro **G**, le corresponde una función wavelet $\psi(t)$. Con esto se puede establecer que la función de escalamiento se asocia con un filtro paso bajas y que una función wavelet se asocia con un filtro paso altas. Entonces en una transformada wavelet discreta directa, una función discreta de entrada X, es convolucionada con las funciones $\phi(t)$ y $\psi(t)$ para obtener los coeficientes promedio y los coeficientes detalle respectivamente. Y que el hecho de realizar el proceso recursivamente con los coeficientes de salida de una etapa se parece al proceso del algoritmo de la pirámide laplaciana, en donde cada vez que se realiza la recursión se reduce en este caso la cantidad de los coeficientes promedio.

La transformada wavelet discreta inversa puede ser realizada invirtiendo el proceso anterior. Este proceso se puede observar de manera más clara en la figura 2.23.

En la figura 2.23 se puede notar que los filtros utilizados en la DWT inversa no son los mismos que los utilizados en la DWT directa, de hecho la barra arriba de la letra **H** y **G** no indican el conjugado, sino que son filtros diferentes formados a partir de los utilizados en la transformación directa. La relación que existe entre éstos es

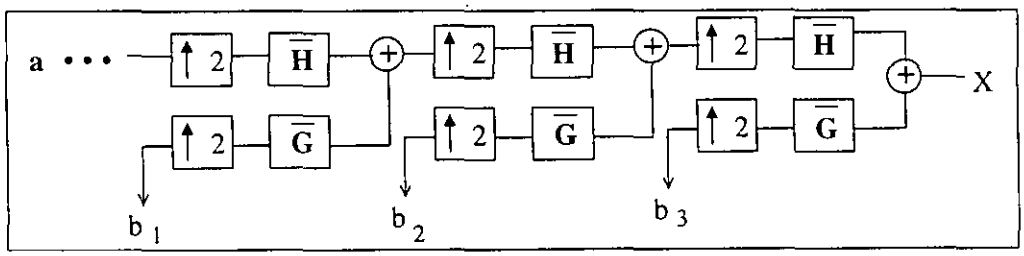

Figura 2.23: Diagrama de la DWT inversa

mostrado a más detalle en [2].

Lo importante en esta teoría, tal y como se ha enfatizado anteriormente, es saber qué función de escalamiento y wavelet utilizar. Se ha mencionado que tienen que ser funciones que deben garantizar ciertas propiedades y tener ciertas características. Estas propiedades y características están muy ligadas con las que deberían tener los filtros en un banco de filtros (QMF), para garantizar la perfecta reconstrucción de la señal original o de entrada. Aparte de esto, también es importante notar como la teoría del análisis en multiresolución hace la conexión entre funciones de escalamiento y las funciones wavelet. La justificación de la utilización de las funciones de escalamiento y wavelet en este proyecto de investigación, está fuera del alcance de esto texto, pero en la mayoría de los referencias citadas, está contenido el porque de la utilización de esas funciones.

Hasta el momento se ha mostrado el análisis con funciones de una dimensión en donde solo está involucrada una variable independiente, pero prácticamente toda la teoría mostrada puede ser aplicada a funciones de dos dimensiones (como las imágenes), esto se puede realizar aumentando otra variable independiente al análisis y en muchos casos basta con sustituir la función unidimensional por otra bidimensional. En el caso por ejemplo, de la transformada wavelet discreta, puede ser aprovechada la propiedad de separabilidad del núcleo en una transformación, es decir, para una transformada wavelet discreta puede ser utilizada la teoría anterior para los renglones de una imagen y de la misma forma para las columnas, haciendo así el análisis en toda la imagen (o función de dos dimensiones).

Para realizar una transformada wavelet discreta directa en una imagen, es necesario tomar en cuenta la propiedad de separabilidad del núcleo en una transformación lineal[18]. De esta forma la DWT puede ser realizada de acuerdo al diagrama siguiente (figura 2.24)

El diagrama anterior muestra como una imagen tiene que ser convolucionada primero por renglones, con las funciones de escalamiento y wavelet, **H** y **G** respectivamente, después del bloque de decimación, las salidas tendrán que volver a convolu-

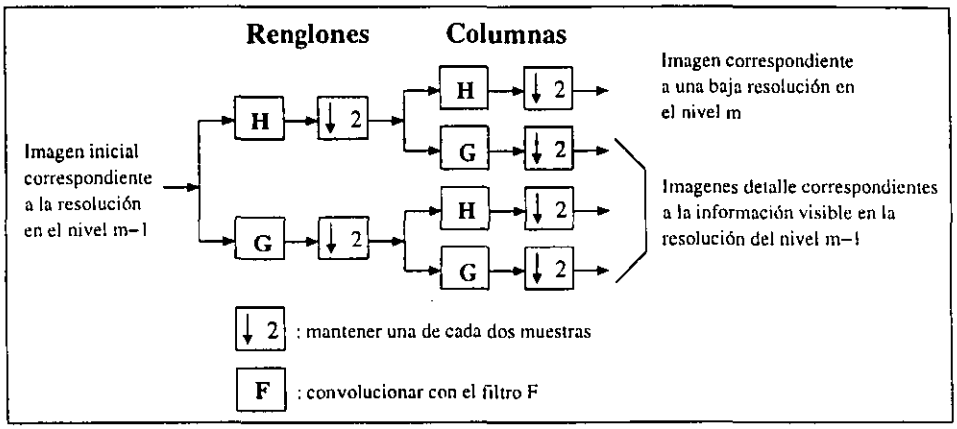

Figura 2.24: Diagrama de la DWT directa para una imagen

cionarse con los filtros **H** y **G** pero ahora por columnas. En el diagrama se indica que la imagen de entrada, se encuentra en la resolución en el nivel $m-1$ y que la imagen que resulta de aplicar el proceso de análisis por renglones y por columnas, corresponde a una imagen en la resolución en el nivel m. Esto quiere decir que, si se aplica este proceso una sola vez, (ya se mencionó que podía ser aplicado recursivamente), la imagen de salida estará en la resolución en el nivel $m = 1$, por tanto la imagen original se encontrará en el nivel $m = 0$. De hecho la imagen de entrada u original, siempre se encuentra en el nivel $m = 0$ y la última imagen obtenida al ser aplicado el proceso n veces, estará en el nivel $m = n$, por ejemplo, si el proceso se aplicó tres veces, entonces la última imagen, que es producida por la salida de los filtros **H**, se encuentra en el nivel $m = 3$, por tanto la anterior en el nivel $m = 2$ y así sucesivamente.

Para el caso de la resolución, se puede observar que si a una imagen de tamaño $N \times N$, a la que se le aplica el algoritmo de la transformada wavelet discreta una sola vez, su imagen de salida de los filtros **H** es de tamaño $N/2$ x $N/2$, es decir, su resolución se disminuyó en un factor de 2, producto del proceso de downsampling o decimación. De esta forma las imágenes subsecuentes en la transformada wavelet, tendrán cada vez que se realice el proceso recursivamente, la resolución más y más baja.

También en el diagrama anterior (figura 2.24), las tres imágenes restantes denominadas imágenes detalle (de igual forma también se le pueden llamar coeficientes detalle) son disminuidas en su resolución y nivel m. De igual forma se nota en el diagrama que la imagen original es convolucionada primeramente por renglones con dos filtros diferentes y después se convolucionan por columnas combinando los dos filtros anteriores. Por ejemplo, cuando el proceso se realiza solo una vez ($m = 1$),

se observa que la imagen original es filtrada paso bajas–paso bajas (HH)[5], paso bajas–paso altas (HG), paso altas–paso bajas (GH) y paso altas–paso altas (GG), a las cuales algunos autores les llaman: subimagen con baja resolución, subimagen con orientación horizontal, subimagen con orientación vertical y subimagen con orientación diagonal respectivamente. En la figura 2.25 se muestra de manera gráfica.

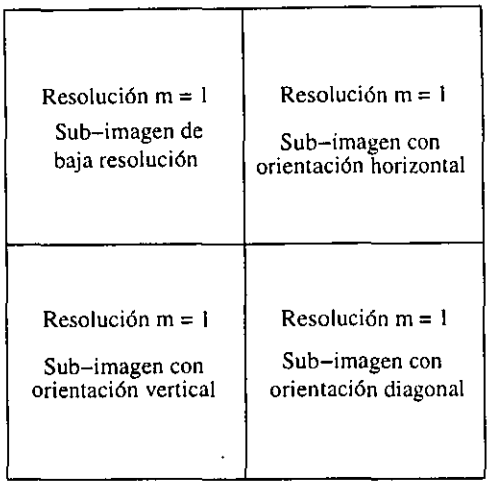

Figura 2.25: Esquema de una imagen de salida de una DWT, cuando m = 1

Como ejemplo, se puede observar como la imagen de entrada (figura 2.26), es afectada por la DWT cuando $m = 1$ (figura 2.27).

Como el proceso de la DWT es recursivo, la figura 2.28 muestra como se producen las subimágenes cuando m es mayor a 1.

Para reconstruir la imagen original se procederá de acuerdo al siguiente algoritmo, mostrado en el diagrama de la figura 2.29.

El diagrama indica que si se realiza este proceso con las subimágenes obtenidas en la etapa de análisis o DWT directa, al final se podrá reconstruir la imagen original.

En el proceso de transformación de la imagen original (DWT), es importante considerar qué funciones de escalamiento y wavelets se tendrán que utilizar ya que como se mencionó al inicio del tema, existe gran diversidad de funciones wavelets y el uso de una u otra podrá afectar en la reconstrucción de la señal original. También se deben considerar aspectos en la convolución de las señales ya que esto puede acarrear cierto error en la etapa de transformación, que será reflejado al momento de reconstruir la señal. Las funciones wavelets que se usan en este proyecto de investigación

[5]En este caso H y G son los filtros.

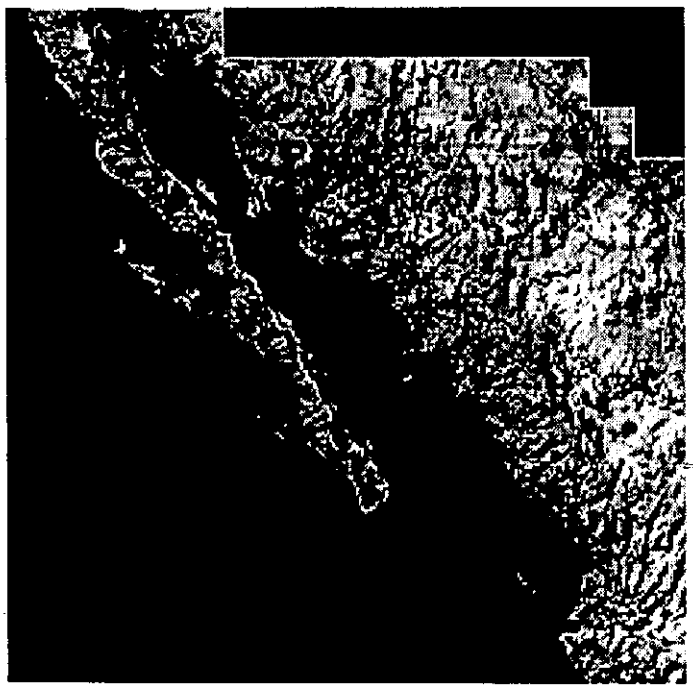

Figura 2.26: Imagen original

son conocidas como la **wavelet biortogonal 7-9**, las cuales fueron descubiertas por Ingrid Daubechies en el artículo citado en [2]. En ese artículo se menciona que si se aplica el algoritmo propuesto por Mallat con estas wavelets, la imagen de entrada podrá ser descompuesta con la DWT y reconstruida perfectamente por medio de la IDWT.

En el siguiente capítulo se presenta la teoría que está detrás de una técnica de compresión de datos y en particular en imágenes. Se presentan algunos conceptos que son necesarios conocer para poder entender como funciona la técnica de compresión propuesta para aplicar en imágenes geográficas. Se presentan algunas ventajas y desventajas de utilizar los distintos tipos de técnicas para compactar datos, para al final del capítulo siguiente detallar como se realiza este proceso de compresión.

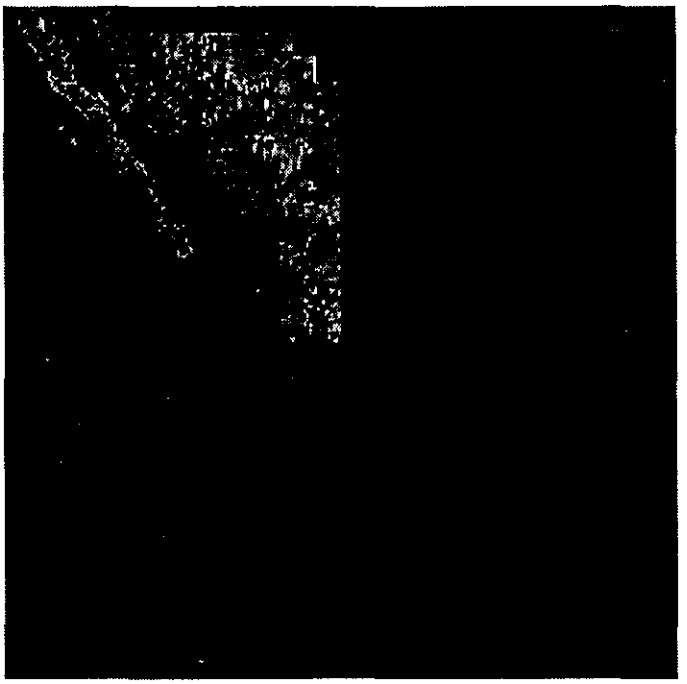

Figura 2.27: Imagen transformada, $m = 1$

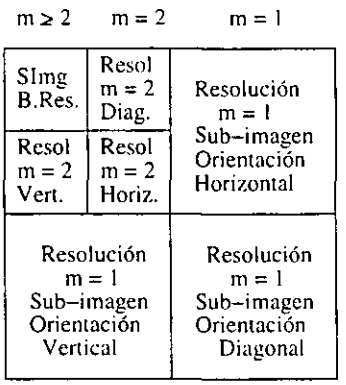

Figura 2.28: Imagen DWT con $m \geq 2$

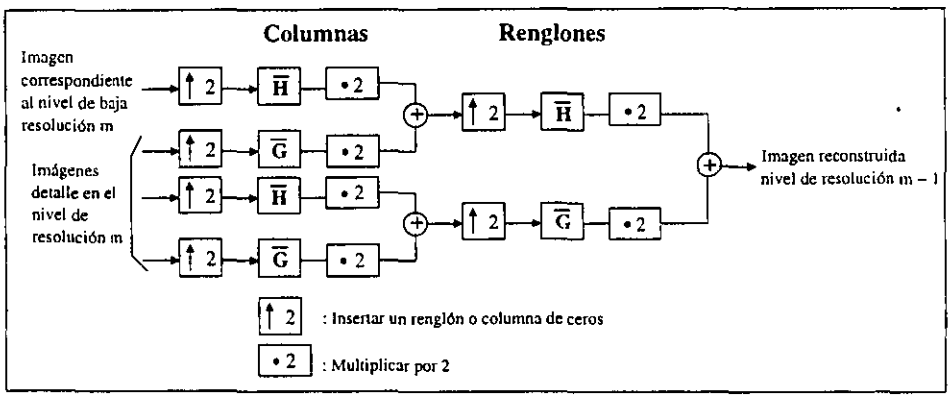

Figura 2.29: Diagrama de la DWT inversa para una imagen

Capítulo 3

Compresión de imágenes

3.1 Introducción

En el capítulo 1 se citaron cuáles son las principales necesidades que acarrea una técnica de compresión de imágenes. Se mencionó que los problemas principales al utilizar imágenes en un sistema son su almacenamiento y transmisión, ya que éstas requieren manejar una gran cantidad de datos. Estos son los problemas primordiales que se tienen que tomar en cuenta al implementar una técnica de compresión.

Por su naturaleza, las imágenes digitales requieren gran cantidad de datos para representarla, estos datos muchas veces contienen información redundante o innecesaria que puede ser eliminada de una imagen, logrando así reducir la cantidad de datos necesarios para representar a la imagen.

La teoría que permite analizar el contenido de información de una cantidad de datos, es llamada teoría de información. Esta teoría describe la posibilidad de representar determinada cantidad de información de una forma compacta, es decir, analiza el contenido de la información de una cantidad de datos, y establece cual es la cantidad mínima, que puede ser usada para representar esa información. De esta forma, esta teoría indica que basta retener esta cantidad mínima de datos, la restante puede ser quitada, de tal manera que el contenido de información no sufra pérdidas. A la cantidad de datos que pueden ser quitados se le conoce como información redundante y es una característica que tienen la mayoría de los datos que transportan información.

Esta teoría es muy importante en el área de la compresión de datos, ya que indica cual es el límite mínimo para poder representar la información con determinada cantidad de datos. Por ejemplo, si se codifica un mensaje de 10 letras, en donde a cada letra de la palabra se le asignan 8 bits, entonces esta teoría, establece la cantidad mínima de bits para representar ese mensaje (lo cual en muchas ocasiones es menor a los 8 bits usados para codificar cada letra). Lo importante es, que cuando

se realice el proceso de decodificación de ese mensaje, se reconstruya tal y como se tenía originalmente, es decir, sin que exista pérdida de información alguna.

La teoría de compresión de imágenes esta enfocada principalmente a tratar de reducir la información redundante en una imagen [5].

Este capítulo comenzará comentando las técnicas de compresión conocidas como "sin pérdidas" o "sin errores" (*lossless*), posteriormente se hará mención de las técnicas de compresión llamadas "con pérdidas" o "con errores" (*lossy*), se mencionarán ventajas y desventajas de usar estas dos técnicas, para finalmente detallar como se realiza la técnica de compresión que se usa en este proyecto.

3.2 Técnicas de compresión sin pérdidas

Como se mencionó en el capítulo 1 las técnicas de compresión eliminan información redundante en una imagen. De esta manera, las técnicas de compresión sin pérdidas recuperarán exactamente a la imagen, después de aplicarle el proceso de "descompresión" [5].

Este tipo de técnicas se pueden dividir básicamente en dos categorías: técnicas basadas en "diccionarios" (*Dictionary-based techniques*) y técnicas por métodos estadísticos. Las técnicas basadas en diccionarios utilizan códigos de longitud fija para comprimir sus archivos, los cuales representan una secuencia de bytes del archivo original.

Los métodos estadísticos en cambio, utilizan la frecuencia de ocurrencia de caracteres de un archivo para comprimir sus datos. La idea es codificar los caracteres de mayor frecuencia con muy pocos bits, mientras que los que tienen menor ocurrencia se les asignan más bits [5].

A continuación se mencionará una de las técnicas basadas en diccionarios.

3.3 Codificación por longitud de series

Por sus siglas en ingles es comúnmente conocido como RLE (*Run Lenght Encoding*). En una imagen, por ejemplo en escalas de grises, es común ver que cierta región de la imagen contiene el mismo tono o en muchas ocasiones se dice que tiene el mismo nivel de gris, entonces los renglones de esa región (conjunto de pixeles) son llamados series. La codificación consiste entonces, en representar toda la serie por medio de alguna secuencia que indique la existencia de una serie de cierto tamaño formada con un tono de gris, evitando así, codificar cada uno de los pixeles de esa región por separado y de la misma forma, es decir, con el mismo valor. Otra forma de ver esta codificación es teniendo un conjunto de 0's y 1's que se generaron a partir

de alguna fuente. Entonces toda secuencia de 0's entre dos 1's puede ser codificada como una serie, al igual que cualquier secuencia de 1's entre dos 0's [15].

Por ejemplo:

La serie de números:

$$\ldots 10000000000000000000000001\ldots$$

puede ser representada como:

$$\ldots 1[25-0]1\ldots$$

en donde 25-0 índica una serie de 25 ceros, los cuales estarán codificados de alguna forma. De igual forma para la serie:

$$\ldots 01111111111111111111111110\ldots$$

puede ser representada como:

$$\ldots 0[25-1]0\ldots$$

Este tipo de codificación logra compresión considerable, cuando en imágenes se encuentra un fondo del mismo tono o color. En cambio cuando las imágenes tienen muchas variaciones en tonos la compresión con esta codificación es muy pobre [5].

Algunos formatos de archivos de imágenes que utilizan la idea del RLE son el Targa, PCX y TIFF.

3.3.1 Codificación Huffman

Este tipo de codificación se basa en métodos estadísticos. Se emplea básicamente para reducir la longitud promedio del código usado para representar los símbolos de un alfabeto [11].

Utiliza un árbol de codificación binario para representar los valores que ocurren con mayor frecuencia (mayor probabilidad de ocurrencia) en una secuencia de caracteres, con la menor cantidad de bits posible, y los que ocurren con menor frecuencia (menor probabilidad) se le asignan más bits. Si el árbol de codificación utiliza una tabla fija que contiene las probabilidades de ocurrencia de los valores, el método es llamado, codificación Huffman estático, y si ésta es generada durante el proceso de codificación el método es conocido como codificación Huffman dinámico. La ventaja del método dinámico es que puede generar tasas de compresión mas altas que la estática, aunque su desventaja es el tiempo de procesamiento [5].

Este tipo de codificación tiene la característica de tener una propiedad de prefijos, lo cual significa que cada código asociado a un símbolo, contiene una secuencia de símbolos al inicio de su código, diferente a cada símbolo. Esto asegura que el código sea descifrable instantáneamente [11].

El algoritmo de codificación de Huffman realiza los siguientes pasos:

1. Ordena las probabilidades de los símbolos en orden decreciente y los considera como nodos hojas de un árbol (reducción de los símbolos de la fuente).

2. Mientras exista más de un nodo:

   - Unir los dos nodos con las probabilidades más bajas para formar un nuevo nodo, el cual, su probabilidad es la suma de los nodos que se unieron.
   - Asignar arbitrariamente 1 y 0 a cada par de ramas que se unen en un nodo (asignación de código).

3. Leer secuencialmente desde la raíz de un nodo hasta el nodo hoja donde sea localizado el símbolo.

El algoritmo anterior produce la tabla de codificación Huffman ("Huffman code book") para cualquier conjunto dado de probabilidades. La codificación y decodificación son realizadas revisando los valores de la tabla [15].

Este tipo de codificación, llamado también codificador de entropía, no es práctico para codificar imágenes en formato crudo (raw), es decir, imágenes a las cuales no se le ha aplicado ningún tipo de preprocesamiento (como una transformación). Sin embargo, es muy utilizado cuando se realiza un tipo de preprocesamiento a los datos originales de la imagen, como en el caso de los algoritmos de compresión por medio de una transformación [15].

Una de las ventajas de este codificador es que consigue el número más pequeño posible de símbolos (bits) por cada símbolo que se produce en una fuente[1]. Esto es porque su desarrollo se basó en la teoría de la información, que como se mencionó en la introducción de este capítulo, indica cual es la mínima cantidad de bits que pueden ser asignados a un conjunto de estos y que pueden representar caracteres o valores de pixeles. Debido a esto, este tipo de codificación es muy utilizado en muchos codificadores de datos y también en técnicas de compresión de imágenes, tal es el caso del estándar de compresión de imágenes actual llamado JPEG[2][23].

Por tanto, la técnica de compresión de imágenes de este proyecto de investigación, utiliza este tipo de codificación. Más adelante en el texto se podrá notar en que etapa del proceso de compresión se ocupa este codificador.

En resumen, la ventaja de las técnicas de compresión sin errores es que permiten regenerar o reconstruir los datos originales sin que estos sufran algún cambio, la desventaja es que estos esquemas ofrecen tasas de compresión comúnmente de 2:1 y en muy pocos casos hasta 4:1, lo cual equivale, a reducir el tamaño de un archivo a

[1]Una fuente puede producir cualquier secuencia de bits, por ejemplo en el caso de un caracter, este conjunto es igual a 8 bits.

[2]Hasta octubre de 2000, todavía este es el estándar, a partir de los primeros meses de 2001, lo reeplazará el nuevo estándar JPEG2000.

la mitad (2:1) o en el mejor de los casos cuatro veces (4:1) [9].

En el caso de algunas imágenes, tiene que enfatizarse, desde antes de utilizar una técnica de compresión, si es necesario preservar los datos originales. Por ejemplo, cuando se tienen imágenes médicas, es muy importante conservar los datos originales, ya que cualquier pérdida en los datos puede ocasionar un diagnóstico erróneo. En otro tipo de aplicaciones puede llegar a aceptarse cierto grado de error en la imagen. En el caso de la imágenes geográficas, no es muy importante reconstruir la imagen original, sino que puede llegar a aceptarse una buena aproximación a la original, en cuanto a calidad perceptual de la imagen se refiere. Esto es, si se pudiera comparar la calidad visual de la imagen original y la que se reconstruyó con cierto grado de error, se notaría que las imágenes son prácticamente idénticas. Este es el aspecto que las técnicas de compresión con pérdidas toman ventaja, ya que el ojo humano no alcanza a percibir esa pérdida de información que sufrió la imagen original.

3.4 Técnicas de compresión con pérdidas

En los esquemas de codificación con pérdidas, se permite que las imágenes reconstruidas tengan cierta degradación en sus datos con respecto a las originales. Estas degradaciones pueden o no ser percibidas visualmente, pero se acrecentarán cuando se aumente la tasa de compresión. Un esquema típico de las técnicas de codificación con pérdidas se ilustra en la figura 3.1.

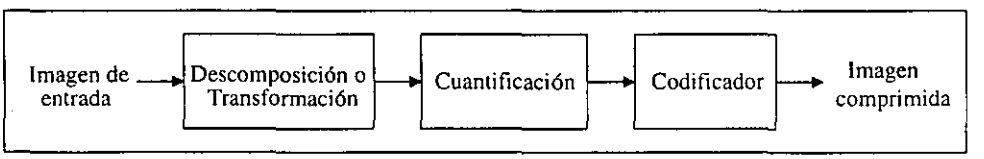

Figura 3.1: Esquema típico de un codificador con pérdidas.

En la figura 3.1 se muestra un diagrama de bloques con tres componentes fundamentales: la descomposición o transformación de la imagen, la cuantificación y un codificador de símbolos. La importancia de uno u otro dependerá de la técnica especifica que se utilice, por ejemplo, si la transformación elegida es la de Fourier o la coseno discreta, va a existir un método de cuantificación que se adapte de mejor manera a ese tipo de transformación. En el caso de la transformada coseno discreta se ha experimentado un método de cuantificación en zig-zag que logra conservar los coeficientes de la transformada con mayor cantidad de información. De cualquier manera y como regla general, el esquema más sofisticado será aquel que logre la mejor calidad de una imagen de acuerdo a una tasa de bit fijada [25].

Existen muchas técnicas de compresión con pérdidas tales como, PCM (codificación por modulación de pulsos) y sus variantes DPCM y ADPCM, codificación

predictiva, codificación por transformación, codificación por subbandas y codificación con fractales. En este capítulo solo se mencionarán aspectos importantes de la técnica de codificación por transformación y en subbandas, ya que esto ayudará a entender como funciona la técnica de compresión de este proyecto.

3.4.1 Codificación por transformación

En este esquema de codificación se utiliza regularmente una transformada lineal reversible, para hacer corresponder a la imagen con un conjunto de coeficientes generados por la transformada. Estos coeficientes son los que se cuantifican y codifican. Una de las ventajas al transformar una imagen es que, se genera un número significativo de coeficientes con magnitud pequeña y que se pueden cuantificar de manera poco precisa y en muchas ocasiones pueden llegar a ignorarse, sin que esto afecte o degrade de manera significativa la calidad de la imagen [10].

El éxito de una transformada es deshacer la correlación de los pixeles de la imagen, en donde comúnmente se dice que la información de la imagen es empaquetada o distribuida en el menor número de coeficientes posible [25].

De acuerdo con la figura 3.1, la etapa de cuantificación entonces, cuantificará con menor precisión los coeficientes que lleven la menor cantidad de información y con mayor precisión aquellos que empaquetan o mantienen mayor cantidad de información de la imagen y que es importante conservarlos para su reconstrucción. El proceso de compresión termina codificando los coeficientes cuantificados por medio de algún código de longitud variable (código Huffman o RLE) [10].

Cuando se realiza una transformación a una imagen de $N \times N$, en muchas ocasiones esta imagen tiene que ser subdividida en bloques de $n \times n$, en donde n comúnmente toma el valor de 8 o 16. Entonces la transformación es aplicada a cada uno de los bloques de la imagen. Cuando una transformación se realiza en una dimensión (1-D), se realiza ya sea sobre las columnas o renglones de la imagen y cuando es una transformación en dos dimensiones (2-D), la transformación se realiza tanto en las columnas como en los renglones. Existen transformadas a las cuales se les dicen que tienen núcleos separables, en donde una transformación en 2-D puede ser descompuesta en dos transformaciones de 1-D, en donde las operaciones horizontales y verticales se hacen por separado. Así una transformada de este tipo, ejecuta una transformada de 1-D sobre los renglones del bloque de $n \times n$ y la otra transformación se realiza sobre las columnas del bloque, en donde al final se obtendrá una transformación en 2-D [25].

Cabe aclarar que, como se mencionó en el capítulo 2, la transformada wavelet discreta no necesita descomponer a la imagen en bloques de $n \times n$, lo cual en cuestión de tiempo de procesamiento, puede ser una ventaja, además de otras desventajas que sufre la imagen al ser descompuesta en bloques, a continuación se especifica cual es

el efecto que esto produce.

La transformación de un bloque de la imagen o subimagen de $n \times n$, permitirá que algunos coeficientes de la imagen tengan la mayor cantidad de información del bloque transformado, lograndose así cuantificar con mayor precisión estos coeficientes y con menor los restantes. Esta última cuantificación puede ocasionar notables cambios en el nivel de gris en los bordes de los bloques. A este efecto se le conoce como bloqueo (*blocking*) y a tasas de bits bajas puede ser muy notable [5].

A continuación se describen algunas de las características que son deseables cuando se realiza una transformación, para comprimir una imagen:

- Empaquetamiento de la energía: una transformada tiende a compactar la energía de una imagen en el menor número de coeficientes posible [9]
- Decorrelación de la imagen: una transformación tiende a decorrelacionar (reducir la dependencia lineal) entre los coeficientes de la imagen, ocasionando que los cuantificadores sean más eficientes [9].
- Rápida implementación: el número de operaciones requeridas para una transformada de 1-D es generalmente del orden de $O(n^2)$. Algunas tienen implementaciones que son más rápidas y reducen el orden a $O(n \log n)$. Para una transformada separable 2-D de un bloque de n x n, ejecutando transformadas de 1-D a renglones y columnas, el número de operaciones se reduce a $O(2n^2 \log n)$ en lugar de $O(n^4)$ [25].

Las transformadas lineales que son consideradas para la codificación de imágenes, pueden ser expresadas como:

$$T_f(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} f(n_1, n_2) a(n_1, n_2; k_1, k_2) \tag{3.1}$$

$$f(n_1, n_2) = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} T_f(k_1, k_2) b(n_1, n_2; k_1, k_2) \tag{3.2}$$

donde $f(n_1, n_2)$ es una imagen de N_1 x N_2 pixeles, $T_f(k_1, k_2)$ es también una imagen de N_1 x N_2 pixeles y representa los coeficientes de la transformada y $a(n_1, n_2; k_1, k_2)$ y $b(n_1, n_2; k_1, k_2)$ son las funciones base que satisfacen la ecuación 3.1. De la ecuación 3.2 se puede establecer que $f(n_1, n_2)$ es una combinación lineal de las funciones base $b(n_1, n_2; k_1, k_2)$ y que los coeficientes transformados $T_f(k_1, k_2)$ son las amplitudes de las funciones base en la combinación lineal. Como se mencionó anteriormente, muchas de las transformadas utilizadas para codificar una imagen son separables y de acuerdo con esto las ecuaciones 3.1 y 3.2 pueden ser expresadas como [17]:

$$T_f(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} f(n_1, n_2) a_R(n_1; k_1) a_C(n_2; k_2) \tag{3.3}$$

$$f(n_1, n_2) = \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} T_f(k_1, k_2) b_R(n_1; k_1) b_C(n_2; k_2) \tag{3.4}$$

En donde el subíndice R es para renglones y C es para columnas.

Un ejemplo de una transformada separable es la transformada discreta de Fourier (DFT), en donde las ecuaciones anteriores pueden ser expresadas como:

$$F(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} f(n_1, n_2) e^{-j(2\pi/N_1)k_1 n_1} e^{-j(2\pi/N_2)k_2 n_2} \tag{3.5}$$

$$f(n_1, n_2) = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} F(k_1, k_2) e^{j(2\pi/N_1)k_1 n_1} e^{j(2\pi/N_2)k_2 n_2} \tag{3.6}$$

Se han considerado muchas transformadas diferentes en los esquemas de codificación por transformación. Tales transformadas difieren en el grado de compactación de energía y muchas veces en requerimientos de cómputo, es decir, muchas transformadas son muy buenas pero su proceso de computo es muy largo. Tal es el caso de la transformada Karhunen-Loeve (KLT), ésta es considerada como la mejor transformación lineal para compactación de energía, pero es muy impráctica ya que necesita de mucho tiempo de cómputo [17].

Otras transformadas usadas comúnmente en este esquema son la transformada de Fourier discreta (DFT), la transformada coseno discreta (DCT), la transformada Walsh-Hadamard (WHT) y actualmente la Transformada Wavelet Discreta (DWT). La transformada DFT es utilizada porque tiene un conjunto de funciones base fijas y su algoritmo es eficiente para su cómputo y buena compactación de energía para imágenes típicas, concentra una parte considerable de la energía de la señal en componentes de baja frecuencia. Esta transformada utiliza la FFT (*Fast Fourier Transform* - transformada rápida de Fourier -) para calcular la DFT directa y la inversa [17]. Una desventaja de esta transformada es que genera coeficientes complejos (consisten de una parte real e imaginaria o de magnitud y fase) y el almacenamiento y manipulación de estos coeficientes a veces causa problemas. Otra desventaja es que genera componentes formados por la periodicidad implícita de un bloque en la imagen, que al ser codificadas a bajas tasas de bits, se produce el efecto de bloqueo (blocking) [25]. La transformada DCT vino a mejorar algunas de las desventajas que ofrecía la DFT. La transformada DCT esta cercanamente relacionada con la DFT y ayuda a mejorar el grado de compactación de energía que proporcionaba la DFT [17]. Los vectores base de la DCT ofrecen una buena aproximación a los de la transformada K-L (KLT), pero con una eficiencia en cómputo muy por arriba de la KLT, por eso es que esta transformada es muy usada, ya que además, forma parte del esquema de compresión del algoritmo JPEG, que es el estándar actual en compresión

de imágenes fijas (hasta octubre de 2000). Esta transformada también disminuye considerablemente el efecto de bloqueo en la imagen, pero no lo elimina del todo. También mejora el grado de compactación de la energía de la señal en comparación con la DFT. Otras transformadas también son usadas pero muy poco, tal es el caso de la transformada WHT, esta transformada no requiere muchos cálculos ni mucho tiempo de cómputo, pero su grado de compactación de la energía no es muy bueno comparado con la DCT [17].

Después de transformar cada bloque de la imagen, un punto importante es determinar o fijar la cantidad de bits que serán asignados a cada coeficiente para su codificación. A este proceso se le conoce como "asignación de bits" (*bit allocation*) [5]. Para realizar este proceso regularmente se utilizan dos criterios, uno que es determinado por la varianza máxima de los coeficientes transformados denominado *codificación por zonas* o el determinado por la máxima magnitud, el cual es denominado *codificación por umbral* [10]. Este proceso está relacionado directamente con el proceso de cuantificación, el cual se describirá mas adelante, haciendo más énfasis en la cuantificación por medio de vectores (cuantificación vectorial) que es el método utilizado por la técnica de compresión para imágenes geográficas.

Las ventajas de utilizar este esquema de compresión, es que se obtiene una tasa de compresión mayor que los esquemas de codificación sin pérdidas, regularmente estas tasas van desde 10:1 hasta 20:1 y la elección entre uno u otra será reflejado en la calidad visual de la imagen. Una de las ventajas es que mientras más grande sea la tasa de compresión el efecto de bloqueo será más notorio, además de que el error de la imagen original y la descomprimida será más grande, es decir, los valores de los pixeles originales serán cada vez menos aproximados después de descomprimir la imagen.

La generación de subbandas, tal y como se mencionó en el capítulo 2, es otra alternativa para descomponer a la imagen original y obtener los resultados de una transformación. Al usar una transformada wavelet discreta en un esquema de compresión, éste puede ser considerado tanto como un esquema de compresión por transformación o por medio de subbandas (codificación en subbnadas), ya que la DWT, cumple con los requerimientos de una transformación como con los de una descomposición en subbandas. Realmente no es muy importante catalogar el esquema en uno u otro tipo, pero es importante aclarar este punto, ya que en gran parte de las referencias aparece como uno y otro. Si el lector esta interesado en la codificación en subbandas puede consultar [25][26].

Como se mostró en la figura 3.1, después de la transformación de la imagen de entrada se necesita que el resultado de este bloque sea cuantificado. Este proceso es muy importante y es en donde se debe tener más cuidado en los métodos de compresión de imágenes, ya que en esta parte es donde se realizará la compresión de las mismas y no en la etapa de transformación. La etapa de transformación o

descomposición en subbandas solo sirve para preparar a la imagen para su posterior cuantificación. Para la cuantificación también existen varios métodos como la cuantificación escalar, cuantificación autoadaptable, cuantificación Zig-Zag o la cuantificación vectorial. Para comprender como funciona el método de este proyecto de investigación solo se describirá la cuantificación escalar y la vectorial.

3.5 Cuantificación

El proceso de cuantificación puede describirse como un mapeo de un conjunto de valores a otro. Este proceso se parece al método de conversión analógico a digital, ya que lo que se pretende es aproximar un conjunto de valores dentro de un rango a un solo valor o valor único, el cual caracterizará a todos los del rango. Por ejemplo, el rango de valores en el intervalo (0.5, 1.5] pueden ser mapeados a un solo valor, el cual en este caso puede ser 1.0. La cuantificación en este ejemplo consiste en aproximar al entero más cercano todo el conjunto de valores dentro de ese rango.

3.5.1 Cuantificación escalar

La cuantificación escalar Q, de L niveles, es un proceso que esta formado por dos conjuntos. Uno es llamado el conjunto de niveles de decisión, el cual lo constituyen los $L+1$ valores $x_0, x_1, \ldots, x_L$ que dividen al espacio formado por los números reales ($\mathbf{R}$). El otro conjunto es el llamado conjunto de valores de reproducción, y que se refieren a valores de salida $y_0, \ldots, y_{L-1}$ tal que:

$$Q(x) = y_i \qquad i = 0, 1, \ldots, L-1 \tag{3.7}$$

$x \in R_i donde R_i = (x_i, x_{i+1})$ y debe satisfacer que:

$$\bigcup_{i=0}^{L-1} R_i = \mathbf{R} \qquad \text{y} \qquad R_i \cap R_j = \emptyset \quad \text{para} \quad i \neq j \tag{3.8}$$

Por ejemplo, un cuantificador de 7 niveles, puede ser el que se muestra en la figura 3.2. En este cuantificador $L = 7$ y los intervalos de cuantificación están dados por $[x_i, x_{i+1}]$.

Por ejemplo, para este cuantificador, si una entrada se encuentra entre x_4 y x_5 su correspondiente valor de salida será y_4. Regularmente se espera que el valor de entrada se encuentre en el intervalo entre $[x_0, x_L]$. Si un valor llegara a ser más grande que x_L entonces su valor de salida será y_{L-1}, y si el valor de entrada es más pequeño que x_0 su valor de salida será y_0.

La operación que tiene que realizar un codificador es obtener el índice i del valor representado por y, es decir, para una entrada x:

$$C[x] = i \qquad i \in 0, 1, \ldots, L-1 \tag{3.9}$$

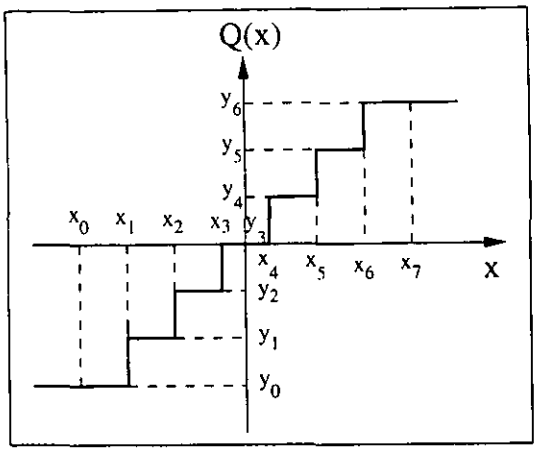

Figura 3.2: Cuantificador escalar de 7 niveles.

donde i es el índice que apunta al intervalo de cuantificación.

Esto quiere decir que con este tipo de cuantificación lo que importa es almacenar o transmitir este índice, ya que tanto en el codificador como en el decodificador los intervalos de cuantificación deben ser los mismos, por tanto, la operación que tiene que realizar el decodificador es:

$$D[i] = y_i \qquad i = 0, 1, \ldots, L-1 \tag{3.10}$$

donde el índice i, corresponde al valor de representación [26].

El error de cuantificación o ruido de cuantificación se define como la diferencia entre el valor de entrada y el valor representado dentro del intervalo $[x_0, x_L]$, en muchas ocasiones también se dice que es una medida de distorsión [3].

Si el cuantificador utilizado tiene intervalos de cuantificación iguales o equidistantes, se dice que es un cuantificador uniforme, sino los tiene, entonces se dice que es un cuantificador no uniforme. Uno de los problemas en la cuantificación es, decidir la distancia de los intervalos de cuantificación para hacerlo óptimo, según la función de densidad de probabilidad (PDF) de la variable x [25].

Si se utiliza el "error medio cuadrático" (MSE) como criterio para reducir el error de cuantificación, el cuantificador más utilizado en estos casos es el llamado cuantificador Lloyd-Max [25].

El cuantificador Lloyd–Max es descrito con más detalle en [9].

La optimización realizada por este cuantificador es producida disminuyendo la distorsión del cuantificador, dado un número fijo de L niveles de representación. Las

ecuaciones del cuantificador Lloyd–Max simplemente indican que los niveles de decisión deberían ser el punto medio entre los niveles de representación vecinos y que los niveles de representación óptimos, son los centroides[3] de la función de densidad de probabilidad en el intervalo apropiado [26].

A continuación se describe la cuantificación vectorial, en donde se podrá notar que es un caso especial de la cuantificación escalar y que de igual forma el problema principal es determinar el conjunto de niveles de representación, el cual se conoce como "codebook".

3.5.2 Cuantificación vectorial

La cuantificación vectorial se refiere a cuantificar no solamente una sola muestra de la señal, sino a un conjunto de éstas. En el caso de una imagen un vector, puede ser el conjunto de valores RGB, que se usan para determinar el valor de un pixel en una imagen a color, o puede ser un conjunto de pixeles, en donde también en muchas ocasiones es llamado bloque. La figura 3.3 ejemplifica esta idea.

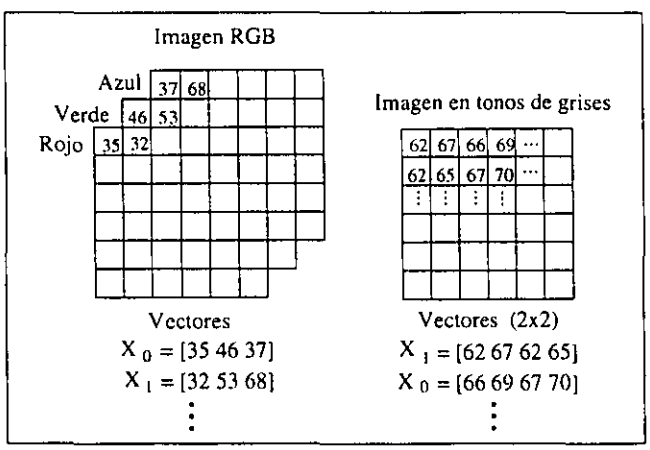

Figura 3.3: Descomposición de una imagen en vectores.

El método de cuantificación vectorial consiste primero en descomponer la imagen a cuantificar en vectores X de cierta dimensión. Cada uno de estos vectores es entonces comparado con el conjunto de vectores representativos $\hat{X}$ para $i = 0, 1, \ldots, N_C - 1$ que forman un "codebook", estos vectores deben ser de la misma dimensión que los formados en la imagen. De acuerdo con la medida de distorsión utilizada, el vector elegido será aquel que tenga la menor distorsión cuando se compara

[3] En este caso, un centroide es considerado como el punto medio entre los dos puntos más distantes de una región, también es conocido como el centro de masa de la región

con todos los vectores dentro del "codebook", esto es:

$$d(\mathbf{X}, \hat{\mathbf{X}}) \leq d(\mathbf{X}, \hat{\mathbf{X}}_j) \quad \text{para toda} \quad j = 0, 1, \ldots, N_c - 1 \tag{3.11}$$

La medida de distorsión utilizada comúnmente es el error medio cuadrático, como en el caso escalar y esto es debido a su simplicidad de cómputo.

La ventaja de la cuantificación vectorial se da cuando los vectores de una imagen son formados por bloques de $n \times m$, por ejemplo 2×2 pixeles, entonces en lugar de cuantificar cada uno de los elementos del vector por separado lo que se hace es comparar cada vector de la imagen con los vectores del "codebook", de esta forma se utilizará solo el índice del vector del "codebook" para cuantificar al vector de la imagen. Por tanto, la operación del cuantificador será obtener cada uno de los índices representativos de los vectores de la imagen y transmitirlos o almacenarlos, mientras que la operación del decuantificador o del proceso de cuantificación inverso, será reconstruir los vectores de la imagen a partir de estos índices, con la ayuda del "codebook". Aquí se puede notar que el mismo "codebook" que se utilizó para la cuantificación, es necesario para la reconstrucción de la imagen. En la figura 3.4 se muestra un esquema o diagrama a bloques de un cuantificador y un decuantificador de una imagen.

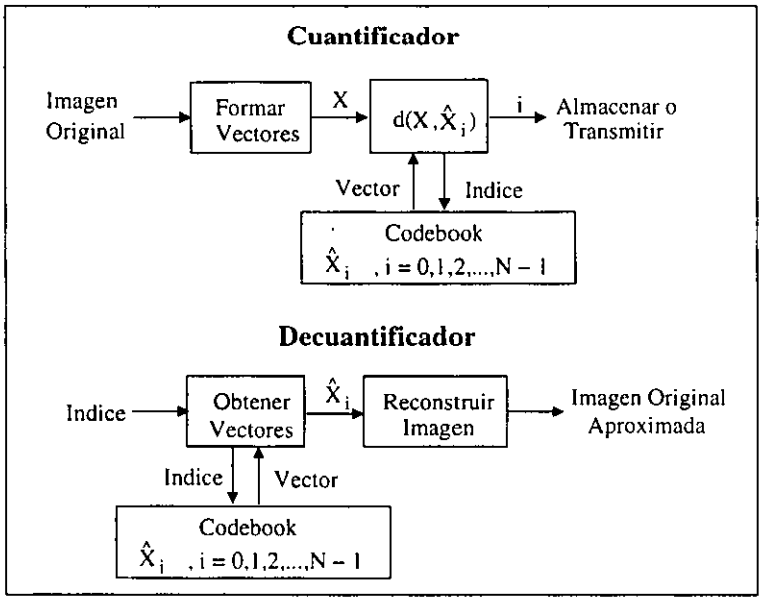

Figura 3.4: Diagrama a bloques de la cuantificación vectorial.

En este cuantificador, solo es necesario almacenar o transmitir $\log_2 N - 1$ bits por cada índice que representa a un vector en la imagen. Por tanto la tasa de bits de un esquema de cuantificación vectorial es $R = (\log_2 N - 1)/n$ bits/pixel.

El aspecto más importante en este tipo de cuantificación es determinar el conjunto de vectores representativos óptimos para cuantificar una imagen ("codebook"), ya que de esto dependerá la buena calidad del cuantificador. Gran parte de la teoría de la cuantificación vectorial está enfocada a determinar la calidad o el rendimiento del cuantificador y esto se refiere a tener un método para generar la cantidad de vectores óptimos, según la distorsión o tasa de bits deseada. No es posible tener un cuantificador con estos dos últimos parámetros de forma ideal, es decir, que los dos sean muy pequeños, ya que si se tiene una medida de distorsión promedio se puede obtener una tasa de bits baja y si se tiene una tasa de bits promedio es posible obtener la distorsión mas baja, pero regularmente siempre hay que sacrificar una cosa para obtener la otra [25].

El algoritmo usado comúnmente para generar el "codebook" es el llamado LBG, donde los vectores para la codificación son generados a partir de un conjunto de imágenes llamadas de entrenamiento, en donde a partir de éstas y por medio de una medida de distorsión establecida (MSE) se obtienen estos vectores. Estas imágenes de entrenamiento son seleccionadas idealmente, de acuerdo a la forma o características de la imagen a cuantificar, con el propósito de que el conjunto de vectores sea óptimo. Idealmente el conjunto de vectores para cuantificar una imagen son los mismos que se generan en la imagen, y si se considera esto, lo que se obtendrá será un codebook local. Esto resulta ser muy impráctico, porque se tendría que generar un codebook por cada imagen a cuantificar, además de que este codebook tendrá que ser almacenado o transmitido junto con los datos representativos de la imagen. Debido a esto, no es necesario que las imágenes de entrenamiento tengan que parecerse a la original, ya que se puede obtener buen "performance" con imágenes con rasgos diferentes, en donde en este caso se generará un "codebook" global [25]. El algoritmo LBG es una generalización del Lloyd-Max usado para la cuantificación escalar. Para más detalle de este algoritmo, se pude consultar en [9].

La forma más sencilla de buscar el vector óptimo a un vector dado es realizando una búsqueda exhaustiva barriendo todo el conjunto de vectores del "codebook". La desventaja de este método radica precisamente en esa característica. El algoritmo LBG tiene la ventaja de que con ese mismo, se pueden generar los "codebooks", a partir de las imágenes de entrenamiento, aunque usando un método de búsqueda exhaustiva puede requerir en ocasiones demasiado tiempo de cómputo, el cual dependerá del tamaño de las imágenes de entrenamiento y de los parámetros de las medidas de error. De acuerdo a lo anterior, se han hecho adaptaciones a ese algoritmo para no usar el método de búsqueda exhaustiva en la generación de los vectores óptimos para el codebook, las cuales proponen ciertas consideraciones a tomar en cuenta para obtener los vectores óptimos con resultados equivalentes al utilizar una búsqueda exhaustiva. Varias de esas adaptaciones pueden ser consultadas en [13].

Una revisión de distintas maneras de usar la cuantificación vectorial puede ser

consultada en [20].

3.6 Técnica de compresión para imágenes geográficas

Dado que ya se describieron las bases teóricas, este tema está dedicado a describir cual es el algoritmo o el procedimiento que se tiene que seguir para comprimir imágenes. Este algoritmo esta basado en el artículo [2]. El diagrama a bloques del sistema de compresión se muestra en la figura 3.5.

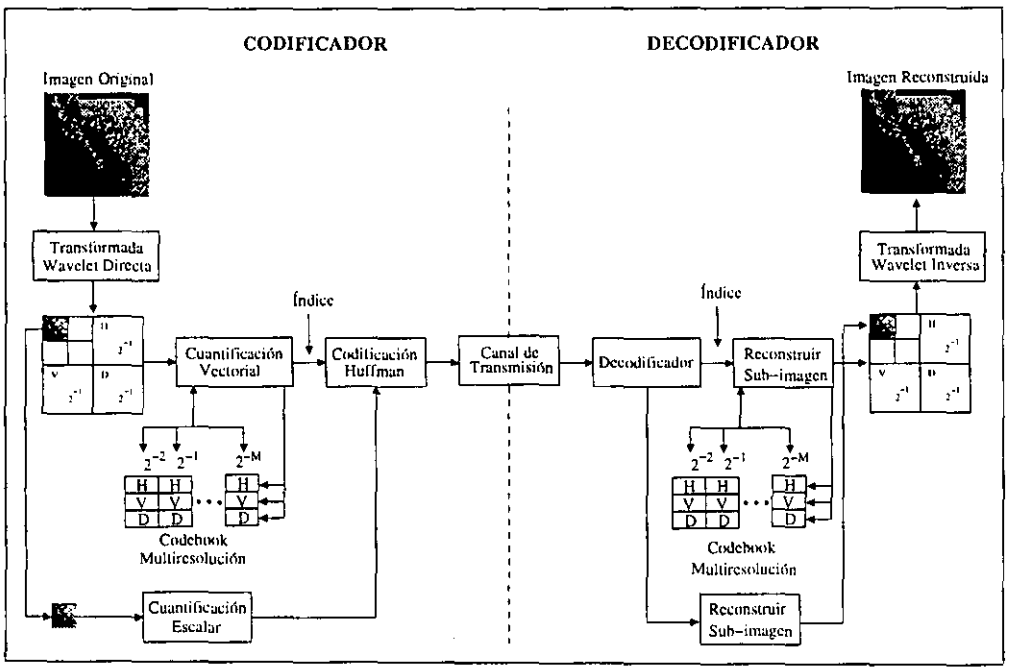

Figura 3.5: Esquema de compresión y descompresión.

Como se observa en la figura 3.5, este sistema, como todos los sistemas de compresión, están formados por un codificador y un decodificador. A continuación se describirá como funciona el bloque de compresión (codificación) y el de descompresión (decodificación).

El codificador de este sistema esta formado por tres partes básicas:

- Bloque de transformación.
- Bloque de cuantificación.
- Bloque de codificación.

A continuación se describe cómo se realiza el proceso de compresión en el sistema.

Como entrada del sistema se tiene una imagen en formato crudo ("*raw*"). A este tipo de formato se le llama así porque presenta los datos de la imagen tal y como se captaron en el proceso de adquisición, es decir, el archivo presenta todos los valores de las tonalidades de cada uno de los pixeles de la imagen, el cual entonces, no presenta ningún tipo de compresión o codificación. Los archivos de este tipo se dice que están en "*ascii*", cuando los valores de los pixeles están expresados explícitamente en el archivo, por ejemplo, si el valor de la tonalidad de un pixel fuera sesenta y dos, dentro del archivo se expresaría este valor como 62 (dos caracteres, un 6 seguido de un 2), en donde cada valor de un pixel estará separado por un espacio en blanco. Este tipo de archivos por consiguiente, ocupa más espacio en un dispositivo de almacenamiento, ya que para cada pixel es necesario ocupar hasta 3 caracteres en caso de ser necesario, más un espacio en blanco entre dos valores de pixeles. Por otro lado este tipo de archivos se dice que está en binario cuando para representar el valor de un pixel se hace uso de una tabla, en donde cada carácter está asociado a un valor entre 0 y 255, regularmente la tabla que se usa es la que representa a todos los caracteres "*ascii*", o también llamada tabla de caracteres ASCII, de esta forma si por ejemplo se quiere representar el valor de sesenta y dos de un pixel, basta solo que en el archivo aparezca el símbolo o carácter ">", el cual dentro de la tabla ASCII representa el valor de 62. El uso de este tipo de archivos reduce de manera considerable el tamaño del archivo en comparación con el denominado archivo "*ascii*", ya que solo será necesario un solo carácter para representar un pixel y los valores entre pixeles no tendrán que estar separados por espacios en blanco.

En particular este sistema acepta tanto archivos de imágenes en ascii y binario, y el formato de imagen que permite manejar ambos casos es llamado PGM (Portable Grey Map).

En la figura 3.6 se muestra una imagen de 128 pixeles por renglón y 128 pixeles por columna con 256 niveles de grises.

Figura 3.6: Imagen de 128 × 128 pixeles, con 256 tonos de grises.

3.6.1 Bloque de transformación

Después de abrir una imagen de tipo "PGM" y ser almacenada en una matriz dentro del sistema, la imagen es primero procesada por el bloque de transformación. Este bloque esta compuesto básicamente por la transformada wavelet discreta (DWT). Como salida de este bloque se tiene una imagen característica de una transformación con wavelets (tal y como se mostró en el capítulo 2), ésta es mostrada en la figura 3.7. Cabe aclarar que lo que se obtiene después de este bloque, es una imagen del mismo tamaño que la original, ya que como se había mencionado, no se obtiene compresión después de transformar una imagen.

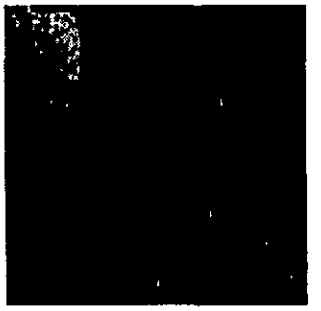

Figura 3.7: Imagen con 2 niveles de transformación.

La señal wavelet usada para este propósito es conocida como **"bi-ortogonal 9–7"** y para efectos de la transformación se utilizó una convolución circular. Cabe señalar que esta señal wavelet es la misma que es utilizada en el estándar de compresión del FBI, con la cual se obtienen resultados experimentales muy favorables para altas tasas de compresión [30].

3.6.2 Bloque de cuantificación

Después de que una imagen es transformada, el siguiente paso es cuantificarla. Como se mencionó anteriormente, este proceso es importante ya que es donde se obtiene la compresión de una imagen, es decir, la técnica usada especificará como tendrá que ser codificado el valor de un pixel, en caso de que la técnica sea escalar, cada valor de un pixel se codificará con su valor correspondiente, y sí la técnica es vectorial, entonces un conjunto de valores de pixeles serán codificados como uno solo.

Este sistema de compresión propuesto utiliza la cuantificación vectorial para cuantificar las subimágenes producidas por la transformada wavelet que tienen orientaciones horizontal, vertical y diagonal, mientras que la subimagen con la característica paso bajas es cuantificada escalarmente. Lo anterior es debido a que la transformada wavelet empaqueta la mayor cantidad de información de la imagen en esa subimagen (paso bajas), para lo cual, es importante preservar estos valores. De

esta forma, se almacenan sus valores sin modificaciones.

El hecho de que la mayoría de las subimágenes de la transformada wavelet se cuantifiquen con vectores, es porque se ha comprobado que cuantificar con vectores mas que escalarmente, aumenta las tasas de compresión cuando existe una tasa alta de asignación de bits en las subimágenes de la transformada, esto es debido a los valores de las estadísticas de estas subimágenes [2]. Esto se puede notar visiblemente en la figura 3.7, donde se observa que es difícil distinguir entre las tonalidades de los pixeles de las subimágenes, ya que la mayoría de estos están cercanos al valor mínimo (cero o de tonalidad negra) o son de una magnitud muy pequeña (con respecto a la subimagen paso bajas). Otra de las ventajas que aprovecha la cuantificación vectorial de una imagen de salida de una transformada wavelet discreta, es que habilita la generación de un codebook en multiresolución ("Multiresolution Codebook"), que es una colección de codebook's pertenecientes a cada una de las subimágenes de la transformada y de esta forma hacer más eficiente el proceso de compresión, ya que estos se pueden generar con subimágenes del mismo nivel de transformación y misma orientación. Cabe aclarar que el conjunto de codebook's usado para cuantificar se tiene que usar también en el proceso de descompresión. La generación de los codebooks y la obtención de los vectores óptimos fue realizado siguiendo el algoritmo LBG, en donde la búsqueda de los vectores fue realizado con un algoritmo propuesto en [13], que tiene la característica de ser equivalente a un algoritmo de búsqueda exhaustiva (Algoritmo I propuesto en [13]).

Un ejemplo de cómo se puede cuantificar una imagen transformada se presenta en la figura 3.8.

Se puede notar en la figura 3.8, cómo la subimagen con la característica paso bajas es cuantificada escalarmente utilizando 8 bits por pixel, es decir, dado que cada nivel de gris va de 0 a 255, se necesitan hasta 8 bits para representar el valor 255 en binario. Posteriormente las tres subimágenes del mismo nivel de transformación son cuantificadas utilizando vectores de dimensión 2 x 2 con 256 vectores pertenecientes al codebook para su codificación, obteniendo así una tasa de 2 bits por pixel. Las subimágenes restantes son cuantificadas con vectores de 4 x 4, obteniendo así una tasa de 0.5 bits por pixel. El hecho de que éstas subimágenes sean cuantificadas con vectores de mayor tamaño que las anteriores es, porque de acuerdo a los estudios realizados en el análisis de la transformada wavelet, éstas subimágenes no contienen demasiada información con respecto a las anteriores y para aumentar la tasa de compresión se pueden cuantificar con vectores más grandes sin llegar a afectar la calidad visual de la imagen, cuando ésta es reconstruida.

Como punto importante en esta parte, los coeficientes de la subimagen paso bajas, debido a las características de la DWT, pueden ser tanto positivos como negativos. Para codificar estos coeficientes sin modificación, es necesario utilizar mínimo 16

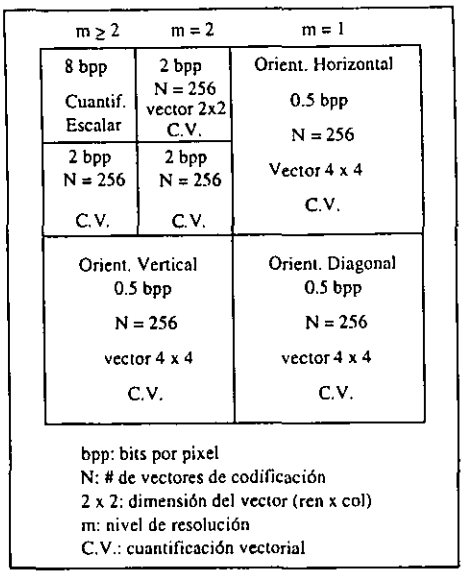

Figura 3.8: Ejemplo de cuantificación de una imagen transformada

bits, ya que hay que conservar el signo. El problema de los signos en esta subimagen, puede hacer que el método de compresión no sea tan eficiente, por el número de bits que se necesitan, para más detalle se puede consultar [29]. Para solucionar este problema, la técnica de compresión hace un análisis del valor de cada uno de los pixeles de esta subimagen, en donde busca el valor positivo y negativo más grande. Entonces, si el valor absoluto del valor negativo más el valor positivo más grande, no es mayor a 255, a todos los valores de esta subimagen se le suma el valor absoluto del negativo, de modo que ahora todos los valores queden entre 0 y 255. De esta forma, ahora si se pueden cuantificar con 8 bits por pixel. Lo importante de esto, es que el valor que se le sumó a los pixeles se tiene que guardar, ya que en el proceso de descompresión, este mismo valor debe ser restado a cada uno de los pixeles.

Continuando con la descripción de la técnica, al final del proceso de cuantificación, lo que se obtiene es una secuencia de números que corresponden a los valores de los índices de la imagen completa. Los valores de estos índices ahora pueden ser codificados mediante algún código de longitud variable, como por medio del código Huffman o el RLE que se mencionaron anteriormente.

Este proyecto en particular, utilizó la codificación Huffman, la cual se describe a continuación.

3.6.3 Bloque de codificación

Cuando se tiene la secuencia final de índices (números enteros), el siguiente paso es codificarlos con un método que reduzca la entropía de la fuente. Para este

propósito, este sistema de compresión utilizará el método de codificación Huffman. Este proceso se realiza siguiendo los pasos que se mencionaron anteriormente.

El método de codificación Huffman es realizado básicamente por dos procesos:

1. Proceso de contracción
2. Proceso de expansión.

Como se mencionó anteriormente, la codificación dinámica obtiene mejores resultados que la codificación estática, por tanto para empezar a aplicar el proceso de contracción, en la codificación Huffman primero es necesario obtener el número de ocurrencias de los valores de los índices que se obtuvieron en el proceso de cuantificación para obtener su probabilidad de ocurrencia. Al final de esto, las probabilidades son ordenadas de mayor a menor.

El proceso de contracción, empieza creando el árbol o tabla binaria, con base en las probabilidades de ocurrencia de los valores.

Después de tener la tabla binaria, el proceso de expansión es el encargado de asignar un código binario (conjunto de 1's y 0's) que represente a cada valor de un índice. Desde luego lo que se pretende es que este código (cantidad de 0's y 1's) que se asigna a cada valor, sea de menor tamaño que el que se tuviere si no se codificara.

Al final se tendrá una serie de símbolos que son producto de la codificación Huffman, los cuales entonces, serán almacenados en un archivo o podrán ser transmitidos por algún canal de comunicaciones.

Es importante mencionar que, si al final del proceso anterior se tuviera un archivo, éste solamente guardaría datos de la imagen original junto con datos originados en el proceso de compresión, como niveles de transformación, dimensión de los vectores o algún otro dato necesario para reconstruir la imagen. Por tanto si se tratara de visualizar el archivo como tal, no se observaría ninguna imagen, ya que para poder observar una imagen, es necesario utilizar este archivo, y con los datos que contiene, realizar el proceso de decodificación.

Particularmente esta técnica genera un archivo con el formato siguiente:

1. Se incluyen datos necesarios para la descompresión del archivo como: número de pixeles por renglón y por columna, niveles de gris, niveles de la transformada wavelet, número de índices producidos en la cuantificación vectorial y el valor que será restado a la subimagen paso bajas. Todos estos datos anteriores se almacenan en formato ascii.
2. Enseguida son almacenados en formato binario los valores de los pixeles de la subimagen con la característica paso bajas producida por la DWT.

3. Y por último, también en formato binario, son almacenados los símbolos producidos por la codificación Huffman.

El decodificador o descompresor está formado básicamente por los mismos bloques del proceso de codificación, solo que estos están ordenados de manera inversa, es decir, para decodificar una imagen, se realizará la contraparte a cada bloque que forma el codificador o compresor. Esto quiere decir que cada proceso realizado anteriormente, necesariamente debe tener un mecanismo que regenere o reconstruya los datos utilizados como entrada de cada bloque. Así , el decodificador estará formado por los siguientes bloques:

- Bloque de decodificación
- Bloque de decuantificación
- Bloque de transformación inversa

En la figura 3.5 se pueden observar claramente estos bloques.

3.6.4 Bloque de decodificación

Los símbolos finales del proceso de codificación fueron generados por el método de codificación Huffman, este bloque toma esos datos como entrada y reconstruye o regenera la secuencia de índices que serán usados en la decuantificación.

3.6.5 Bloque de decuantificación

Este proceso se realiza en dos partes. Como en el conjunto final de índices, producto de la cuantificación vectorial, se incluyeron los datos de la subimagen con la característica paso bajas, se realizará primero la extracción de este conjunto de valores, de modo que el conjunto restante, sea el correspondiente a los índices de las subimágenes de la imagen transformada, por lo que se procederá a realizar el método inverso de la cuantificación vectorial.

De esta manera y con la ayuda del "Multiresolution Codebook", el mismo que fue usado para generar el conjunto de índices, se extraerán los vectores para reconstruir las subimágenes. Al final de este bloque se tendrá una imagen que tiene la característica de ser muy aproximada a la imagen que resultó del proceso de transformación (DWT), ya que como se enfatizó anteriormente, en esta parte hay cierta pérdida de la imagen original.

3.6.6 Bloque de transformación inversa

Con la imagen resultante del proceso anterior, el siguiente paso es aplicar la transformada wavelet inversa a esa imagen. Para realizar esta tarea también serán

utilizados los mismos coeficientes de los filtros (wavelets) utilizados en la transformación directa. Como resultado se tendrá una imagen aproximada a la imagen original.

El grado de aproximación dependerá de los parámetros que se hayan utilizado en el proceso de compresión, como el nivel de transformación, la dimensión de los vectores utilizados en la cuantificación, los codebook's utilizados y en el caso de no almacenar los datos en un archivo, es decir, si se transmiten, también hay que considerar el posible ruido agregado al conjunto de datos finales, producto del canal de transmisión. Este factor depende en gran parte de los protocolos usados, como el TCP/IP o por problemas físicos en la red. Estos factores no son tomados en cuenta en este proyecto debido al alcance del mismo y además porque en la actualidad, la implementación de los protocolos es muy eficiente y en caso de suceder errores en la transmisión de datos, el mismo protocolo trate de resolver el problema.

El siguiente capítulo muestra los resultados obtenidos al aplicar la técnica de compresión a imégenes geográficas. Se explica de manera breve y concisa como es que la calidad de la imagen es afectada a distintas tasas de compresión, mencionando asímismo, cómo se pueden establecer los parámetros para que la técnica varíe la tasa de compresión.

Capítulo 4

Resultados

4.1 Introducción

Las pruebas fueron realizadas a imágenes producidas por datos geográficos y a imágenes multiespectrales. Las primeras son resultado de procesar datos geográficos (modelo digital de terreno) con un software para sistemas de información geográfico (GRASS, Ermapper). Las imágenes que producen estas herramientas pueden ser tanto en color como en tonos de grises, pero este proyecto de investigación solo contempló el uso de imágenes en grises.

Las pruebas consistieron en determinar algunas medidas de error que pudieran establecer de manera cuantitativa la similitud entre las imágenes original y descomprimida. Las medidas que se usaron fueron principalmente el error medio cuadrático (E_{rms}) entre las imágenes y el pico de la proporción señal–ruido ($PSNR$). Estas medidas fueron usadas cada vez que una imagen era comprimida a una tasa diferente, es decir, cada vez que se modificaban los parámetros de compresión para obtener más o menos compresión, la cual se reflejaba en el tamaño del archivo final.

A continuación se describen de manera breve, las medidas E_{rms} y $PSNR$.

4.2 Medidas de error

El error medio cudrático "E_{rms}" entre dos imágenes, es usado para determinar el grado de similitud en promedio, que existe entre ellas. En ocasiones se le denomina como "criterio de fidelidad" ya que puede indicar el nivel de pérdida de información de la imagen comprimida con respecto a la original[25]. Si $f(x,y)$ representa una imagen de entrada y $\hat{f}(x,y)$ representa una aproximación a $f(x,y)$, entonces para cada valor x e y el error $e(x,y)$ entre $f(x,y)$ y $\hat{f}(x,y)$, puede ser expresado como:

$$e(x,y) = \hat{f}(x,y) - f(x,y) \tag{4.1}$$

Si se quisiera percibir de manera visual el error entre las imágenes, la ecuación 4.1 indica que si se restan las imágenes, puede ser observada la imagen de error, que en el caso óptimo esta imagen tendría que estar compuesta solo de ceros, por lo cual si se establece que un pixel negro tiene un valor cero, entonces la imagen de error se observaría completamente negra.

El error total de las imágenes puede ser cuantificado como:

$$E = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x,y) - f(x,y)] \quad \text{donde} f(x,y) \quad \text{es de dimensión } M \times N \qquad (4.2)$$

Por tanto el error medio cuadrático (E_{rms}) entre las imágenes, es la raíz cuadrada del error cuadrático promedio sobre el arreglo $M \times N$:

$$E_{rms} = \left[\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x,y) - f(x,y)]^2 \right]^{1/2} \qquad (4.3)$$

La desventaja de usar esta medida de error, es que indica de manera poco objetiva qué tanto la imagen descomprimida (o imagen aproximada) se parece a la original, ya que podemos aceptar que si el valor E_{rms} es cercano a cero, entonces las imágenes son muy parecidas o casi idénticas, pero si este valor no es cercano a cero (quizá mayor a uno) no podemos saber que tanto las imágenes varían, de hecho es muy difícil determinar una función matemática que corresponda a la percepción humana de una estimación del error.

Existen otras medidas como la proporción señal ruido (SNR) o el pico de la proporción señal–ruido ($PSNR$), las cuales también hacen una evaluación numérica de la cantidad de pérdida de la imagen descomprimida y que son usadas comúnmente para determinar la calidad del método de compresión.

El pico de la proporción señal–ruido en imágenes mide o estima la calidad de la imagen reconstruida (descomprimida) con la imagen original. La idea básica es calcular un número que refleje esta calidad. Las imágenes reconstruidas con altos índices son juzgadas como mejores que las que tienen índices bajos. La fórmula del pico de la SNR está dada por:

$$PSNR = 10 \log_{10} \left(\frac{255}{\sqrt{\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x,y) - f(x,y)]^2}} \right) \qquad (4.4)$$

donde M y N correspoden al número de renglones y columnas de la imagen, respectivamente. El $PSNR$ es medido en decibeles (dB) y sus valores típicos están entre 20 y 40 dB. Estos valores por si solos no indican nada, pero sí lo hacen al momento de comparar estas medidas con otros métodos o técnicas de compresión.

4.3 Tasa de asignación de bits

Al realizar las pruebas de compresión también es posible establecer otra medida o parámetro del compresor, ésta se refiere a establecer la tasa de asignación de bits para cada índice del vector cuantificado. Esta tasa es establecida considerando un codificador de entropía óptimo, para este método de compresión en particular. Esta tasa se basa en el hecho de que en cada nivel de transformación y para cada subimagen se puede establecer la cantidad de información promedio de cada "codebook", la cual es llamada medida de entropía y es calculada por[3]:

$$H(Y) = -\frac{1}{n}\sum_{i=1}^{L} p(Y_i)\log_2 p(Y_i) \qquad \text{bits por pixel} \tag{4.5}$$

donde $p(Y_i)$ es la probabilidad de elegir el vector Y_i con tamaño n, perteneciendo al "codebook" de un nivel de transformación, en cada una de las subimágenes. De esta forma la tasa de bits global R_t para la técnica de compresión basada en la transformada wavelet, es calculada por la suma de la entropía en cada subimagen de acuerdo a:

$$R_t = \frac{1}{2^{2I}} R_I^{dc} + \sum_{i=1}^{I} \frac{1}{2^{2I}} \sum_{d=(H,V,D)} R_{i,d} \qquad . \tag{4.6}$$

donde I indica el nivel de transformación máximo, R_I^{dc} es la tasa de asignación de bits de la subimagen paso bajas y $R_{i,d}$ es la tasa de asignación de bits en el nivel de transformación i y para la subimagen con orientación d (Horizontal, Vertical y Diagonal).

4.4 Pruebas

Para ejecutar el programa de compresión es necesario indicar una serie de opciones o parámetros que de alguna manera afectarán a la tasa de compresión. Estos parámetros son principalmente:

- el número de niveles de transformación y
- el tamaño o dimensión de los vectores de las subimágenes

Es importante mencionar en esta parte que el valor de esos parámetros afectarán la calidad de la imagen reconstruida. Para que la imagen reconstruida no sufra tanta pérdida, existe un compromiso entre los niveles de transformación y la dimensión de los vectores, ya que si esto es deseable, los valores tanto de transformación como de la dimensión, tendrán que ser pequeños. Por ejemplo, si lo que se pretende es tener una tasa baja de compresión aproximadamente de 2:1 y 5:1, entonces el número de niveles de transformación deberá ser igual a 1 y la dimensión del vector deberá ser de 2×2, con el propósito de codificar la mayor catidad de datos pertenecientes a la

imagen que se está comprimiendo.

La relación entre tasa de compresión y los valores de los parámetros no es tan directa, ya que como se mencionó anteriormente (capítulo 3) se pueden establecer distintos tamaños de vectores en las subimágenes en cada nivel de transformación. Para imágenes grandes (1200 × 1200) está relación se mantuvo más directa, no así para imágenes menores o iguales a 512 × 512, ya que mientras más pequeña sea la imagen, menor es el grado de compresión que se puede lograr de acuerdo a una tasa límite, la cual es indicada por la entropía. Para imágenes de 1200 × 1200 fue posible establecer la siguiente relación entre parámetros y tasa (cuadro 4.1):

Tasa	Nivel (DWT)	Dimensión (vector)	HH n1
2:1	1	2 × 2	si
3:1	1	2 × 2	no
4:1	1	4 × 4	si
5:1	1	4 × 4	no
6:1	1	8 × 8	no
10:1	2	4 × 4	si
11:1	2	4 × 4	no
17:1	2	8 × 8	si
19:1	2	8 × 8	no
23:1	2	16 × 16	si
37:1	3	8 × 8	si
44:1	3	8 × 8	no

Cuadro 4.1: Relación de la tasa de compresión y los parámetros.

En el cuadro 4.1, la columna que indica, **HH n1**, se refiere a tomar en cuenta o no, a la subimagen con orientación diagonal del nivel de transformación 1. Esta subimagen tiene la característica de que sus valores corresponden a niveles de detalle (o coeficientes detalle) de la imagen original no muy significantes (son valores muy cercanos a cero), por lo cual es posible eliminarlos o no considerarlos, ya que aportarán poca información al momento de reconstruir la imagen y en cambio ayudan a aumentar la tasa de compresión.

Las dimensiones de los vectores mostrados en el cuadro 4.1 fueron utilizados en cada una de las subimágenes de cada nivel de transformación, es decir, que para obtener por ejemplo un tasa de compresión de 10:1 (renglón 6), se tuvieron que utilizar vectores de 4×4 en cada subimagen de cada uno de los dos niveles de transformación.

Al establecer vectores de diferente dimensión en cada nivel de transformación, se pudo variar el nivel de compresión hasta alcanzar tasas de 100:1.

Para que los errores de reconstrucción no fueran tan altos, a las subimágenes del nivel mas alto de transformación se les asignaron vectores de dimensión corta, típicamente 2×2, mientras que en cada nivel subsecuente (hacia abajo) la dimensión de los vectores se les aumentó en un factor de 2, es decir, si la imagen tenía 3 niveles de transformación, al momento de cuantificar, se generaban vectores de 2×2 para el nivel de transformación más alto, de manera que para las del nivel 2 se generaban vectores de 4×4 y a las del nivel 1, vectores de 8×8 (figura 3.8). Para esto también fue tomada en cuenta la posibilidad de eliminar la subimagen HH del nivel 1. Esta consideración de igual forma no pudo haber sido tomada en cuenta y para lo cual se tuvieron que asignar vectores de dimensiones arbitrarias, lo que si es importante es tratar de seguir la regla de asignar vectores de dimensiones cortas a las subimágenes en los niveles de transformación altos y vectores más grandes a las de los niveles subsecuentes, con el propósito de minimizar la pérdida de información. De esta forma, también para imágenes de 1200×1200, se creo la siguiente relación entre tasa y parámetros de compresión (cuadro 4.2).

Tasa	Niveles (DWT)	n4	n3	n2	n1	HH n1
7:1	2			2×2	4×4	si
8:1	2			2×2	4×4	no
10:1	2			2×2	8×8	si
11:1	2			2×2	16×16	si
22:1	3		2×2	4×4	8×8	si
24:1	3		2×2	4×4	8×8	no
27:1	3		2×2	4×4	16×16	si
28:1	3		2×2	4×4	16×16	no
29:1	3		2×2	4×4	32×32	no
50:1	3		4×4	8×8	16×16	si
53:1	3		4×4	8×8	16×16	no
68:1	4	2×2	4×4	8×8	16×16	si
73:1	4	2×2	4×4	8×8	16×16	no
96:1	4	4×4	4×4	8×8	32×32	si
100:1	4	4×4	4×4	8×8	32×32	no

Cuadro 4.2: Relación entre la tasa de compresión y los parámetros.

Como se puede notar en el cuadro 4.2 no fueron tomados en cuenta niveles de transformación más grandes a 4, esto debido a que la generación de "alias" o defectos hacían que las imágenes reconstruidas tuvieran demasiada distorsión en los bordes.

La primera imagen de prueba que se muestra es la clásica imagen de Lena, la cual tiene una dimensión de 512×512 pixeles y hasta 256 tonos de grises y su archivo en formato "pgm" ocupa 262 174 bytes (figura 4.1). Cabe mencionar que esta imagen es usada para comparar los resultados de las técnicas de compresión o codificadores que se proponen, utilizando para esto las medidas de error mencionadas anteriormente,

con el propósito de observar las variaciones en los resultados y determinar si una técnica fue mejor que otra. El codebook que se utilizó fue generado con 2 imágenes, en donde una de ellas era la imagen Lena original. El número de vectores de cada codebook que se generó fue 256.

En el cuadro 4.3 se muestran datos completos de las pruebas que se realizaron con la imagen Lena.

Imagen Lena 512 × 512, 262 174 bytes							
N. dwt	Dim. Vectores	Tasa (R_t:bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (s)
1 -x	2 × 2	3.00	95495	2:1	2.470170	34.558907	5
3 d1 -x	2 × 2, 4 × 4, 8 × 8	0.375	10880	24:1	8.004479	24.338643	5

Cuadro 4.3: Resultados de las pruebas de la imagen Lena 512 × 512.

En el cuadro 4.3 y en las siguientes que se van a presentar, **N dwt** se refiere a los niveles de transformación, el **d1** que aparece en el segundo renglón del cuadro, indica solamente que la dimensión de los vectores incrementará de forma progresiva en la codificación. **Dim. Vectores** se refiere a la dimensión de los vectores en cada nivel de transformación, empezando por el nivel más alto, es decir, 2 × 2, 4 × 4, 8 × 8, quiere decir que el nivel 3 tienen una dimensión de 2 × 2, en el nivel 2 de 4 × 4 y en el nivel 1 de 8 × 8, R_t es la tasa de asignación de bits por pixel (bpp), **Tamaño** es el tamaño del archivo de la imagen comprimida, **Tasa comp.** es la tasa de compresión que se obtuvo, **Erms** es el error medio cudrático entre las imágenes original y descomprimida, **PSNR** es el pico de la proporción señal–ruido y la **t** se refiere al tiempo de codificación (en segundos).

La figura 4.2 corresponde a la imagen con parámetros **3 d1 -x** (segundo renglón del cuadro 4.3). Se puede notar que en esta imagen, los resultados de las medidas de error no fueron satisfactorios, debido a que el conjunto de vectores de los codebooks fueron creados con imágenes de entrenamiento que resultaron ser poco adecuadas para esta imagen, es decir, si las imágenes de entrenamiento se hubieran escogido con razgos y características muy similares a la que se va a comprimir, los resultados hubieran sido más favorables, aún así la pérdida de calidad es apenas notoria.

Con la finalidad de comparar este esquema de compresión con el estándar JPEG, se presentan a continuación unas gráficas que muestran las medidas de error obtenidas al comprimir la imagen Lena (512 × 512) con JPEG y con esta técnica, usando para esto codebooks generados sin incluir la imagen Lena en las imágenes de entrenamiento (peor caso) y usando codebooks generados con la imagen Lena (mejor caso). En la figura 4.3 se muestra la gráfica generada por el $PSNR$ en (dB) a distintas tasas de compresión (asignación de bits, *bits/pixel*), para cada uno de los casos (JPEG,

Figura 4.1: Original Lena, 512 × 512, 8pp

Figura 4.2: Descomprimida, $E_{rms} = 8.0049$, $PSNR = 24.3386$, $R_t = 0.375$

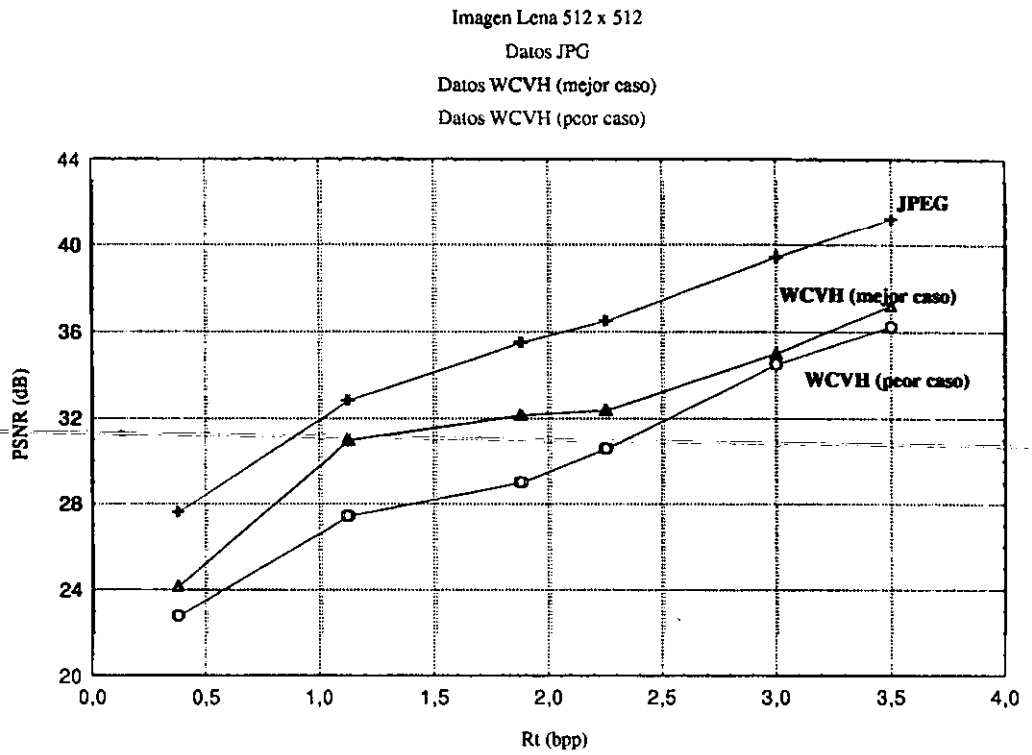

Figura 4.3: Gráfica comparativa del JPEG con esta técnica de acuerdo al $PSNR$.

mejor y peor caso). En la figura 4.4 se muestra la gráfica generada por el E_{rms} a distintas tasas de compresión.

Se nota en esas gráficas (figs. 4.3 y 4.4) como el estándar JPEG muestra mejores resultados que los generados con esta técnica, tanto en el peor como en el mejor caso. Esto se debe principalmente a que el JPEG es el estándar actual en compresión de imágenes fijas y existe demasiado desarrollo para optimizar el rendimiento de este método y aunque los resultados obtenidos por esta técnica no fueron tan malos, indican que se tiene que poner más atención en el proceso de cuantificación; esto porque los resultados publicados en [2] muestran medidas de error mejores a las mostradas aquí tanto para el JPEG como para este método. El hecho de que los resultados obtenidos con este método difieran con los publicados en [2] y que son generados con técnicas basadas en el mismo esquema, se debe precisamente al proceso de cuantificación, ya que por una parte, no se menciona la forma en la que se generaron los codebooks (parámetros del cuantificador), ni fueron publicados el conjunto de codebooks. Por otra parte, el método propuesto en el artículo usaba cuantificación vectorial por búsqueda exhaustiva y particularmente este método no lo considera, lo cual también puede provocar resultados diferentes aunque dichas variaciones no suelen ser muy significativos. Estos resultados no son tan alarmantes debido al tipo

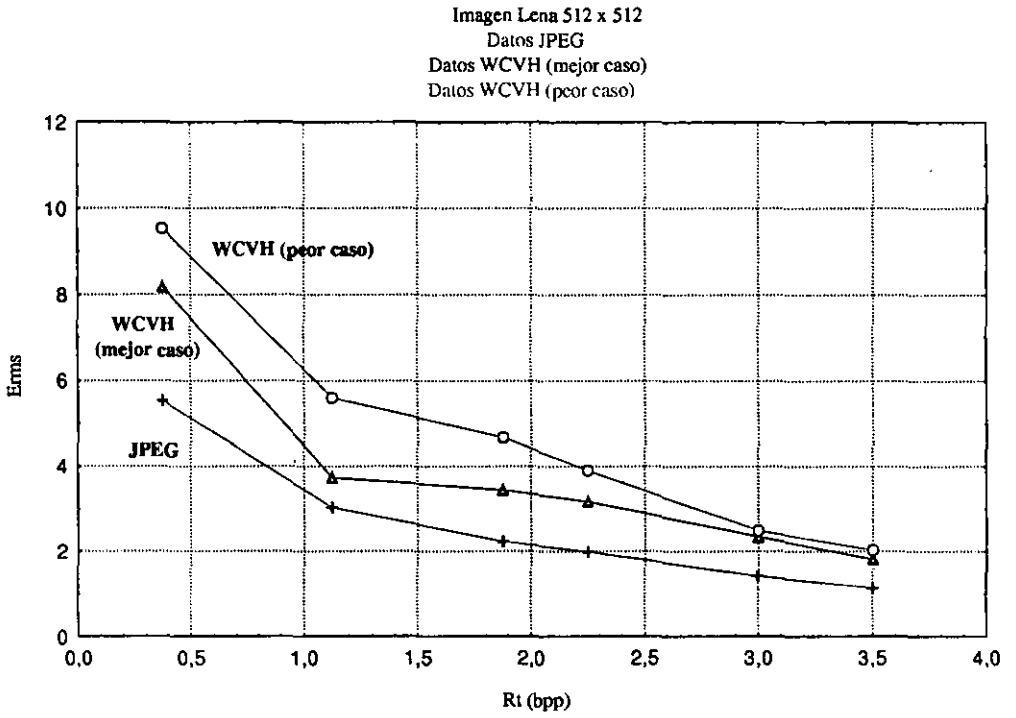

Figura 4.4: Gráfica comparativa del JPEG con esta técnica de acuerdo al E_{rms}

de imágenes que se manejan (geográficas) y no las que son deniminadas como "escenas naturales" (*"pictures"*), ya que una de las característica de la cuantificación vectorial es que es muy favorable para cuando se usa en imágenes con características o reasgos similares, como es el caso de las imágenes geográficas.

Siguiendo con las pruebas, para las imágenes geográficas se utilizaron codebooks de 256 vectores con el propósito de que cada índice fuera codificado en el peor de los casos con un byte.

Las siguientes pruebas fueron realizadas a imágenes producidas por un software que maneja datos de modelos digitales de terreno ("Ermapper") y que para fines de las pruebas, se tomó solamente una región de la imagen (512 × 512 pixeles). Esta imagen ocupa 262 202 bytes (en formato pgm) y es presentada en la figura 4.5. El codebook que se utilizó fue generado por tres imágenes totalmente independientes a la que se comprime.

En el cuadro 4.4 se muestran los resultados completos de las pruebas sobre la imagen de la figura 4.5. Con éstos se generaron unas gráficas, que muestran la relación entre la tasa de asignación de bits y el $PSNR$, E_{rms} y el tamaño final del archivo.

Imagen Dem 512 × 512, 262 202 bytes							
N. dwt	Dim. Vectores	Tasa (R_t:bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (s)
1	2 × 2	3.50	76606	3:1	4.620584	22.198992	5
1 -x	2 × 2	3.00	65107	4:1	7.599994	17.833673	4
2 d1	2 × 2, 4 × 4	1.25	29261	9:1	10.342814	15.078738	6
2 d1 -x	2 × 2, 4 × 4	1.125	26421	10:1	11.116716	14.428497	5
3 d1	2 × 2, 4 × 4, 8 × 8	0.406	10546	24:1	15.087590	11.625451	5

Cuadro 4.4: Resultados de las pruebas de la imagen Dem 512 × 512.

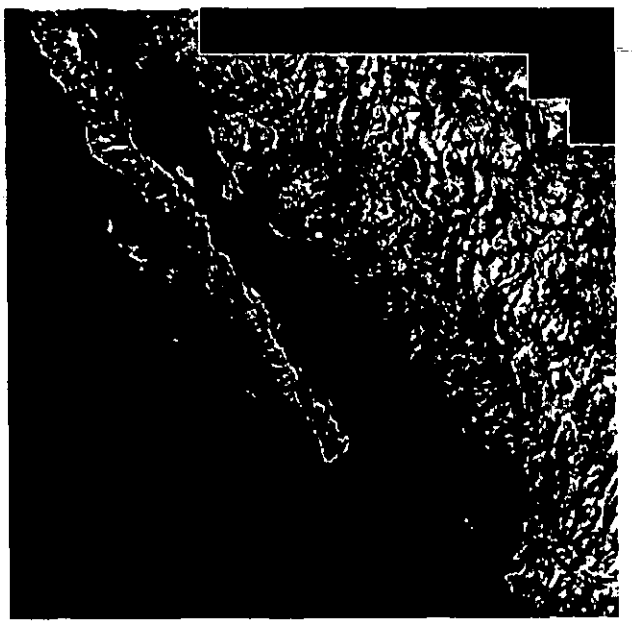

Figura 4.5: Original 512 × 512. 8bpp

Las imágenes de las figuras 4.6 y 4.7, corresponden a los parámetros del primer y último renglón del cuadro 4.4 respectivamente. En estas imágenes, si bien las medidas de error muestran datos bastante malos, la pérdida de calidad de esa imagen es difícil percibirla, en parte por el tipo de imagen y por la resolución a la que se presenta, haciendo notar lo mencionado con respecto a las medidas de error, ya que aunque los datos mostrados son bajos, la calidad visual de la imagen no es mala.

Los resultados del cuadro 4.4 pueden ser apreciados en las gráficas de las figuras 4.8, 4.9 y 4.10. En éstas se puede observar que el rango permisible para estas condiciones está entre una R_t de 1.25 y 3.5, ya que aunque el E_{rms} y el $PSNR$ no son muy satisfactorios, es difícil percibir la pérdida de información que muestra la imagen, obteniendo en el mejor de los casos un archivo de tan sólo 30 Kbytes. Estos resultados

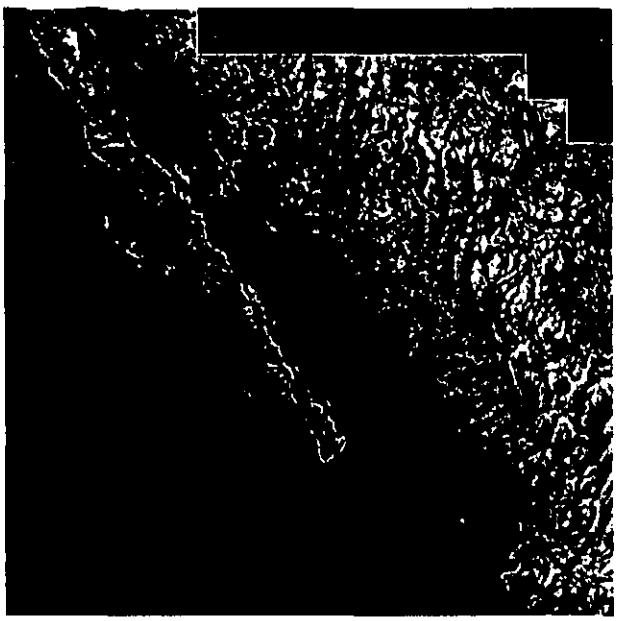

Figura 4.6: Descomprimida, $E_{rms} = 4.620584$, $PSNR = 22.198992$, $R_t = 3.50$

se debieron principalmente a que los codebooks no se crearon de manera adecuada, ya que las imágenes de entrenamiento no fueron escogidas de forma apropiada para esta imágen, ya que como se mencionó anteriormente, la cuantificación vectorial da buenos resultados cuando el conjunto de imágenes de entrenamiento se selecciona acorde con la imagen que se va a comprimir. De acuerdo con este punto es importante mencionar que no se planteó una técnica de compresión para todo tipo de imágenes (*pictures*) sino que las imágenes que se van a comprimir, mantienen las mismas características o rasgos, como se puede apreciar en las imágenes generadas por el modelo digital de terreno y las imágenes multiespectrales que se van a presentar a continuación.

Siguiendo con los datos del modelo digital, se hizo una prueba con la imagen que presenta el modelo digital de terreno de la República Mexicana completa (figura 4.11), obteniendo para esto, los resultados mostrados en el cuadro 4.5.

Imagen Dem República Mexicana 1090 × 715, 779 409 bytes							
N. dwt	Dim. Vectores	Tasa (R_t:bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (s)
1 -x	2 × 2	3.00	197203	4:1	6.171325	17.378920	9
3 d1 -x	2 × 2, 4 × 4, 8 × 8	0.37	21125	36:1	—	—	12

Cuadro 4.5: Resultados de las pruebas de la imagen Dem Rep. Mex. 1090 × 715.

Figura 4.7: Descomprimida, $E_{rms} = 15.08759$, $PSNR = 11.625451$, $R_t = 0.40$

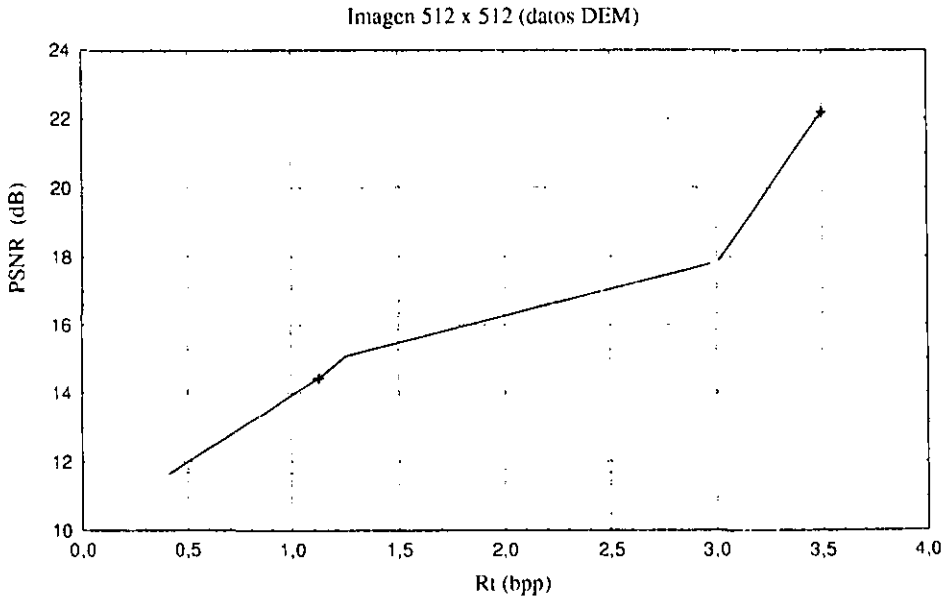

Figura 4.8: Relación entre tasa de asignación de bits (R_t) y $PSNR$.

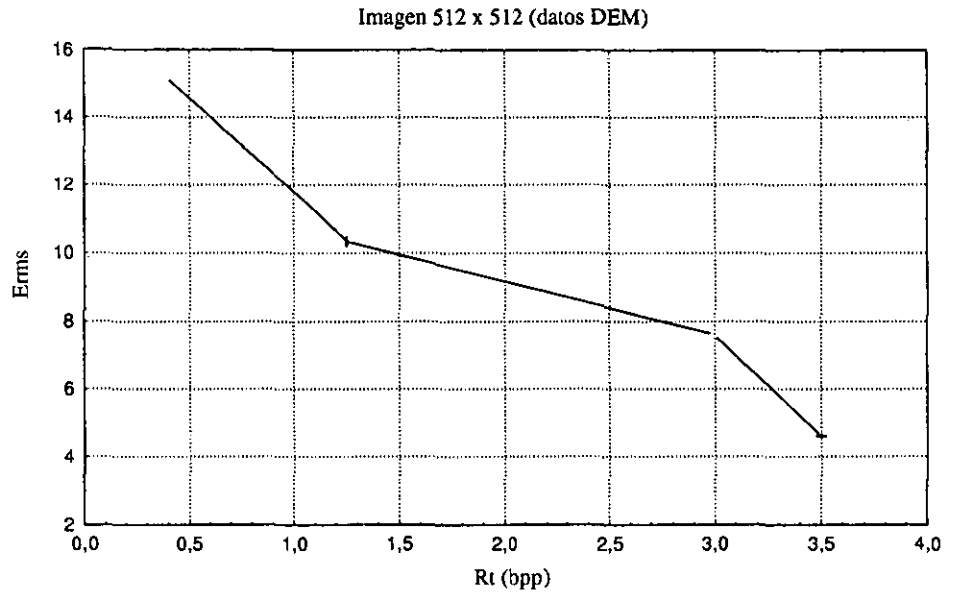

Figura 4.9: Relación entre tasa de asignación de bits (R_t) y el E_{rms}.

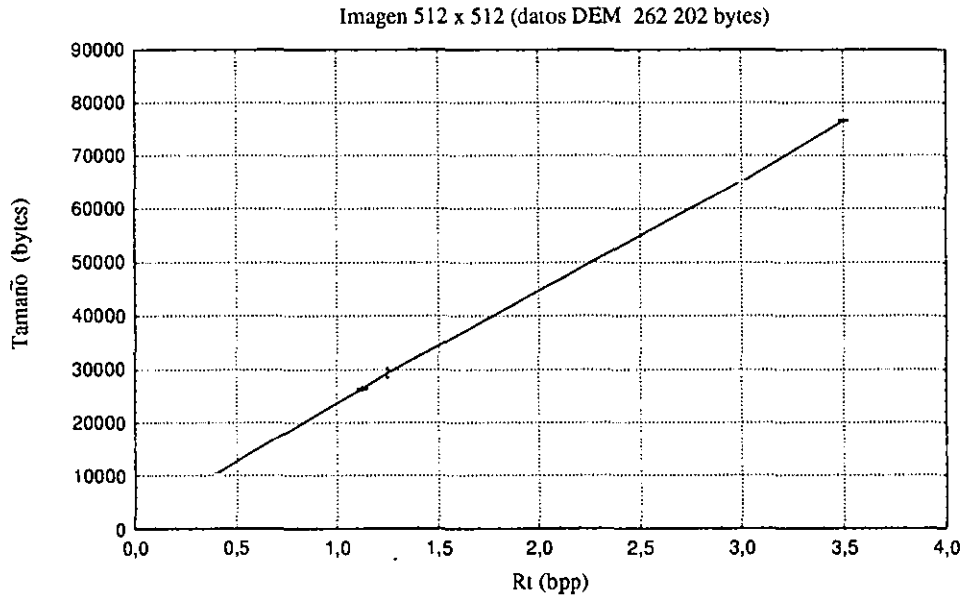

Figura 4.10: Relación entre tasa de asignación de bits (R_t) y el número de bytes finales.

Figura 4.11: Descomprimida, $R_t = 0.375$

Se puede notar en el cuadro 4.5 que el segundo renglón no muestra los valores de error obtenidos, esto se debió a que la imagen sufrió una especie de traslación, producida por la dimensión de la misma y el efecto de aplicar una transformada wavelet a imágenes con este tipo de dimensiones (número impar en el número de renglones). Precisamente por este factor y acompañado de la consideración de periodicidad de la imagen a tomar en cuenta en la transformada wavelet (debido al problema de los bordes), en ocasiones y dependiendo del nivel de transformación que sufra la imagen que a comprimir, la imagen reconstruida presenta una serie de pixeles en los bordes que no precisamete son de la imagen original, esto también ayuda a que los resultados de las medidas de error se disparen y produce el efecto de cierta traslación de la imagen. Lo anterior es difícil percibirlo en la figura 4.11, debido al factor de escala aplicado para ser presentada. También por efecto de la compresión, la imagen descomprimida pierde cierto grado de nitidez.

Las siguientes pruebas fueron realizadas a una imagen multiespectral. Esta imagen mide 4000 pixeles a lo ancho por 3600 a lo alto. Debido a su gran tamaño, lo cual no permite ser visualizada en un monitor, las pruebas se realizaron sobre una región de 1200×1200 de esa imagen. Como se había mencionado, esta imagen multiespectral está formada por 7 bandas, en donde tres de esas son las que comúnmente se utilizan para hacer el análisis de una región. Esta bandas son la 2, 4 y 7 y representan la imagen captada a distintas longitudes de onda.

Las pruebas con este tipo de imagen se realizaron generando el conjunto de codebooks con las regiones de 1200 × 1200 de las bandas 1, 2, 4, y 7, haciendo el papel de imágenes de entrenamiento. De la misma forma, como en los casos anteriores, el tamaño de cada codebook fue de 256 vectores.

Después de presentar los resultados obtenidos al comprimir y descomprimir cada una de estas bandas, se hará una comparación de éstos por medio de unas gráficas generadas a patir de los datos mostrados en cada uno de sus cuadros.

Los primeros resultados que se muestran son los de la banda 2 (cuadro 4.6). La imagen multiespectral muestra una zona del estado de Hidalgo, conocida como Molango (figura 4.12).

Imagen Molango Banda 2, 1 440 045 bytes							
N. dwt	Dim. Vectores	Tasa (R_t:bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (s)
1	2 × 2	3.50	435439	3:1	1.102667	26.726044	19
1 -x	2 × 2	3.00	357963	4:1	1.136595	26.462322	17
2 d1	2 × 2, 4 × 4	1.25	159040	9:1	1.319478	25.163458	23
2 d1 -x	2 × 2, 4 × 4	1.12	139280	10:1	1.332348	25.078749	20
3 d1 -x	2 × 2, 4 × 4, 8 × 8	0.37	46787	30:1	1.728067	22.811029	22
4 d1 -x	4×4,4×4,8×8,32×32	0.08	10650	135:1	2.909612	18.247297	23

Cuadro 4.6: Resultados de las pruebas de la imagen Molango banda 2, 1200 × 1200.

La imagen de la figura 4.13, muestra los resultados obtenidos con los parámetros del quinto renglón del cuadro 4.6. Debido a la resolución de la imagen original y la necesidad de reducirlas para ponerlas en este trabajo, es difícil poder percibir algun tipo de degradación o error, ya que la imagen resultó ser un tanto obscura, por tanto para apreciar de qué manera se modificó, se tuvo que realizar el proceso de igualación del histograma tanto a la imagen original como a la descomprimida, ya que al aumentar el brillo y el contraste se perdía nitidez en las imágenes y no permitían dejar ver las modificaciones. Al compararlas despúes de realizar este proceso, se puede observar de manera clara la pérdida de datos de la descomprimida, ya que la imagen es un poco más obscura que la original. Lo importante en este caso es que la imagen se degradó en la misma proporción, es decir, no solo una región de la imagen se degradó sino que ésto se hizo de manera general; lo cual no es grave ya que con un procesamiento simple, se logra recuperar mayor información de la imagen previa a los cambios.

El cuadro 4.7 presenta los resultados obtenidos al compactar y descompactar la imagen de 1200 × 1200 de la banda 4 de la imagen multiespectral de Molango (figura 4.14).

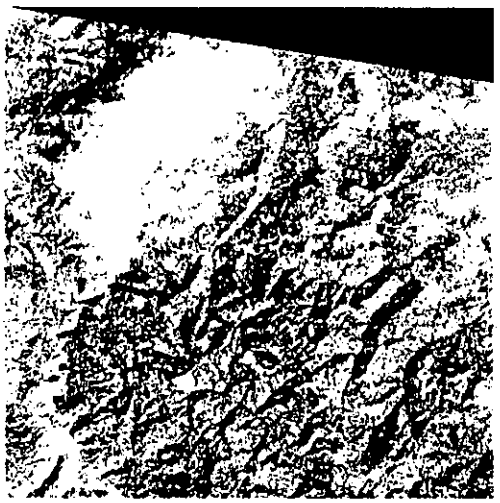

Figura 4.12: Original Molango Banda 2, 1200 × 1200, 8bpp

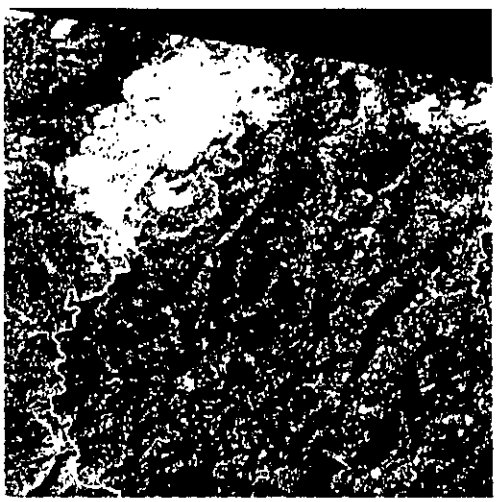

Figura 4.13: Banda 2, $E_{rms} = 1.728067$, $PSNR = 22.811029$, $R_t = 0.375$

Imagen Molango Banda 4, 1200 × 1200 1 440 045 bytes							
N. dwt	Dim. Vectores	Tasa (R_t:bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (s)
1	2 × 2	3.50	570854	2:1	1.360212	34.478443	27
1 -x	2 × 2	3.00	478655	3:1	1.523912	33.490948	22
2 d1	2 × 2, 4 × 4	1.25	210947	6:1	2.710209	28.484392	31
2 d1 -x	2 × 2, 4 × 4	1.12	188609	7:1	2.762260	28.388690	27
3 d1 -x	2 × 2, 4 × 4, 8 × 8	0.37	64713	22:1	4.908625	23.308975	30
4 d1 -x	4×4,4×4,8×8,32×32	0.08	15149	95:1	8.374202	18.631840	27

Cuadro 4.7: Resultados de las pruebas de la imagen Molango Banda 4, 1200 × 1200.

La imagen de la figura 4.15 es generada con los resultados obtenidos con los parámetros del quinto renglón del cuadro 4.7. Como en los casos anteriores no es tan fácil percibir los errores o pérdida de calidad en la imagen, debido a la resolución a la que se presenta. Esta banda fue captada con una longitud de onda más visible, por lo que no es tan obscura como la anterior; debido a esto no fue necesario aplicar el proceso de igualación del histograma y pueden ser apreciadas sin modificaciones. Es importante mencionar que esta imagen resulta ser la que contiene más información del rango visible, esto puede ser notado en primera, porque no es tan obscura como las demás bandas, lo cual quiere decir que los valores de sus pixeles están mas distribuidos a lo largo de la escala de grises (histograma plano) y no como en la anterior que se podía notar que los valores estaban más cargados hacia el lado obscuro. Esto también puede ser notado en el resultado del renglón seis, columna cinco de su cuadro, ya que en comparación con el correspondiente del cuadro de la banda 2, se alcanzó solo una tasa de 95:1 y no 135:1 como la banda 2. Lo cual puede indicar que las imágenes que tengan una distribución más uniforme de los valores de sus pixeles en la escala completa de grises, alncanzaran menores tasas de compresión, que las que no tienen esta característica.

En el cuadro 4.8 se muestran los resultados de la banda 7 de una región de 1200 × 1200 pixeles de la imagen Molango (figura 4.16).

La figura 4.17 muestra una imagen con los resultados obtenidos al aplicar los parámetros del quinto renglón del cuadro a la imagen original. Al igual que la banda 2, estas imágenes también son un tanto obscuras, por lo que para mostrar los resultados se realizó de nueva cuenta el proceso de igualación del histograma.

A partir de los resultados mostrados en los cuadros anteriores fue posible crear una serie de gráficas que describen el comportamiento y la relación que existe entre la tasa de asignación de bits, que es la que establece cuánto se comprime una imagen, con las medidas de error ($PSNR$ y E_{rms}) y el tamaño final de los archivos que

Figura 4.14: Original Molango Banda 4, 1200 × 1200, 8bpp

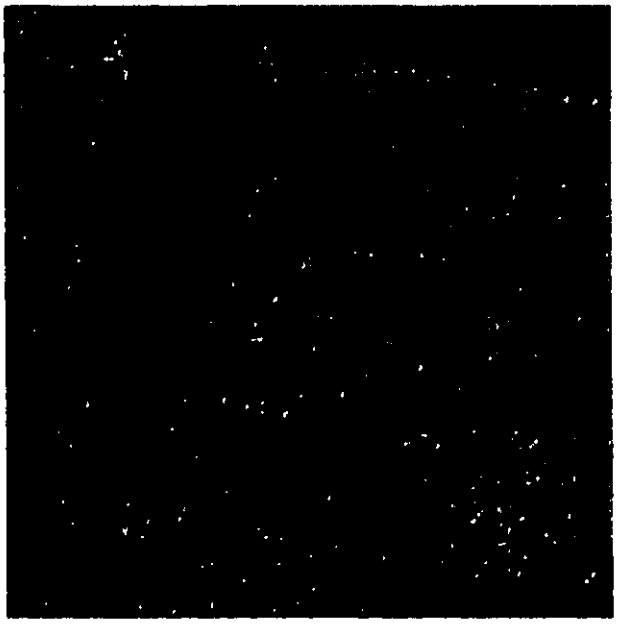

Figura 4.15: Banda 4, $E_{rms} = 4.908628$, $PSNR = 23.308975$, $R_t = 0.375$

Imagen Molango Banda 7, 1200 × 1200 1 440 045 bytes							
N. dwt	Dim. Vectores	Tasa (R_t, bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (*)
1	2 × 2	3.50	499121	2:1	1.137793	24.930584	23
1 -x	2 × 2	3.00	410576	3:1	1.206924	24.416128	19
2 d1	2 × 2, 4 × 4	1.25	184203	7:1	1.616964	21.864433	27
2 d1 -x	2 × 2, 4 × 4	1.12	161510	8:1	1.642834	21.725334	34
3 d1 -x	2 × 2, 4 × 4, 8 × 8	0.37	55410	25:1	2.531819	17.930935	26
4 d1 -x	4×4,4×4,8×8,32×32	0.08	12364	116:1	4.558160	12.683607	26

Cuadro 4.8: Resultados de las pruebas de la imagen Molango Banda 7, 1200 × 1200.

guardan los datos de las imágenes comprimidas.

En las gráficas de la figura 4.18, se muestra la relación entre la tasa de asignación de bits (R_t) con el $PSNR$ obtenido para cada una de las bandas. En éstas se puede observar cómo el comportamiento de las bandas es muy similar, ya que la forma de sus curvas son muy parecidas. Se puede notar cómo la banda 4 fue la que obtuvo los mejores resultados y la banda 7 los peores, aún así todas se mantuvieron dentro del rango permisible hasta una tasa $R_t = 1.125$, con la que se obtiene una tasa de compresión de alrededor de 30:1. El hecho de que la banda 4 fuera la que tuviera los mejores resultados fue muy importante para la técnica de compresión, ya que los resultados hasta una tasa de asignación de 0.37 estuvieron por arriba de los 20 dB y fue importante porque como ya se mencionó, **ésta es la que tiene la mayor cantidad de información visual**. También es posible observar cómo la gráfica de la banda 2 se mantuvo casi constante en el rango de 1.125 a 3.5, lo cual ocasionó que no hubiera mucha pérdida de información en ese rango, resultando un tanto lógico este hecho, ya que la imagen de esta banda no presenta muchas variaciones en sus tonalidades.

En las gráficas de la figura 4.19, se puede observar la relación entre la tasa de asignación de bits y el E_{rms}. En éstas es posible apreciar, aunque a primera vista resulte engañoso, cómo los resultados hasta una tasa de asignación de 0.37 están por debajo de un error medio cuadrático de 5.0, lo cual es bueno; y por arriba de este valor los resultados de la banda 4 tienden a ser malos.

También en la figura 4.20 se presentan las gráficas que se obtuvieron de la relación de la tasa de asignación y el tamaño final de los archivos. En éstas es importante apreciar cómo sus formas son idénticas, lo cual se explica en parte porque los tamaños de los archivos originales medían lo mismo.

Con la ayuda de las gráficas mostradas, se pueden establecer rangos que permitan determinar que tanta pérdida existirá a cierta tasa y determinar un tamaño

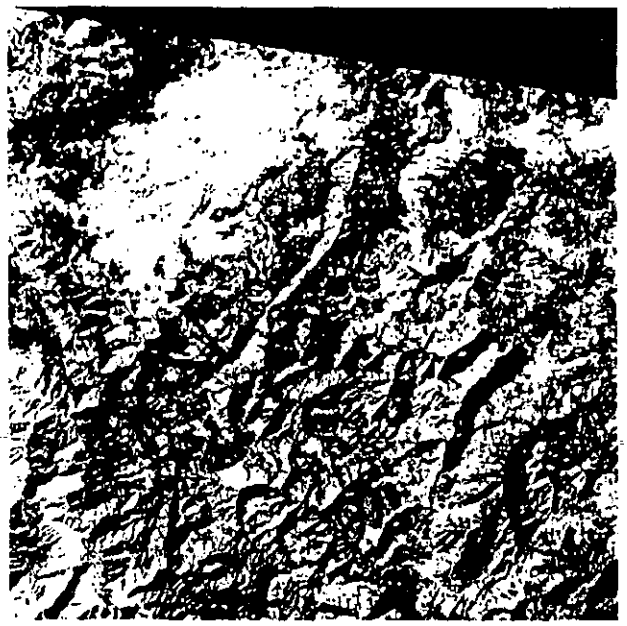

Figura 4.16: Original Molango Banda 7, 1200 × 1200, 8bpp

promedio del archivo final. La ventaja de realizar lo anterior radica en el hecho de que las imágenes multiespectrales no varían mucho su forma y para ciertas regiones importantes se pueden adecuar los codebooks para obtener resultados semejantes o mejores a los mostrados anteriormente. La única desventaja es precisamente la de no poder hacer consideraciones o tomar parámetros para aplicarlas a todo tipo de imágenes.

De las imágenes que resultaron de aplicar la técnica a una tasa de compresión alrededor de 100:1, se generó una imagen RGB solamente para comparar la calidad visual de ésta con la generada con los datos originales (figura 4.21). Esta imagen se muestra en la figura 4.22.

Se puede notar que la calidad visual de la imagen de la figura 4.22, es bastante mala en comparación con la mostrada con los datos originales, pero es importante mencionar que aún a esa tasa, la imagen presenta razgos y características de la imagen original, es decir, la información visual que proporciona esta imagen no se perdió, ya que es posible notar claramente los relieves en las montañas, las zonas en donde existe vegetación e incluso algunas de las zonas por donde pasa un río.

Con el único propósito de comparar visualmente la imagen de la figura 4.22 con una generada de manera similar, las imágenes fueron guardadas con un formato *JPEG* a una tasa también alrededor de 100:1, para lo cual se presenta también el resultado de una imagen RGB (figura 4.23). Las imagenes en formato JPEG fueron

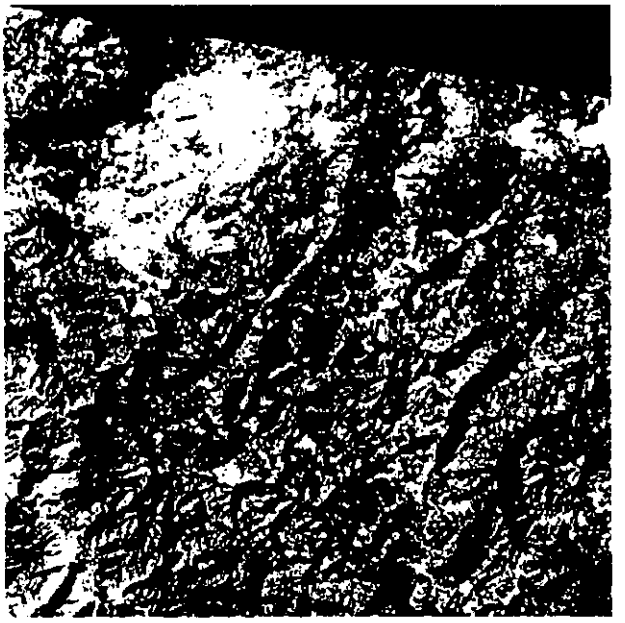

Figura 4.17: Banda 7, $E_{rms} = 2.531819$, $PSNR = 17.930935$, $R_t = 0.375$

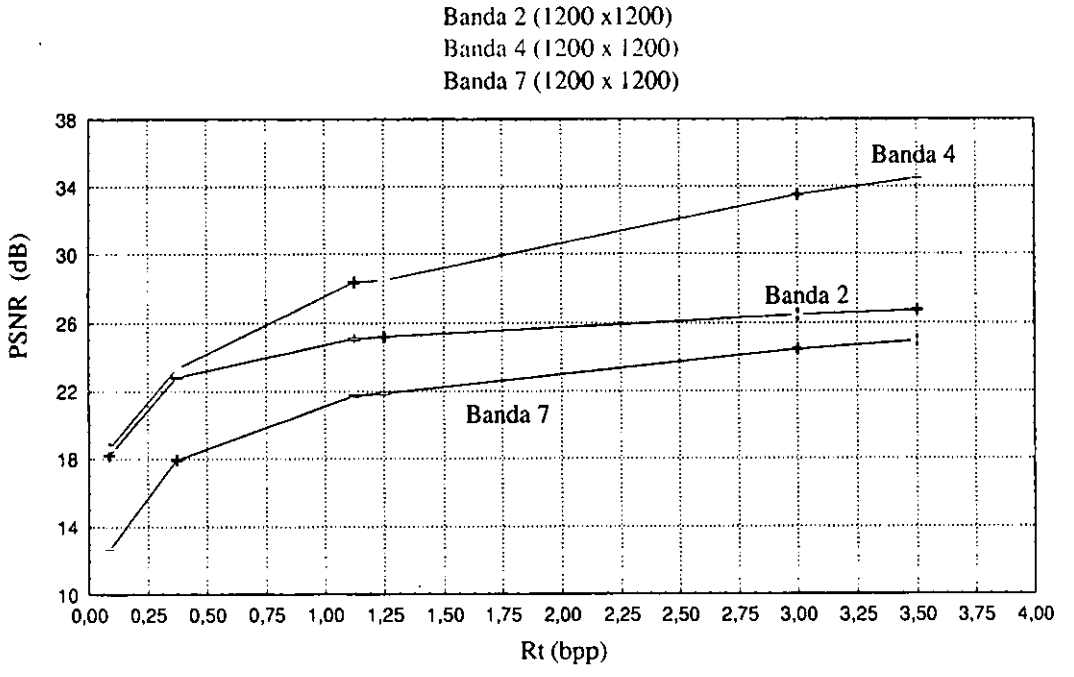

Figura 4.18: Relación tasa de asignación de bits *vs* $PSNR$.

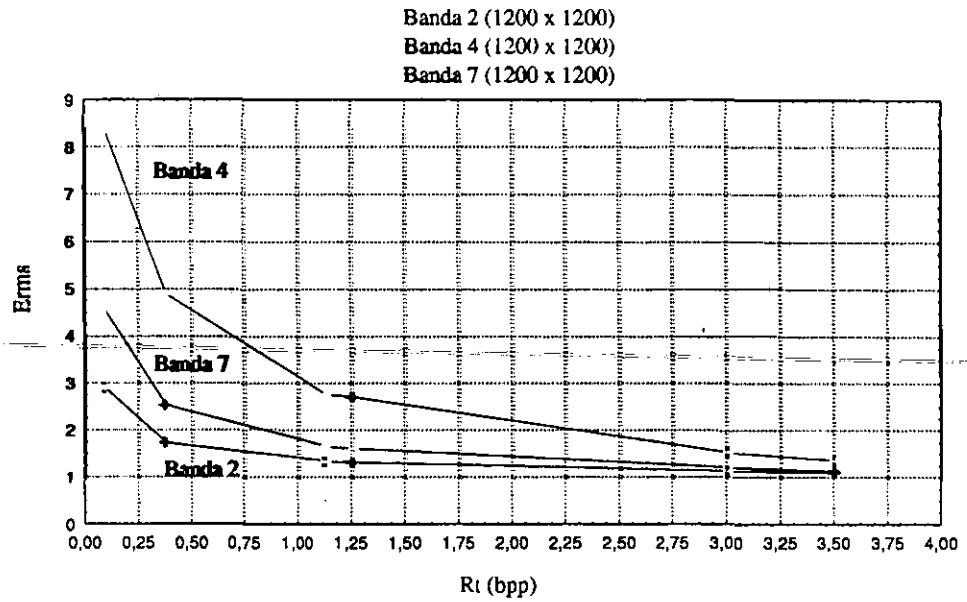

Figura 4.19: Relación tasa de asignación de bits *vs* E_{rms}.

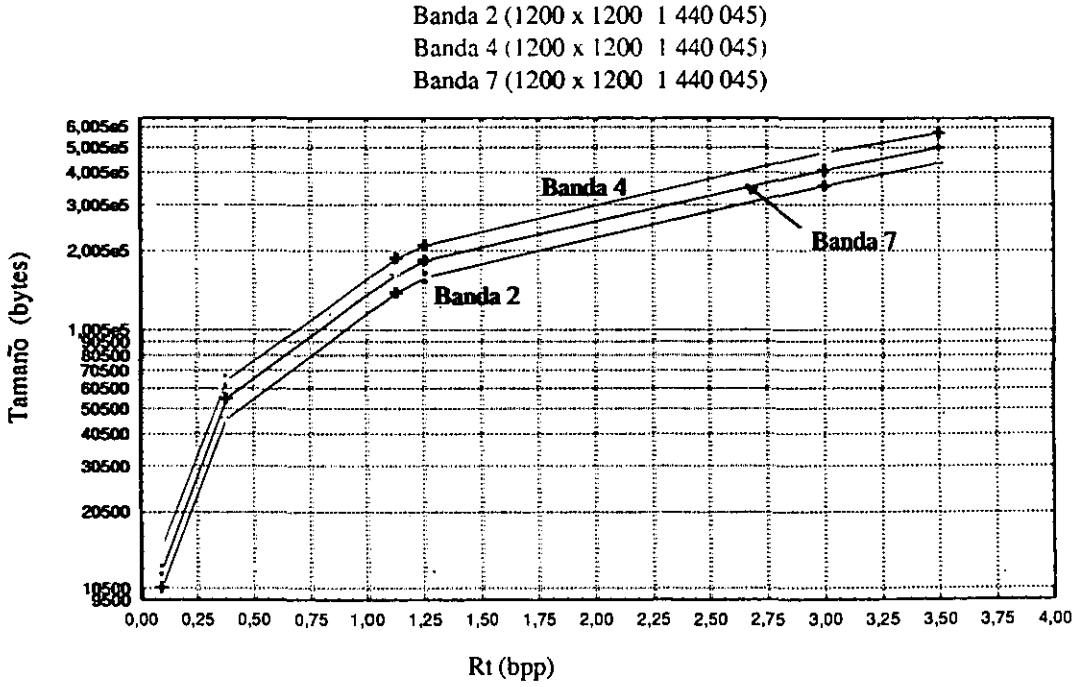

Figura 4.20: Relación tasa de asignación de bits *vs* número de bytes del archivo final.

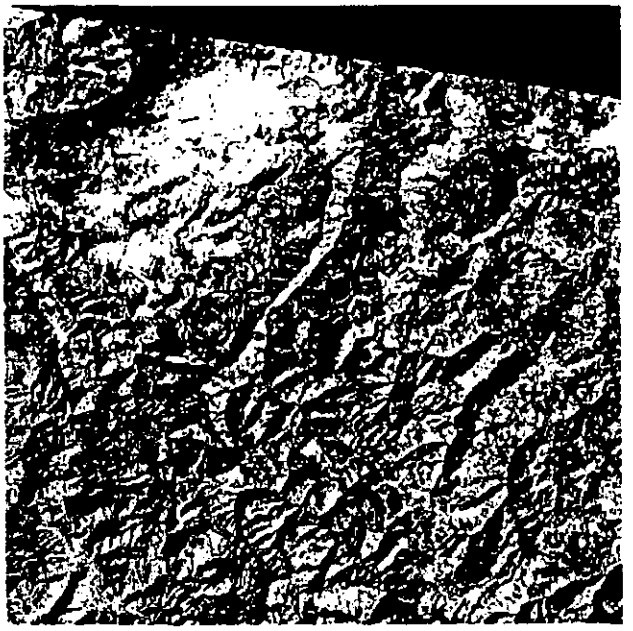

Figura 4.21: Imagen RGB, formada con las bandas 7:4:2 (datos originales).

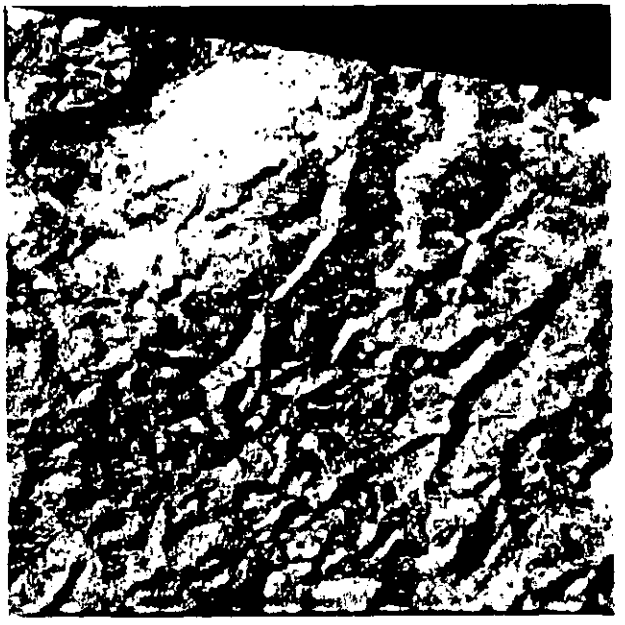

Figura 4.22: Imagen RGB, formada con las bandas 7:4:2 a una tasa de 100:1

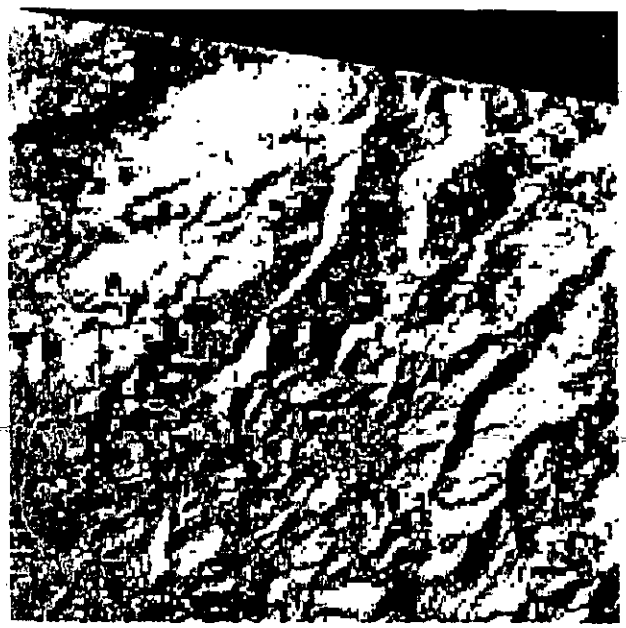

Figura 4.23: Imagen RGB, (bandas 7:4:2) a una tasa de 100:1, generadas con JPEG.

generadas con un software de edición de imágenes de SGI[1](*imgview*).

Se pueden notar en la figura 4.23 algunos problemas que tiene el formato JPEG cuando es utilizado a tasas altas de compresión. Estos se refieren por un lado a generar una imagen "pixelada", producida por la necesidad de partir la imagen en bloques de 8 × 8 y por otro lado la poca cantidad de coeficientes para cuantificar los bloques transformados (producto del algoritmo de compresión mismo). En esta imagen se puede notar que fue demasiada la cantidad de pérdida de información de la imagen original, al no poder determinar características ni zonas de la imagen original.

Para concluir el proceso de pruebas, ahora se consideró la imagen multiespectral completa. En primer lugar se hizo una prueba a la banda 4 (4000 × 3600 pixeles), considerando el factor de compresión dado por los parámetros del quinto renglón de los cuadros anteriores. Los resultados son mostrados en el cuadro 4.9.

Imagen Molango, 4000 × 3600, 14 400 042 bytes							
N. dwt.	Dim. Vectores	Tasa (R_t:bpp)	Tamaño (bytes)	Tasa comp.	E_{rms}	$PSNR$ (dB)	t (s)
3 d1 -x	2 × 2, 4 × 4, 8 × 8	0.37	616833	23:1	4.890450	22.518618	335

Cuadro 4.9: Resultados de las pruebas de la imagen Molango banda 4, 4000 × 3600.

[1]Silicon Graphics

Para poder ser mostrada esta imagen tuvo que ser reducida en un 87% (500×450), esta imagen es mostrada en la figura 4.24).

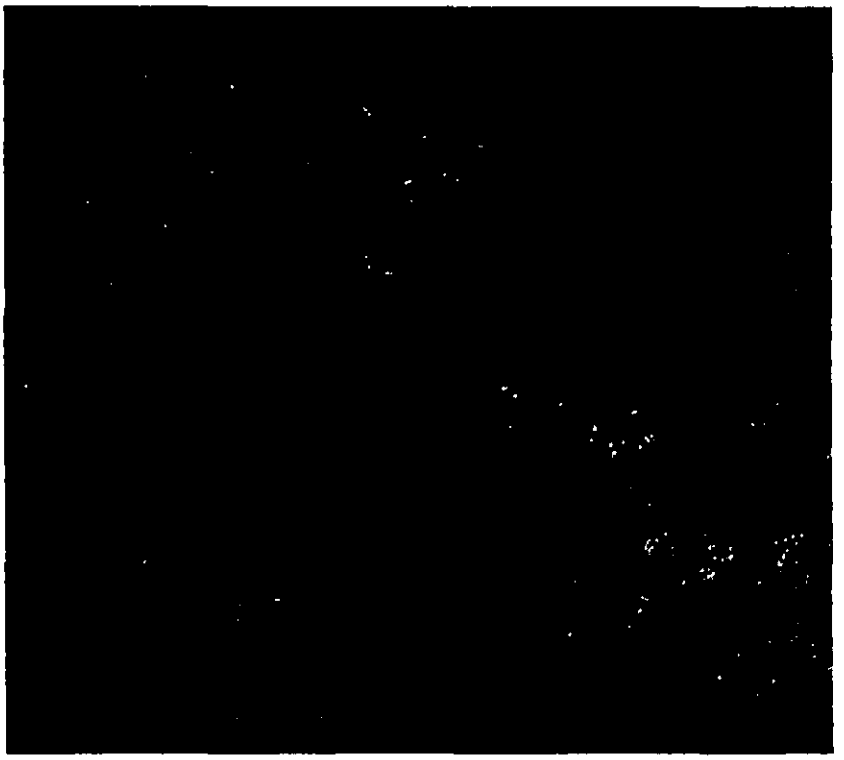

Figura 4.24: Descomprimida, $E_{rms} = 4.890450$, $PSNR = 22.518618$, $R_t = 0.375$

Comparando los resultados obtenidos de esta imagen con los obtenidos con la región de 1200 × 1200 de la misma banda, se puede apreciar que no tuvieron mucha variación (ni siquiera en una unidad) en las medidas de error, lo cual hace pensar que el comportamiento puede ser similar si se obtuvieran resultados con los demás parámetros. Uno de los factores importantes en la compresión de esta imagen es el tiempo. Como se puede notar en el cuadro correspondiente a la región de 1200 × 1200 de la banda 4, el proceso de compresión se llevo tan sólo 30 segundos, mientras que con la banda completa el proceso se tardó 335 segundos, lo cual lo hizo un factor importante y no muy práctico, ocasionando que la disminución de este tiempo sea un reto a futuro.[2]

Para finalizar se presenta a continuación como afectan las imágenes comprimidas a una tasa límite (100:1), en un proceso conocido como *clasificación de pixeles*. Este proceso regularmente es aplicado a imágenes multiespectrales y es realizado

[2]**Nota:** Los tiempos mencionados anteriormente fueron obtenidos en una supercomputadora gráfica Onyx de 6 procesadores MIPS R10000 a 194 MHz.

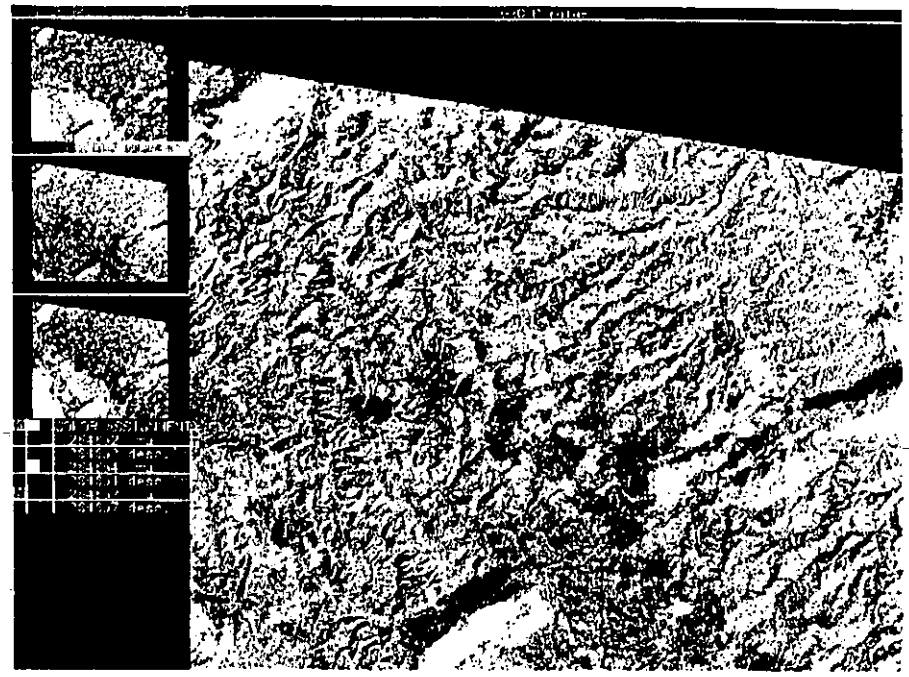

Figura 4.25: Imagen RGB, producida por GRASS, con los datos originales.

por gente que trabaja principalmente con sistemas de información geográficos (GIS). Para este análisis se consideraron las bandas completas (4000 × 3600). Cabe mencionar que el objetivo práctico o útil del trabajo con este tipo de imágenes es el de procesamiento de información y no puramente el de percepción visual. Por lo que el interés de los usuarios con esta técnica de compresión, es que el proceso no afecte la estructura de los datos para que los resultados de una clasificación de pixeles no cambie significativamente.

Para comenzar, se muestran a continuación dos imágenes generadas por un programa que es utilizado para el análisis de este tipo de imágenes, llamado *GRASS*. La primera (fig. 4.25) fue generada con los datos originales de las bandas 7,4 y 2. La segunda (fig. 4.26) fue generada con las mismas bandas pero con los datos obtenidos después de aplicar el proceso de descompresión a dichas bandas. El objetivo de mostrarlas es solamente para comparar la calidad perceptual de la imagen RGB generada con las bandas descomprimidas y la original.

A la resolución a la que son presentadas cualquiera diría que las imágenes son iguales, sin embargo al ser observadas en un monitor de alta resolución se podrá observar que el único defecto que tiene la imagen formada por los datos descomprimidos es la pérdida de nitidez de la imagen, ya que como se puede observar, las regiones establecidas por la combinación RGB de sus pixeles son exactamente las mismas, en las dos imágenes se pueden notar las mimas zonas coloreadas del mismo

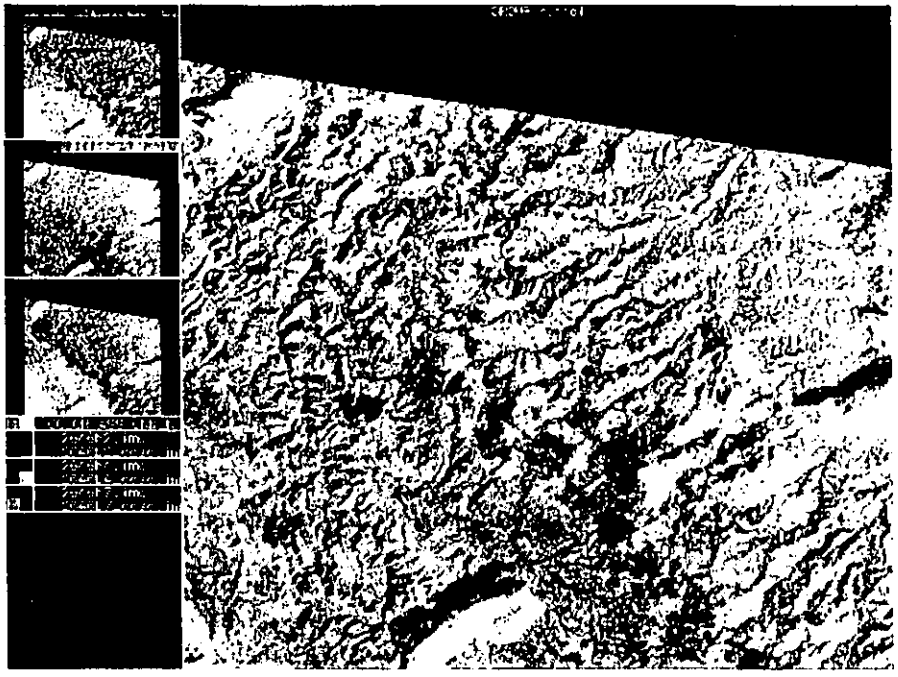

Figura 4.26: Imagen RGB, producida por GRASS, con los datos descomprimidos.

color, siendo la pérdida de detalle, la única diferencia que existe entre ellas.

Uno de los procesos más comunes (y quizá el más importante) que se realiza con imágenes de satélite, es el de clasificación de pixeles para encontrar la covertura vegetal (dependiendo del tipo de sensor, existen diferentes imágenes multiespectrales para clasificar vegetación, tipos de suelos, minerales, entre otros). Este es un proceso que consiste en tomar seis de las siete bandas, excluyendo la banda siete por ser la banda infrarroja, ya que está muy influenciada por la temperatura y otros aspectos como la hora del día y otros que afectan la clasificación. Al final de este proceso se obtiene una sola imagen en la que los pixeles están etiquetados con el tipo de vegetación o uso de suelo (suelos desnudos, selva, pastizales, zona urbana, etc.) que se encuentre en ese punto de muestreo. El proceso pasa de seis bandas con valores de pixeles entre 0 y 255 a una sola imagen con tantos valores como categorías de vegetación se consideren, por ejemplo, ocho si se considera que en ese sitio existen tal cantidad de tipos de vegetación.

La información que arroja es utilizada para conocer los recursos naturales de una zona y así, realizar planes de manejo y explotación o simplemente para inventariar los recursos. Este conocimiento se consigue con estrategias muy diversas, dependiendo de los objetivos partículares del estudio; sin embargo, un procedimiento típico y que generalmente constituye el arranque del análisis, es la clasificación no asistida

con *componentes principales y cluster*. Esto significa que el procedimiento es completamente automático y es considerado como un estándar en Ciencias de la Tierra; características muy convenientes para propósitos de comparación y evaluación.

El proceso consiste principalmente en transformar datos a través de un *Análisis de componentes principales* (PCA) para reducir la dimensión (en lugar de usar las seis bandas, se concentra la información en tres componentes, los cuales mantienen la mayor catidad de variación que proporcionan las imágenes originales). Posteriormente se obtienen firmas espectrales[3] mediante un análisis de conglomerados o "Cluster". Con esta información, se procede a etiquetar a cada pixel, como una de las clases (tipos de vegetación) a la que más se parece; esto se realiza con una técnica conocida como *máxima verosimilitud*.

Este procedimiento arroja una imagen simbólica, esto es, los valores de sus pixeles corresponden a cada una de las clases (generalmente alrededor de ocho) y están etiquetadas en una tabla con los tipos de vegetación correspondientes.

Para realizar una evaluación empírica de la pérdida de información ocasionada por el proceso de compresión, se siguió el procedimiento de clasificación no asistida antes descrito; utilizando para esto tanto las bandas originales como las descomprimidas, es decir, el mismo proceso será aplicado a ambos tipos (figura 4.27). Al comparar los tres primeros componentes principales (con los cuales se realiza el análisis de clasificación) se observa que se mantiene el mismo patrón y que la pérdida es muy pequeña; observar como en la figura 4.28, los círculos huecos (datos descomprimidos) están casi en la misma posición que los círculos sólidos (datos originales). Esto indica que aunque se han perdido datos por el proceso de compresión (además de la que ocasionó la transformación a componentes principales), se mantiene la información (reflejada en el patrón de los tres primeros componentes), originando así que se extraiga el mismo conocimiento.

Esta evaluación empírica indica que con una compresión de 100:1, es posible recuperar toda la información y lograr una imagen de covertura vegetal casi idéntica a la conseguida tras analizar los datos originales. Es de suponerse que con tasas de compresión menores, se mantengan los mismos resultados satisfactorios.

Hasta aquí llega la parte de análisis de resultados. El siguiente capítulo está dedicado a describir cómo funciona esta técnica, en un sencillo sistema cliente/servidor. Se explica de manera breve, cómo fue construído con la ayuda del lenguaje de programación Java y su planteamiento, para que sea utilizado como un **servidor de imágenes geográficas**.

[3]considerada como las características estadísticas de grupos de pixeles parecidos

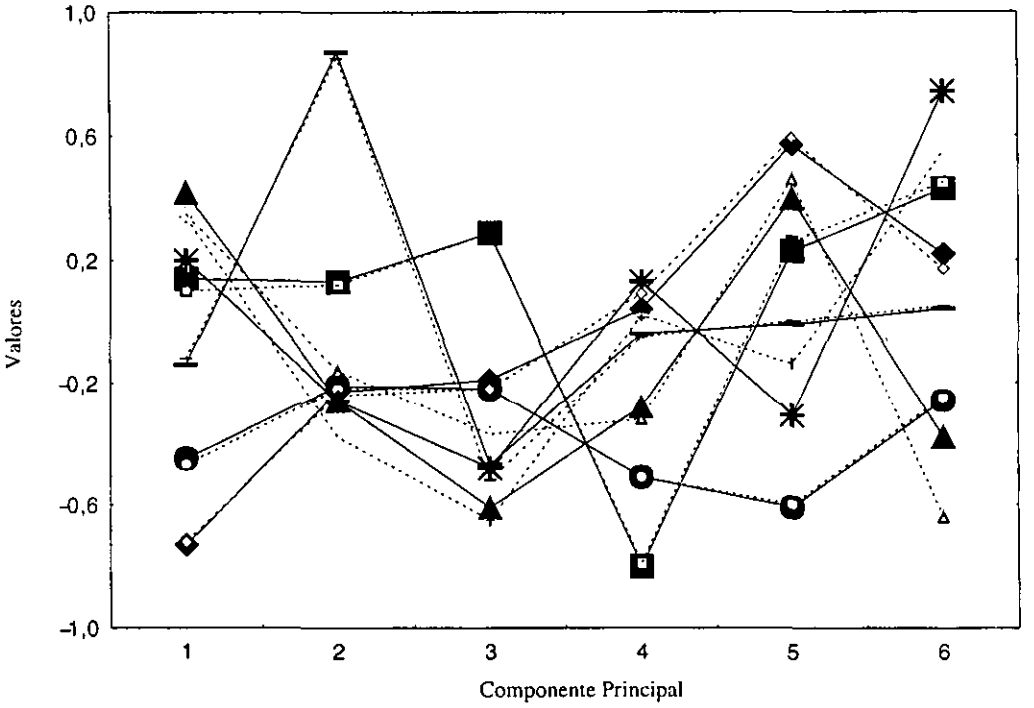

Figura 4.27: Comparación de los componentes de las bandas originales (líneas sólidas) y descomprimidas (líneas punteadas).

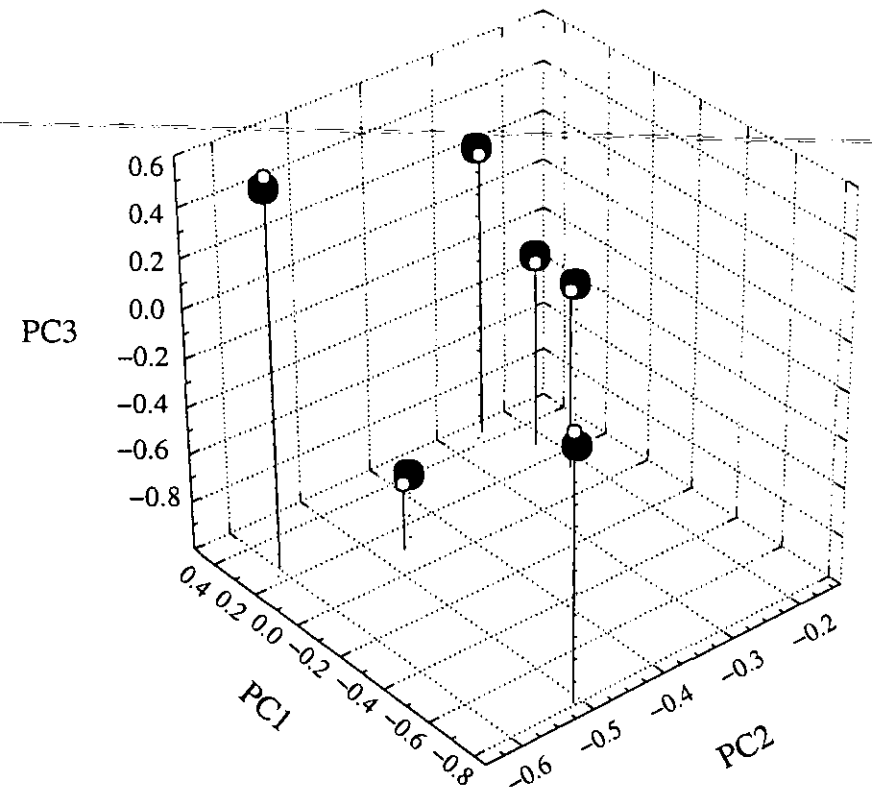

Figura 4.28: Análisis de componentes principales con los datos originales (círculos sólidos) y con las bandas descomprimidas (círculos huecos).

Capítulo 5

Aplicación

5.1 Introducción

Uno de los objetivos de tener una técnica de compresión es, "tener cierto control sobre los datos de una imagen", es decir, la técnica de compresión debe poder controlar la cantidad de pérdida que una imagen puede sufrir en el proceso de compresión. Este grado de pérdida afectará en qué tanto se reduce la cantidad de datos que representará a la imagen (archivo comprimido), esto es, mientras más pequeño sea este conjunto de datos, los valores de los pixeles de la imagen serán menos aproximados a los originales.

Considerando este factor, el cual permitirá que en un canal de transmisión la imagen viaje más rápido y con la ayuda de las redes de computadoras, se puede plantear un sistema que tenga la capacidad para que desde una máquina cliente, se pueda tener acceso a una máquina que cumpla la función de un servidor de imágenes. En éste la máquina servidor atenderá cada una de las peticiones que haga la máquina cliente, el cual principalmente proveerá de imágenes a dicha máquina. Este tipo de esquemas son conocidos como sistemas cliente/servidor.

Con los avances actuales en los lenguajes de programación, es posible hacer aplicaciones sencillas, que permitan la comunicación entre máquinas en una red. Tal es el caso del lenguaje de programación Java, el cual, a partir de su versión 1.1.6 permite el manejo de métodos para la comunicación remota entre máquinas, mediante RMI (Remote Method Invocation) o "invocación de métodos remotos".

En el siguiente tema se describe de forma más detallada el sistema cliente/servidor propuesto, como una aplicación del uso de la técnica de compresión.

5.2 Sistema cliente/servidor

En principio y sin entrar tanto en detalle, un sistema cliente/servidor es una aplicación que permite la comunicación de dos o más computadoras, las cuales pueden o no estar en una red. El propósito principal es tener una maquina que pueda proveer de servicios a otras que los requieran. En particular, se puede dar el caso de que estos servicios sean sólo para proporcionar archivos (como un servidor ftp).

De acuerdo a lo anterior fue propuesto un esquema que consta básicamente de un servidor que proporciona imágenes, tal y como lo pudiera hacer un servidor ftp, pero con la característica de que el servidor tiene la capacidad de realizar el proceso de compresión a una imagen para su posterior transporte en una red. La figura 5.1 muestra este esquema.

Se puede observar en la figura 5.1, las operaciones que en general tienen que hacer el servidor y el cliente. Cuando el programa cliente hace la petición de una imagen, el servidor atiende la petición y de acuerdo a unos parámetros de compresión previamente establecidos, se comprime la imagen, guardando los datos en un archivo temporal; el cual inmediatamente es transmitido por la red. Al llegar los datos de la imagen comprimida, el programa cliente ejecuta el proceso de descompresión y despliega la imagen en el monitor de la máquina cliente.

En el siguiente tema se describe cómo fue programado el sistema, utilizando los RMI de Java

5.2.1 Desarrollo en Java

El sistema fue desarrollado mediante una aplicación en Java. Para esto fue necesario utilizar los siguientes paquetes:

- El conjunto de clases base de Java
- Las clases RMI de la versión 1.2 de Java
- Para propósitos de la interfaz gráfica, las clases de Java Swing
- Para el manejo y despliegue de imágenes, las clases de JAI (Java Advanced Image)

La parte principal del sistema la componen las clases de RMI. Este conjunto de clases han sido implementadas principalmente para programar aplicaciones distribuidas. Este tipo de aplicaciones se basa en establecer comunicación entre varias máquinas con el propósito de intercambiar recursos.

De acuerdo a la parte de redes en el capítulo 2, se mencionó que para establecer comunicación entre dos máquinas, regularmente se hace uso de "sockets", los cuales

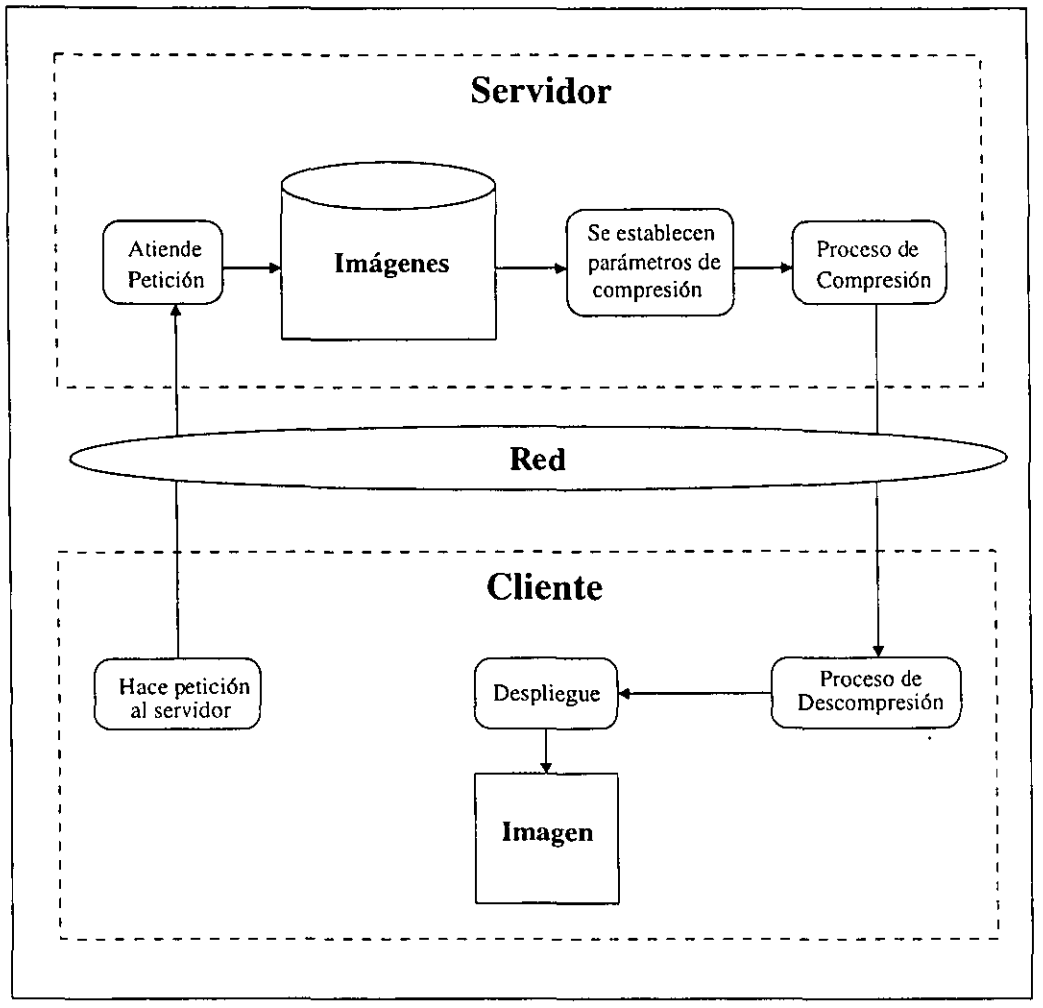

Figura 5.1: Esquema cliente/servidor

pueden ser vistos como una especie de puertos de entrada/salida y mediante éstos, una maquina podía establecer una conexión con otra, para interactuar entre ellas. Con el uso de RMI, no es necesario establecer conexiones con sockets, ya que éstos están programados de manera que su uso sea transparente para el programador, lo cual facilita en gran medida la aplicación, ya que el programar haciendo uso de sockets no es una tarea fácil.

Formalmente el paquete RMI es un mecanismo que habilita a un objeto para invocar métodos sobre otro objeto que esté corriendo en una máquina virtual de Java diferente a la del primero; en muchas ocasiones estos objetos son referidos como "objetos remotos". En el ámbito de programación se dice que un objeto remoto es una instancia de una clase que implementa una "interfaz remota". La interfaz

remota debe declarar cada uno de los métodos que serán llamados desde diferentes máquinas virtuales de Java.

Para programar aplicaciones con RMI, es necesario fijar una serie de reglas, las cuales no serán descritas a detalle en este texto, pero pueden ser consultadas a través de la ayuda en línea en las páginas oficiales de Java a través de internet o en [8].

El esquema de la implementación del sistema se muestra en la figura 5.2.

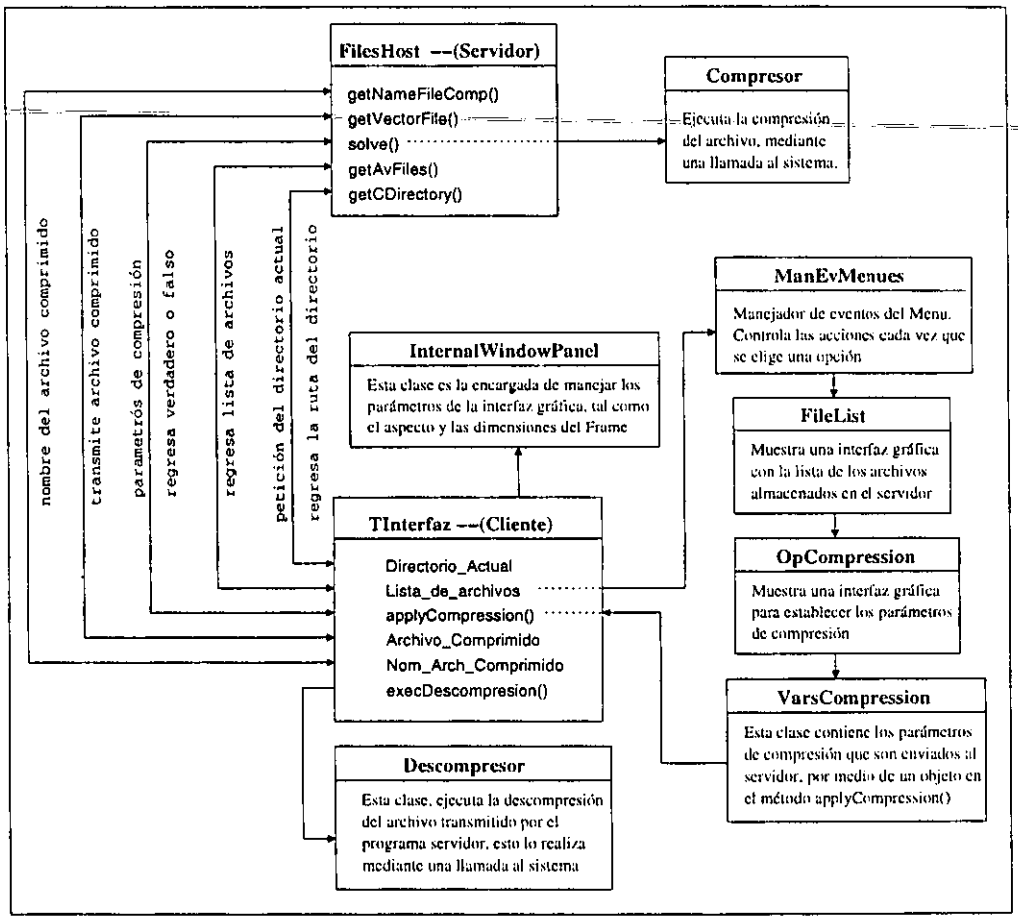

Figura 5.2: Esquema del sistema.

La figura 5.2 muestra básicamente las clases empleadas en el sistema. En los cuadros se observan los nombres de las clases y debajo de éstos, se dan algunos nombres de los métodos o una descripción de como interactua la clase en el sistema. Mediante las flechas se puede notar la forma en que se comunican los objetos de cada clase, ya sea por métodos o por instancias de una a otra. A continuación se presenta una breve explicación de como funciona el sistema.

5.3 Funcionamiento del sistema

El uso principal del sistema está enfocado a tener un servidor de imágenes. Como se vió en capítulos anteriores, las imágenes geográficas requieren de gran capacidad de almacenamiento, lo cual hace que el manejo de éstas sea impráctico, en aplicaciones en donde sea necesario transportarlas. Cuando el uso de éstas se hace en una red de computadoras, hay que tomar en cuenta factores como el tráfico en la red, ancho de banda del canal de comunicaciones y en algunas ocasiones la velocidad de procesamiento de las máquinas, esto último porque si se está considerando que las imágenes van a ser transportadas de manera comprimida, la máquina que recibe ese archivo idealmente se requiere que tenga rápidez de cómputo para ejecutar el proceso de descompresión del archivo y mostrar la imagen en pantalla, aunque claro, estos requerimientos no son indispensables.

Para describir el funcionamiento del sistema, se va a suponer que se quiere hacer el traslado de una imagen, ubicada en el servidor, a una máquina cliente.

Suponiendo que ya está corriendo el servidor de imágenes, al abrir el programa cliente, trata de hacer la conexión con el servidor, en caso de no poder hacerlo el programa lanza una serie de excepciones indicando el motivo por el cual no pudo hacer la conexión. En caso de poder establecerla, el programa desplegará una interfaz gráfica. En esta interfaz se muestra un menú, en donde una de las opciones permite elegir una imagen (archivo) que se encuentre en la máquina servidor. Al ser elegida la imagen, se presenta un cuadro de diálogo para establecer los parámetros de compresión que sufrirá la imagen. Al ser establecidos, el servidor se encarga de realizar el proceso de compresión para que posteriormente se ejecute el traslado de la imagen comprimida. Cuando la imagen ha llegado por completo a la máquina cliente, ésta ejecutará el proceso de descompresión sobre los datos, para posteriormente ser visualizada en la pantalla.

En el apéndice A es mostrado un pequeño manual de usuario que indica los pasos que hay que seguir para correr la aplicación.

Algunas de las posibles mejoras de este sistema pueden ser principalmente hacerlo más amigable, es decir, la aplicación generada solo considera la conexión a una sola máquina (servidor), la cual no puede ser modificada, esto sencillamente se puede resolver agregando un cuadro de dialogo al programa cliente, para que el usuario pueda determinar la direccion IP o el DNS de la máquina remota. Otra de las consideraciones importantes es hacer el manejo de posibles errores que salgan dentro de la interfaz (excepciones), como cuando se suscitan problemas en la conexión o en algún momento dentro de la ejecución, los mensajes de error en este momento son mandados a pantalla y no son considerados como parte del uso de la interfaz. El manejo de estos hará que el usuario identifique y corriga posibles errores que se hayan generado durante el uso de la aplicación.

5.4 Conclusiones

El objetivo de contar con la implementación de una técnica de compresión fue alcanzado. Se logró conjuntar un esquema típico de compresión para ser aplicado y adaptado a requerimientos de un tipo específico de imágenes; esto es, de acuerdo a los resultados obtenidos, se observó que con ciertas consideraciones, la técnica pudo ser adaptada con mejores resultados a las imágenes geográficas que ser aplicada a todo tipo de imágenes (conocidas como "escenas naturales").

Las conclusiones con respecto a la técnica de compresión son las siguientes:

- La técnica de compresión permite establecer parámetros variables para obtener distintas tasas de compresión.
- La tasa mínima que soporta la técnica es 2:1 y es alcanzada estableciendo un nivel para la transformada wavelet y eligiendo vectores de 2×2 para cuantificar los coeficientes detalle de la transformada wavelet (subimágenes con orientación horizontal, vertical y diagonal).
- De acuerdo a los resultados obtenidos se establece que no es conveniente aplicar más de 4 niveles de transformación para comprimir la imagen, debido a la distorsión en los bordes generados en la imagen descomprimida.
- Debido a la descomposición de la imagen con la transformada wavelet, se establece que es conveniente cuantificar las subimágenes de los niveles más altos con vectores de dimensiones pequeñas (2×2, 4×4), ya que esto permite conservar la mayor cantidad de coeficientes detalle más importantes para aumentar la calidad visual de la imagen reconstruida (descomprimida).
- Se comprobó que omitiendo la subimagen con orientación diagonal en el proceso de compresión aumenta la tasa de compresión y las medidas de error no varían significativamente con respecto a las obtenidas cuando se considera esa subimagen.
- La tasa máxima promedio alcanzada es 100:1 para imágenes de dimensión mayor a 512×512, en donde al comparar la calidad visual con imágenes generadas con el estándar JPEG a la misma tasa, se notaron las deficiencias que tienen los dos esquemas y que en algunos casos, como en las bandas de una imagen multiespectral, fue mejor la calidad visual de la imagen generada con la técnica que usa la transformada wavelet que la generada con el JPEG.
- Se determinó que es viable el uso de la técnica de compresión en imágenes multiespectrales, debido a los resultados obtenidos en un análisis de componentes principales.
- Debido a los tiempos alcanzados en el proceso de compresión a una banda espectral a una tasa límite máxima, se concluye que hay que hacer más eficiente

los algoritmos tanto de transformación como de cuantificación para eficientar el desempeño y reducir los costos de cómputo requeridos para estos procesos.

En la implementación del esquema de compresión se aportaron varias consideraciones en el esquema final, como son el uso de un método equivalente de cuantificación al usado originalmente en la propuesta [2] y la consideración del signo de los coeficientes de la subimagen paso bajas de la transfomada wavelet.

Otra de las aportaciones importantes fue el estudio de las imágenes multiespectrales reconstruidas (descomprimidas), aplicadas a un análisis de componentes principales para observar en que grado afecta el proceso de compresión considerando una tasa alta (100:1) con respecto al mismo análisis realizado con los datos originales; donde se pudo notar que el proceso de compresión no afectó en gran medida y que por tanto es viable aplicar una compresión a esa tasa, sin que se incremente el error en la clasificación de los pixeles; obteniendo los mismos resultados finales del análisis.

Así mismo, se propuso un esquema básico de una aplicación que involucrara el uso de un servidor de imágenes, para acceder a éstas desde una máquina cliente.

El estudio de esta técnica permitió conocer las bases y diversos usos que tiene esta importante herramienta conocida como la *Tansformada Wavelet*, la cual surgió como otra alternativa para utilizarse en diversas áreas del procesamiento de imágenes. La experiencia de haberla usado deja una serie de retos: tratar de realizar algoritmos más eficientes para reducir el espacio de memoria requerido para obtenerla y otro puede ser implementar el algoritmo en paralelo, ya que la misma forma en la que funciona lo permite, es decir, el algoritmo de la transformada es considerado como un algoritmo altamemente paralelizable [16]. También se puede explorar el uso de esta transformada en 3 dimensiones, aplicándolas por ejemplo, a todo el conjunto de bandas de una imagen multiespectral. Usando el concepto de multiresolución se pueden plantear esquemas de transmisión progresiva, no solamente a imágenes sino también a datos en general para su uso en aplicaciones en visualización científica.

Por el lado de la cuantificación vectorial se ha demostrado que este método puede ser tan eficiente como lo sean sus algoritmos de búsqueda y la manera en la que se genera el conjunto de vectores representativos (codebook's), para lo cual también la experiencia de este trabajo deja una serie de retos que van más por el lado de la programación del algoritmo de cuantificación, ya que este puede ser realizado por diversas técnicas como: redes neuronales o métodos autoadaptables y con las cuales se podría investigar cual será la mejor manera de adaptarse a una transformada wavelet.

También el uso de la codificación Huffman podría ser reemplazado por uno de los métodos más usados en la actualidad, este es conocido como *codificación aritmética*. Para lo cual se tendrá primero que conocer sus bases teóricas y después ver la manera de adaptarlo al esquema de codificación.

El conocimiento de este esquema de compresión permitió observar algunas de las deficiencias que tiene el estándar de compresión JPG, motivo por el cual y después de revisar muchas propuestas, el grupo de investigadores en este campo, optó por utilizar un esquema de descomposición con la transformada wavelet, dando como producto un nuevo estándar aprobado y que tendrá el nombre de JPEG2000.

En la aplicación de la técnica se notó que una de las ventajas de tener este tipo de aplicaciones en Java, es poder interactuar con software ya desarrollado, es decir, existe software programado en Java al que se le pueden agregar nuevas aplicaciones a manera de "plug–in's", tal es el caso de una aplicación conocida como "ImageJ" (la cual fue utilizada para generar las imágenes RGB mostradas con las regiones de 1200 × 1200 de la bandas multiespectrales). Esta aplicación es usada básicamente para procesamiento de imágenes, ya que tiene incluidos varios algoritmos y operadores para trabajar con imágenes. Esta aplicación permite que se le puedan insertar nuevos módulos para que funcionen dentro del mismo, de esta forma, la aplicación previamente mostrada puede ser añadida a dicho programa en determinado momento, con el propósito de tener un sistema más completo y poder hacer cierto procesamiento a las imágenes que se transmiten.

En resumen, este trabajo de investigación aportó resultados a una de las primeras propuestas publicadas, para usar la transformada wavelet en un esquema de compresión, adaptándola para aplicarse a imágenes de tipo goegráficas. Se lograron resultados satisfactorios para su uso y se propuso el funcionamiento de un pequeño sistema que trabaja de manera remota, haciendo uso de la técnica de compresión y del envío de datos en una red.

Apéndice A

Uso del sistema

Primero que nada hay que compilar los archivos .java en la máquina cliente y en la máquina servidor, después es necesario generar algunos archivos llamados "skeleton" y "stub", los cuales forman una interfaz para comunicar los objetos remotos. El "skeleton" es usado para crear nuevas instancias de los objetos y "rutear" las llamadas de los métodos remotos. El "stub" es usado por el cliente para rutear las invocaciones de los métodos al objeto que se encuentra en el servidor. Para crear estos archivos es necesario utilizar un programa que viene con el compilador de Java, llamado rmic [8].

Como este esquema se basa en una conexion remota, hay que utilizar un archivo llamado "policy", el cual contiene los permisos para usar algunos de los servicios, ya sea de la máquina local o de la máquina remota. Esta serie de servicios pueden ser: el acceso al sistema de archivos, permisos de lectura y escritura, etc. Como este sistema solo pretende mostrar una aplicación de ejemplo, este archivo policy tiene indicado que no tiene restricción de permisos. Claro está que en un sistema más robusto, este archivo "policy" debe contemplar todo tipo de restricciones referentes a la seguridad del sistema.

Después de compilar y generar los archivos necesarios para establecer la conexión remota, hay que correr un programa llamado rmiregistry. Este programa sirve como manejador de objetos remotos para sistemas que usan objetos distribuídos[8]. Este programa debe correr en la máquina que sirve como servidor, por tanto el programa tiene que ser invocado de la siguiente manera:

% rmiregistry &

En seguida puede ser corrido el programa servidor, de la siguiente manera:

```
% java -Djava.security.policy = ./policy FilesHostImpl
```

En seguida empezará el programa servidor a registrar los objetos y a actualizar sus variables, en donde al final se mostrará el mensaje:

```
Servidor remoto listo ...\\
```

De esta forma el servidor estará esperando cualquier petición del programa cliente. Como el sistema se tendrá que comunicar en ambas direcciones, es decir, tanto el programa cliente pide datos al programa servidor como éste lo hace al cliente, entonces también en la máquina cliente, se tendrá que correr el programa rmiregistry y hacer uso del archivo "policy". Para correr el programa cliente es necesario teclear desde la máquina cliente:

```
% java -Djava.security.policy=./policy Tinterfaz
```

Como en este caso se está planteando que una máquina en particular haga la función de servidor, no será necesario indicarle a que máquina se tendrá que conectar, sino que en el programa esa opción es fija, por tanto al correr el cliente, aparecerá la siguiente información:

```
Conectando a rmi://ursa.labvis.unam.mx/TheChooser...
```

En donde ursa.labvis.unam.mx, es la máquina a la cual se está haciendo la conexión. La palabra "TheChooser" solamente indica el nombre del objeto remoto. Si la conexión fue existosa, se presentará una interfaz gráfica como la mostrada en la figura A.1.

Para desplegar una imagen que se encuentra en la máquina servidor, se le puede dar click en el menu *imagen*, ahí hay que elegir la opción "Remota". Esta opción va a desplegar un cuadro de diálogo para elegir el nombre de la imagen que se quiere visualizar (figura A.2). Al elegir una imagen y darle aceptar se mostrará otro cuadro de diálogo para especificar los parámetros de compresión del archivo (figura A.3).

Después de establecer las opciones, se mandan todos los datos necesarios para que la máquina servidor realice el proceso de compresión de la imagen elegida y la transmita comprimida a la máquina cliente. Cuando el archivo se haya transmitido por completo, el programa cliente empezará a realizar el proceso de descompresión, para posteriormente mostrarla en pantalla (figura A.4). Ya que es mostrada en pantalla, también aparecen algunos datos de la reconstrucción de la imagen, tales como su E_{rms}, el $PSNR$ y el tiempo de decodificación (figura A.5).

Como se puede observar, este sistema no considera muchos factores, lo cual no lo hace un sistema muy complejo. El propósito de mostrarlo fue únicamente para ejemplificar una de tantas aplicaciónes que podría tener la técnica de compresión.

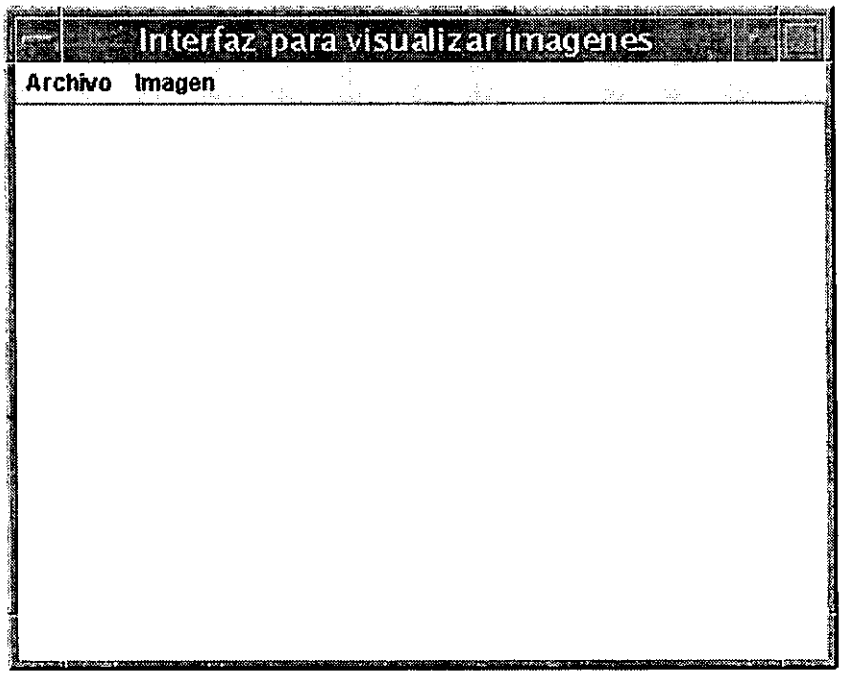

Figura A.1: Pantalla de la interfaz gráfica.

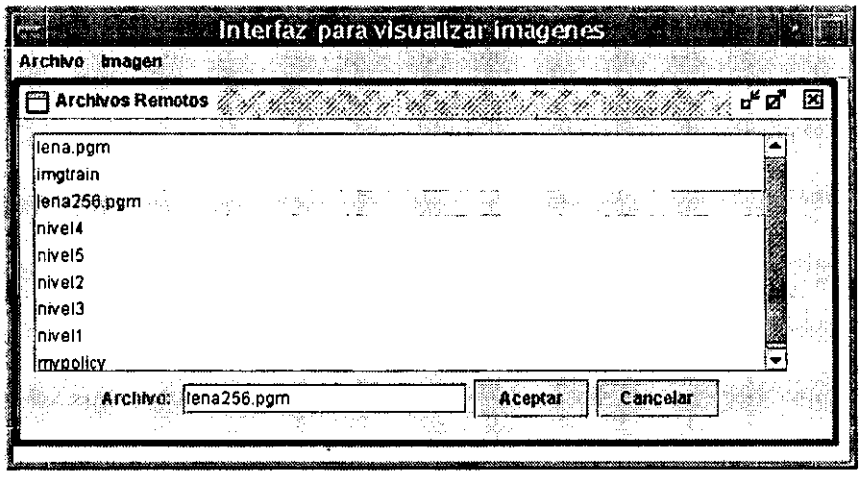

Figura A.2: Lista de imágenes del servidor.

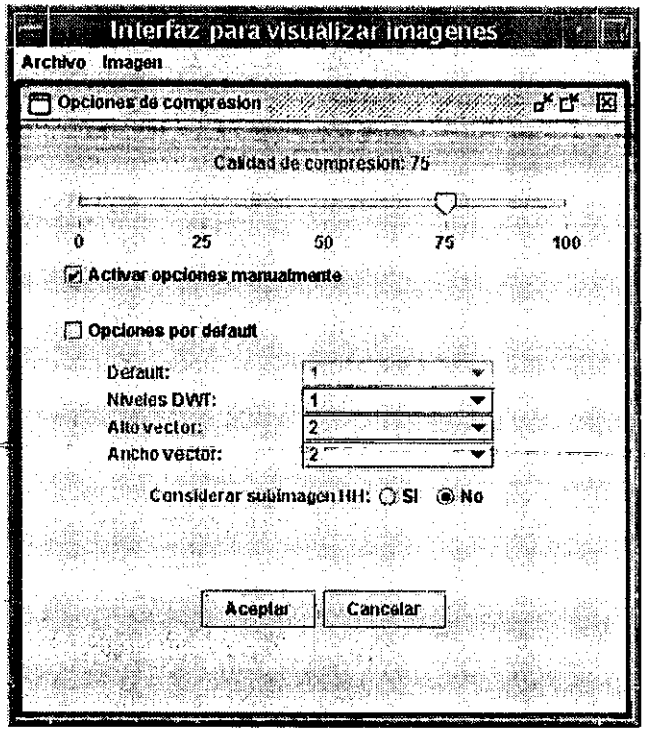

Figura A.3: Cuadro de diálogo para establecer parámetros de compresión.

Figura A.4: Imagen descomprimida, después de ser transportada.

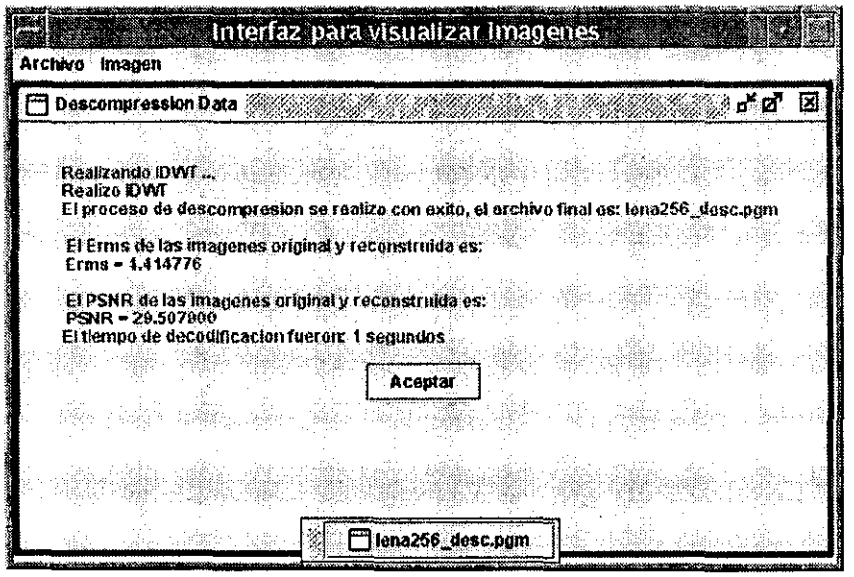

Figura A.5: Resultados mostrados después de realizar el proceso de descompresión.

Bibliografía

[1] John G. Ackenhusen. *Signal Processing Technology and Applications.* IEEE Technical Activities Board, Piscataway, NJ, 1995.

[2] Marc Antonini, Michel Barlaud, Ingrid Daubechies, and Mathieu Pierre. Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 1(2):205–220, April 1992.

[3] Michel Barlaud. *Wavelets in Image Communication.* Elsevier, Netherlands, 1994.

[4] Peter Burton and Edward Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Image Communication*, 31(4), April 1983.

[5] R Castleman Kenneth. *Digital Image Processing.* Prentice Hall, 1996.

[6] A. Chellapa, Rama y Sawchuk Alexander. *Digital Image Processing and Analysis.* IEEE Computer Society, 1 edition, 1985.

[7] Microsoft Corporation. *Networking Essentials.* Redmon, 1996.

[8] Jim Farley. *Java Distributed Computing.* O'Reilly, California, USA, 1st. edition, 1998.

[9] Allen Gersho and Robert M. Gray. *Vector quantization and signal compression.* Kluwer academic, Boston, 1992.

[10] Rafael Gonzalez. *Tratamiento Digital de Imágenes.* Addison–Wesley, Mexico, 1996.

[11] Gilbert Held. *Data Compression: Techniques and applications, hardware and software considerations.* J. Wiley, Chichester, 1987.

[12] Hewi Piao Hsu. *Análisis de Fourier.* Addison-Wesley, México, 1987.

[13] C.M. Huang, Q. Bi, S. Stiles, and Harris R.W. Fast full search equivalent algorithms for image compression using vector quantization. *IEEE Transactions on Image Processing*, 1(3):413–416, July 1992.

[14] Andrew Inglis and Arch Luther. *Video Enginnering.* McGraw Hill, New York, 1996.

[15] K. Jain Anil. *Fundamentals of Digital Image Processing.* Prentice Hall, Englewood cliffs, New Jersey, 1989.

[16] Bo Kågström, Jack Dongarra, Erik Elmroth, and Jerzy Waśniewski. *Applied Parallel Computing: Large Scale Scientific and Industrial Problems.* Springer, Germany, 1998.

[17] Jae S. Lim. *Two-dimensional signal and image processing.* Prentice Hall, Englewood cliffs, New Jersey, 1990.

[18] Stephane G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Transactions on Patern Analysis and Machine Intelligence*, II(7), July 1989.

[19] Yves Mayer. *Wavelets, Algorithms & Applications.* SIAM, Philadelphia, 1993.

[20] Nasser M. Nasrabadi and Robert A. King. Image coding using vector quantization: A review. *IEEE Transactions on Communications*, 36(8):957–971, August 1988.

[21] Gregory M Nielson and Hans Hagen. *Scientific Visualization.* 1997.

[22] Timothy Parker. *Aprendiendo TCP/IP en 14 días.* Prenntice Hall, 1995.

[23] Wiiliam B. Pennebaker. *JPEG Still image compression standars.* Van Nostrand Reinhold, New York, 1993.

[24] Larry Peterson. *Computer Networks a System Approach.* Morgan Kauffman, 2 edition, 1996.

[25] Majid Rabbani and Jones Paul W. *Digital Image Compression.* Spie optical enginnering, Bellingham Washington, 1991.

[26] T.A. Ramstad and S.O. Aase. *Subband Compression of images: Principles and Examples.* Elsevier science, Amsterdam, The Netherlands, 1995.

[27] Azriel Rosenfield and Kak Avinash. *Digital Picture Processing.* Academic, Orlando, 1982.

[28] Erick Stollnitz, D. Tony DeRose, and H. David Salesin. *Wavelets for Computer Graphics.* Morgan Kauffman, San Francisco, California, 1996.

[29] James.A. Storer and Martin. Cohn. *Proceedings, Data Compression Conference.* IEEE Computer Society, Los Alamitos, California, 1 edition, 2000.

[30] Gilbert Strang and Troung Nguyen. *Wavelets and Filter Banks.* Wellesley–Cambridge Press., 1997.

[31] Andrew Tanenbaum. *Redes de Operadores.* Prentice Hall, 1991.

[32] Brani Vidakovic and Peter Müller. Wavelets for kids. Technical report, University of Duke, 1991.