JPEG CON PARTICIONAMIENTO ADAPTIVO QUADTREE. ANALISIS DE LA PÉRDIDA

Analista Raúl Alberto Flores

Director : Ing. Armando De Giusti

Co-Director : Lic. Claudia C. Russo

Resumen

En este trabajo se presentan y comparan dos técnicas de compresión de imágenes con pérdida basadas en el método usado por el Standard de compresión Joint Photographic Experts Groups (JPEG) y en el método de particionamiento adaptivo Quadtree propuesto, estudiando en particular la pérdida producida.

Como medidas de perfomance se estudiarán el PSNR y SNR y para el análisis de la pérdida el EMS.

Se analizará el algoritmo JPEG baseline y se propondrán optimizaciones que pongan énfasis en la cuantificación y que no modifiquen las restricciones del standard.

Se analizarán los resultados obtenidos de la generalización en dos imágenes con características muy diferentes: LENA y VLSI; y se considerarán los efectos del tamaño de los bloques en áreas conflictivas como los bordes y en aquellas cuya extensión es muy amplia y con la característica de poseer variaciones poco perceptibles en los niveles de grises.

La alteración en el brillo de la imágen descomprimida podrá apreciarse en la visualización de los histogramas correspondientes.

Palabras Claves: Imágenes, Compresión, JPEG, Cuantificación, Quadtree, Adaptivo, Pérdida

Introducción

En términos generales el ***procesamiento de imágenes*** involucra manipulación y análisis de información gráfica, que en nuestro caso se trata de imágenes visuales bidimensionales. Cualquier operación que se aplique sobre una imagen para mejorarla, corregirla, analizarla o de alguna manera cambiarla es un procesamiento de imagen.

Así como hay operaciones que pueden mejorar la calidad de una imagen hay otras que pueden extraer automáticamente información de ellas. En todos los casos se aplica una técnica digital a una imágen digital para obtener un resultado digital, como una nueva imágen o una lista de datos extraídos.

Las operaciones de procesamiento de imágenes digitales pueden agruparse en cinco clases fundamentales, cada una con operaciones específicas:

Realce :

Las operaciones de realce mejoran las características de una imágen. Generalmente se alcanzan como un resultado final o como un proceso previo a alguna otra operación.

Pueden ser utilizadas para mejorar el contraste y brillo de una imágen, reducir su contenido de ruido, o resaltar sus detalles.

Lo que es considerado una mejora en una imágen es generalmente subjetivo y depende tanto de la aplicación como de la opinión del observador. Por ejemplo una imágen podría requerir un balanceo de su contraste para hacer su apariencia más placentera a un observador humano mientras que otra aplicación requiera para la misma imágen un incremento dramático del mismo, de manera que sus características específicas sean resaltadas para un posterior análisis automatizado.

Restauración :

Las operaciones de restauración, al igual que las de realce también mejoran la calidad de la imágen. Sin embargo este tipo de operaciones es estrictamente objetivo, ya que se basa en medidas y degradaciones conocidas de la imágen original.

Usualmente las imágenes a las que se le puede aplicar este tipo de operaciones han sufrido algún tipo de degradación en el sistema de formación de la imágen por ejemplo una cámara fotográfica o de video.

Análisis :

Las operaciones de análisis generalmente no producen como resultado una imágen, en su lugar producen información numérica o gráfica basadas en las características de la imágen original. Dividen la imágen en objetos discretos y luego los clasifican usando algún proceso de medición.

Compresión

La operación de compresión de una imágen, tema que engloba a este trabajo, reduce el contenido de los datos a los necesarios para describirla.

El objetivo de la compresión de imágenes es la eliminación de la información redundante, contenida en abundancia por la mayoría de las imágenes.

Existen dos formas generales de compresión de imágenes. En primer lugar tenemos las técnicas de compresión de imágenes sin pérdida, las cuales conservan los datos de la imágen original. La otra forma consiste de técnicas de compresión con pérdida que no reconstruyen exactamente los datos de la imágen original, pero tratan de mantener un cierto nivel de calidad subjetivo.

Síntesis :

Estas operaciones crean imágenes de otras imágenes y de datos que no lo son. Son utilizadas cuando una imágen deseada es imposible de adquirir o no existe en alguna forma física. Se utilizan en tomografías computadas, resonancia magnética, CADS etc.

Se hará incapié en la etapa de compresión.

Compresión y descompresión sin pérdida

Esta técnica es utilizada cuando los datos deberán ser recuperados exactamente, y puede ser aplicada a imágenes médicas y de otras fuentes científicas. También es usualmente aplicada cuando a una imágen se le aplicará posteriormente técnicas de realce o restauración.

Compresión y descompresión con pérdida

Este tipo de compresión se aplica cuando la calidad de la imágen reconstruida debe ser mantenida en un cierto nivel pero no necesariamente ser idéntica a la original. Esto significa que solo es importante la calidad subjetiva o como el observador ve la calidad de la imágen. Imágenes apropiadas para este tipo de compresión incluyen secuencias de video, videoconferencia, imágenes fijas, y algunas aplicaciones de impresión.

JPEG es uno de los standards más conocidos para compresión de imágenes con pérdida. Se obtiene buen radio de compresión al aplicarlo sobre fotografías, trabajos de arte y material similar, aunque no ocurre así con textos, dibujos simples o líneas. JPEG explota las limitaciones del sistema visual humano [Cla95] [Nels91].

JPEG define tres sistemas de codificación diferentes:

Un sistema de codificación baseline con pérdida, que utiliza como base la Transformada del Coseno Discreta (DCT).

Un sistema de codificación extendido para aplicaciones con requerimientos de más precisión, de reconstrucción progresiva, etc.

Un sistema de codificación independiente sin pérdida para compresión reversible.

Para ser compatible con JPEG, cualquier producto debe incluir soporte para el sistema baseline. El standard propone una sintaxis que debería cumplir cualquier secuencia de bits para ser JPEG, dejando amplias libertades en las etapas de cuantificación y codificación, de manera de poder efectuar mejoras y optimizaciones. No se especifica ningún formato de imagen, resolución espacial o particularidades sobre el color. Una propiedad útil del método es que el grado de pérdida puede ser variado, ajustando parámetros de compresión, que se explicarán mas adelante.

Descripción del algoritmo JPEG baseline

El sistema recomendado por JPEG es una modificación al propuesto por Chen y Pratt y se basa en la técnica de codificación utilizando la transformada del coseno discreta (DCT). El proceso de compresión se realiza en tres pasos secuenciales [Say96]:

Transformación: Se realiza una conversión del dominio espacial al dominio frecuencial para lo cual se utiliza la transformada DCT.

Cuantificación: Utiliza cuantificación MIDTREAD uniforme para cuantificar los coeficientes de un bloque transformado.

Codificación: Los coeficientes resultantes de la cuantificación son codificados utilizando códigos de longitud variable.

A continuación se verán con mas detalle cada una de estas partes.

Transformación:

La transformada utilizada por el esquema JPEG es la transformada DCT ya mencionada, para la cual los valores de los pixels de la imágen de entrada son decrementados en $2^{(P-1)}$, donde P es el número de bits utilizados para representar cada pixel. En nuestro caso se utilizan imágenes de 8 bits donde los pixels toman valores entre 0 y 255 de manera que al substraerles 128 sus valores finales de entrada variarán entre -128 y 127.

El algoritmo comienza con el particionamiento de la imágen en bloques de 8x8 pixels. Si alguna de las dimensiones de la imágen no tiene un valor múltiplo de ocho el codificador replica la última columna y/o fila hasta que el tamaño final sea múltiplo de 8. Esas filas y columnas adicionales son removidas durante el proceso de decodificación. Luego cada bloque es transformado independientemente usando la Transformada del Coseno Discreta (DCT) descripta en las ecuaciones de más abajo dando como resultado otra matriz de 8x8 pero donde sus componentes o coeficientes varían en el rango de -1023 .. 1023. Un coeficiente DC resultante de una transformación es el valor más importante y se encuentra en la posición (0,0) de la matriz transformada. Su importancia se debe a que representa el coeficiente de la frecuencia mas baja, la cual puede caracterizar en forma general a una imágen.

Los coeficientes correspondientes a las restantes posiciones de la matriz se denominan AC y son los valores que sirven para aumentar en forma progresiva los detalles de la imagen.

Fórmulas de la transformada DCT :

$$F(u,v) = (1/\sqrt{2N})\,C(u)\,C(v)\sum_{i=0}^{7}\sum_{j=0}^{7} f(i,j)\cos\left[(2i+1)u.\pi\ /(2N)\right]\cos\left[(2j+1)v.\pi\ /(2N)\right]$$

$$f(i,j) = (1/\sqrt{2N})\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)\,C(v)F(u,v)\cos\left[(2i+1)u.\pi\ /(2N)\right]\cos\left[(2j+1)v.\pi\ /(2N)\right]$$

donde

$C(u) = 1/\sqrt{2}$ para $u = 0$ en otro caso $C(u) = 1$

$C(v) = 1/\sqrt{2}$ para $v = 0$ en otro caso $C(v) = 1$

Si tomamos por ejemplo el bloque de 8x8 de pixels de la tabla 1, substraemos 128 a cada uno de sus valores y aplicamos la transformada DCT a la matriz resultante obtenemos los coeficientes DCT de la tabla 2. En esta última puede notarse que los coeficientes de más baja frecuencia cercanos al vértice superior izquierdo toman valores absolutos más grandes que los coeficientes de más alta frecuencia. Esto es lo que generalmente ocurre, ya que existen excepciones como en los casos en que se tiene gran cantidad de variaciones en el bloque (ej. bordes).

Tabla 1 Bloque de 8x8 de la imágen LENA

124	125	122	120	122	119	117	118
121	121	120	119	119	120	120	118
126	124	123	122	121	121	120	110
124	124	125	125	126	125	124	124
127	127	128	129	130	128	127	125
143	142	143	142	140	139	139	139
150	148	152	152	152	152	150	151
156	159	158	155	158	158	157	156

Tabla 2 Coeficientes DCT correspondientes al bloque de pixels de la imágen LENA

39.88	6.56	-2.24	1.22	-0.3	-1.08	0.79	1.13
-102.43	4.56	2.26	1.12	0.35	-0.63	-1.05	-0.48
37.77	1.31	1.77	0.25	-1.5	-2.21	-0.10	0.23
-5.67	2.24	-1.32	-0.81	1.41	0.22	-0.13	0.17
-3.37	-0.74	-175	0.77	-0.62	-2.65	-1.30	0.76
5.98	-0.13	-0.45	-0.77	1.99	-0.26	1.46	0.00
3.97	5.52	2.39	-0.55	-0.051	-0.84	-0.52	-0.13
-3.43	0.51	-1.07	0.87	0.96	0.09	0.33	0.01

Cuantificación:

El algoritmo JPEG utiliza un tipo de cuantificación llamado cuantificación midtread uniforme para cuantificar los diversos coeficientes de un bloque transformado.

Las longitudes de los intervalos de cuantificación están organizados en una tabla de cuantificación y pueden ser vistos como parte fija de esta etapa del proceso. Un ejemplo de tabla de cuantificación recomendada por JPEG está representada en la tabla 3. Cada valor cuantificado es representado por un label. El label correspondiente al valor del coeficiente transformado θ_{ij} es

$$l_{ij} = \left[\frac{\theta_{ij}}{Q_{ij}} + 0.5 \right]$$

donde θ_{ij} es el elemento ij de la tabla de cuantificación, y [x] es la parte entera de x. Como ejemplo consideremos el coeficiente θ_{00} de la tabla 2 cuyo valor es 39.88. De la tabla 3 tenemos que Q_{00} es 16 y por lo tanto

$$l_{00} = \left[\frac{39.88}{16} + 0.5 \right] = [2.9925] = 2$$

El valor reconstruido es obtenido del valor cuantificado multiplicado por la correspondiente entrada en la tabla de cuantificación. Para el caso anterior el valor reconstruido del coeficiente θ_{00} es

$$l_{00} x Q_{00} = 2x16 = 32$$

El error de cuantificación en este caso es 39.88 - 32=7.88. Siguiendo los mismos pasos para los demás coeficientes se obtienen los labels de la tabla 4.

Por la tabla 3 podemos ver que las longitudes de los intervalos de cuantificación se incrementan a medida que nos movemos desde el coeficiente DC(posición 0,0) hacia los coeficientes de alto orden. Debido a que el error de cuantificación es una función creciente de las longitudes de los intervalos los más grandes errores de cuantificación ocurrirán en los coeficientes de más alta frecuencia. La decisión de cual debe ser la longitud de los

intervalos depende de como serán percibidos los errores en esos coeficientes por el sistema visual humano. Coeficientes en posiciones diferentes tienen importancia perceptual diferente. Los errores de cuantificación en el coeficiente DC y en los AC(coeficientes de posiciones distintas de 0,0) de baja frecuencia son mas fácilmente detectables que los errores en coeficientes de más alta frecuencia. Por lo tanto podríamos utilizar longitudes de intervalos más grandes para los coeficientes menos importantes

Distintos grados de pérdida se pueden lograr multiplicando la matriz de la tabla 3 por un escalar.

La flexibilidad dada para esta etapa permite construir matrices de cuantificación adaptivas acorde a las características de la imágen que está siendo comprimida. Por ejemplo en zonas donde se producen variaciones bruscas(como el pelo o los bordes del sombrero en la imágen LENA) podrían aparecer coeficientes de alta frecuencia con un peso importante a la hora de reconstruir la imágen y que podrían no ser tenidos en cuenta si se utilizara una matriz no adaptiva.

Tabla 3 Matriz de cuantificación default propuesta por JPEG

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Tabla 4 Coeficientes DCT transformados y cuantificados

2	1	0	0	0	0	0	0
-9	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Codificación:

En este paso los coeficientes DC y AC resultantes de la cuantificación son codificados utilizando códigos de longitud variable, pero ambos en forma diferente.

Los coeficientes DC (coeficiente de la posición 0,0) son codificados en forma diferencial con respecto al mismo elemento del bloque previo. La razón de esto es que el elemento de esta posición es un múltiplo del valor promedio de los pixels del bloque y, que en general estos promedios no difieren substancialmente entre bloques vecinos. Es de esperarse que esto ocurra en la mayor cantidad de situaciones. Por lo tanto para codificar las diferencias se asigna un código Huffman al conjunto de categorías en las que pueden caer dichos valores. Una diferencia en particular se codifica grabando primero la palabra de código correspondiente a su categoría y agregando luego bits extras al final de esa palabra de código para especificar su valor en una cantidad indicada por el número de categoría. Las categorías y sus correspondientes rangos de valores son mostrados en la tabla 5.

Si generalizamos para una categoría de diferencia (por ejemplo K) se utilizan K bits para codificar ya sea los K bits menos significativos de la diferencia positiva o bien los K bits menos significativos de la diferencia negativa menos 1.

La tabla 6 muestra un código default provisto por JPEG donde la primer columna contiene categorías de diferencia, la segunda la palabra de código para esa categoría y la tercera la cantidad total de bits necesaria para codificar una diferencia.

Tabla 5 Categorías de coeficientes y sus respectivos rangos

Nro. de Categoría	Rangos de valores
0	0
1	-1 1
2	-3 ... -2 2 ... 3
3	-7 ... - 4 4 ... 7
4	-15 ... -8 8 ... 15
5	-31 ... -16 16 ... 31
6	-63 ... -32 32 ... 63
7	-127 ... -64 64 ... 127
8	-255 ... -128 128 ... 255
9	-511 ... -256 256 ... 511
10	-1023 ... -512 512 ... 1023
11	-2047 ... -1024 1024 ... 2047
12	-4095 ... -2048 2047 ... 2048
13	-8191 ... -4096 4096 ... 8191
14	-16383 ... -8192 8192 ... 16383
15	-32767 ... 16384 16384 ... 32767
16	32768

Tabla 6

Categoría	Palabra de código	Long. Total
0	010	3
1	011	4
2	100	5
3	00	5
4	101	7
5	110	8
6	1110	10
7	11110	12
8	111110	14
9	1111110	16
A	11111110	18
B	111111110	20

Dado que la categoría 0 contiene solamente un elemento no se necesitan bits extras para especificar el valor. La categoría 1 contiene dos elementos de manera que solo se necesita agregar 1 bit al final del código Huffman para la categoría 1 para especificar alguno de los dos elementos. De igual forma se necesitan 2 bits para especificar un elemento en la categoría 2, 3 bits para la categoría 3 y n bits para la categoría n.

El código binario para los coeficientes AC es generado en una forma diferente. La matriz transformada y cuantificada es reordenada siguiendo un trayecto de zigzag(ver Figura 1) para obtener una secuencia unidimensional de los coeficientes cuantificados, de manera que la secuencia resultante se encuentre dispuesta de acuerdo al crecimiento de las frecuencias. El mayor beneficio que resulta del nuevo reordenamiento es que se puede tomar ventaja de una mayor longitud en las corridas de ceros.

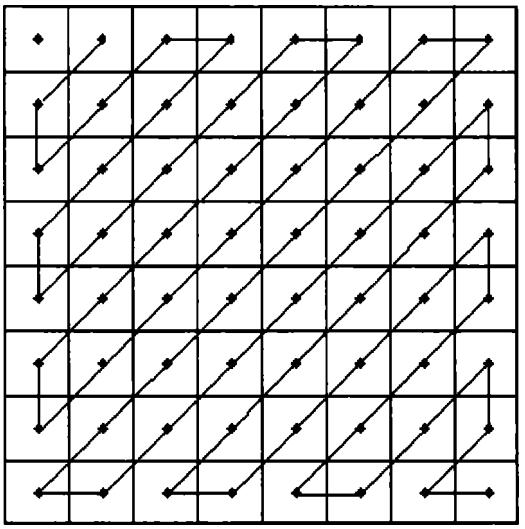
Concretamente los coeficientes AC son codificados usando un código de longitud variable como el de la tabla 7 donde se definen: C como la categoría en la que cayó un coeficiente y Z como el número de coeficientes con valor cero que ocurrieron en la secuencia después del último coeficiente distinto de cero. Una versión más completa de esta tabla se encuentra en la sección de este trabajo. De acuerdo con la bibliografía para bloques de 8x8 y pixels de 8 bits C puede tomar valores que varían de 0 a 10. Para los valores de Z JPEG establece en su tabla de codificación default un valor máximo de 15 ceros consecutivos. De esta manera cuando en la secuencia de zigzag es encontrado un coeficiente cuyo valor cae en una cierta categoría C precedido por Z ceros se graba el código correspondiente a la entrada Z/C en la tabla. Seguidamente se graba el valor del

coeficiente en C bits de la misma forma en la que se graban las diferencias entre coeficientes DC.

Tabla 7 Algunas palabras de código de la tabla de codificación default provista por JPEG

Z/C	palabra de código	longitud total
0/0	1010(=EOB)	4
0/1	00	3
0/2	01	4
0/3	100	6
0/4	1011	8
0/5	11010	10
.	.	.
F/0(ZRL)	111111110111	12
F/1	11111111111111010	17
F/2	1	18
F/3	11111111111111011	19
F/4	0	20
F/5	11111111111111011	21
	1	22
	11111111111111100	
	0	
	11111111111111100	
	1	
	.	

Figura 1. Secuencia de zigzag para un bloque de 8x8 de coeficientes



Existen dos palabras de código especiales llamadas EOB, utilizada para indicar que el resto de los coeficientes en la secuencia de zigzag son ceros, y ZRL para el caso en que el número de ceros consecutivos en la secuencia excede 15.

Como se ve, al igual que en la cuantificación, también se proveen tablas de codificación default aunque el usuario puede construir las propias de manera de que puedan ser adaptadas a las características de la imágen que esta siendo comprimida.

Generalización a particionamiento fijo

La primer implementación es una generalización para particionar la imágen en bloques de tamaños fijos (4x4, 8x8, 16x16, 32x32), Se utilizan diferentes tablas de cuantificación y codificación: para particionamiento fijo con bloques de 8x8, se utilizó la matriz de cuantificación default y los distintos niveles de pérdida se lograron multiplicando esta matriz por un escalar o factor; las tablas de codificación Huffman para bloques de 8x8 son las provistas por el standard. Para los demás tamaños de bloques se construyó la siguiente matriz de cuantificación: =1+(1+i+j)*factor; 0<= i, j < N.

Si bien existen técnicas para construir tablas de cuantificación adaptivas, el criterio utilizado es que generalmente los coeficientes decrecen en importancia desde del vértice superior izquierdo hacia el inferior derecho. Por esta razón la matriz se construye de tal forma que los coeficientes más cercanos a la posición (0,0) conserven mayor precisión a la hora de ser decuantificados.

Las tablas de códigos Huffman se construyeron realizando un recorrido sobre diversas imágenes sobre las que se aplicaron distintos grados de pérdida. Estos recorridos permitieron extraer las probabilidades para los símbolos, que se eligieron de acuerdo al tamaño del bloque.

Observaciones y Resultados de la Generalización

Se utilizo la imágen VLSI.BMP para realizar las pruebas, es una fórmula escrita en blanco sobre un fondo negro, a la cual se le extrajo una subimágen de 64x64 pixels perteneciente a la parte de color negro. A esta subimágen se le aplicó el algoritmo de particionamiento de tamaño fijo para los 4 tipos de bloques con factor 1 y se obtuvieron radios de 18, 36, 55 y 65 para N=4, 8, 16 y 32 respectivamente los cuales permanecían fijos a pesar de hacer variar los factores, en tanto que las imágenes descomprimidas no presentaban cambios en el color negro, por lo menos visualmente. Luego se aplicó el mismo algoritmo a la imágen original completa y se obtuvieron los siguientes radios:

N=4		N=8		N=16		N=32	
Facto	Radio	Factor	Radio	Factor	Radio	Factor	Radio
2	6	2	13	2	6	5	13
8	8	7	24	6	11	9	21
18	10.4	9	27			10	23
25	10.8						

Los resultados óptimos en calidad visual se lograron con el procesamiento en bloques de 4x4, aunque los radios de compresión aumentaron con menos rapidez al aplicar factores mayores a 18. Como se observa en la tabla para factores 18 y 25 la diferencia en radio es mínima aunque con calidades similares en las imágenes descomprimidas. Con

factores 7, 6 y 10 para bloques de 8x8, 16x16 y 32x32 respectivamente las distorsiones en la zona de la fórmula comenzaron a ser más notorias. Si bien se logró mayor compresión con tamaños de bloques más grandes se produjeron distorsiones visibles que afectaban las regiones de color negro alejadas de la fórmula.

Particionamiento propuesto:Quadtree

Este particionamiento es utilizado en el procesamiento de imágenes, tratando de superar las dificultades presentadas en el particionamiento fijo. El algoritmo básico consiste en tomar un bloque cuadrado de imágen el cual es dividido en cuatro subloques cuadrados de igual tamaño. Esto se repite recursivamente desde la imágen original completa hasta que los cuadrados alcancen un tamaño deseado de acuerdo a un test de decisión, que dependerá de la aplicación. Se puede realizar el mismo particionamiento pero en forma *button up* comenzando el proceso con los subloques más chicos y uniéndolos para formar otros más grandes.

Generalización utilizando particionamiento Quadtree

Con Quadtree se trata de adaptar las diversas zonas de la imágen a un determinado tamaño de bloque, de acuerdo a las variaciones de los niveles de grises dentro del mismo. Zonas grandes con cambios lentos en los niveles de detalle, son tratadas con bloques más grandes y zonas con muchos detalles sean tratadas con bloques más pequeños.

Esta implementación se basa en un esquema propuesto por Chen (1989) en el cual la imágen es dividida en bloques iniciales de 32x32, los que a su vez son particionados completamente hasta obtener bloques de 4x4. Se aplica un test sobre 4 subloques hermanos de 4x4 para determinar si estos se procesan individualmente o bien son unidos para formar un posible bloque de 8x8. El mismo proceso se realiza para cada 4 bloques hermanos de 8x8, y 16x16.

El test es de la forma: if $abs(M_k(i)-M_k(j))>T_k$ para cualquier $i \neq j$

Procesar los 4 subloques individualmente;

Else

Unir los subloques y marcarlos como un posible más grande;

Los términos $M_k(i)$ y $M_k(j)$ $(i,j=1..4)$ representan promedios de niveles de grises de bloques hermanos en el k-ésimo nivel del árbol y T_k, el límite de decisión. La idea es que para determinar cuan activa es una zona de NxN la dividimos en 4 bloques iguales, hallamos sus promedios y vemos que tan similares son. Si no existe ningún par de promedios cuya diferencia supere un límite establecido previamente entonces consideramos a la zona de NxN como poco activa (un límite sugerido es 10 para imágenes de 8 bits) y la procesamos como un solo bloque. En caso contrario la dividimos.

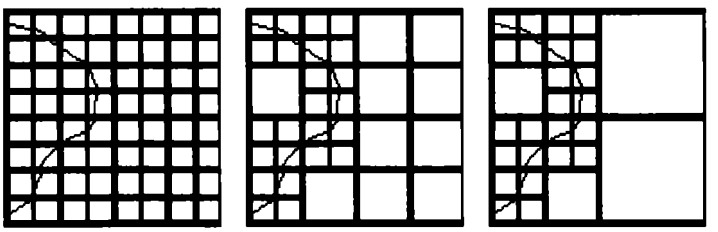
Para representar la información del header solamente necesitamos transmitir un bit por cada subrogue indicando si este se particiona o no, con lo cual en el peor de los casos

codificaremos 21 bits por cada bloque de 32x32 equivalente a un costo de 0.02 bits por pixel. De acuerdo al tamaño de cada subrogue del particionamiento final (4x4, 8x8, 16x16 o 32x32) el procesamiento es similar al aplicado por JPEG a cada bloque de 8x8 con la diferencia de que el coeficiente DC solo se codifica con respecto al mismo elemento del bloque anterior si es que ambos bloques tienen el mismo tamaño. Si el bloque anterior es de tamaño diferente se codifica el mismo valor del coeficiente cuantificado.

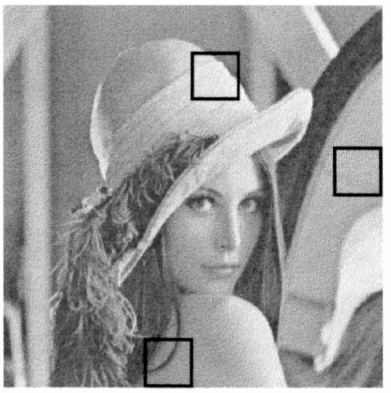
Se pueden variar los factores de pérdida para los distintos tamaños de bloques de acuerdo al nivel de calidad deseado para las distintas zonas. Por ejemplo, si se desea aumentar la pérdida en zonas con mucha actividad lo más conveniente es aumentar el factor de pérdida para los bloques más chicos, debido a que es de esperarse que estos tamaños de bloques caigan en esas zonas. De la misma manera que si se quiere disminuir la calidad de la imágen en zonas grandes con poca actividad se debe ajustar los factores para los tamaños de bloques más grandes.

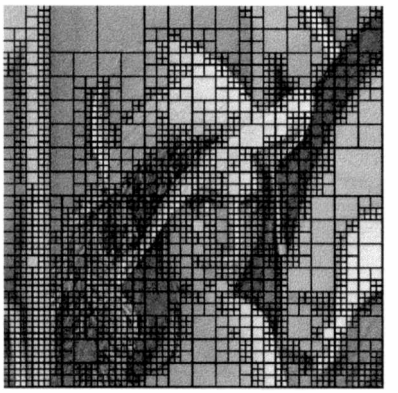
En general las zonas con poca actividad forman parte del fondo de la imágen y en consecuencia serán procesadas con bloques más grandes (32x32, 16x16).

Ejemplo de una secuencia de pasos que se siguen para particionar un bloque de 32x32.



a) **b)** **c)**

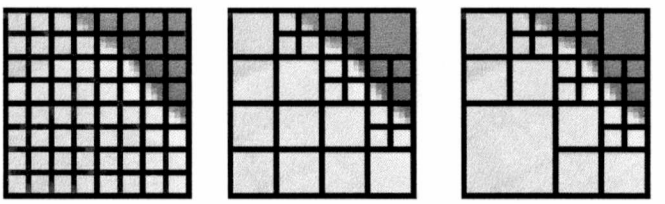




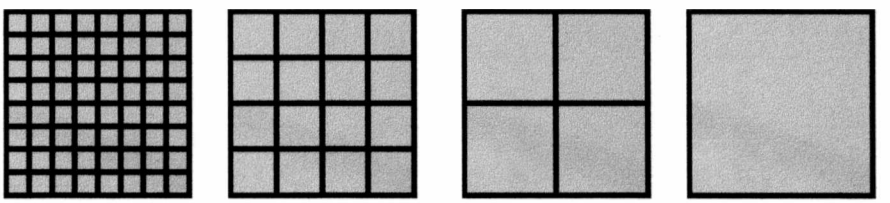
Imágen particionada con limites de 20,20,20

Tres ejemplos de secuencias de pasos que se siguen para particionar un bloque de 32x32 en Lena .bmp

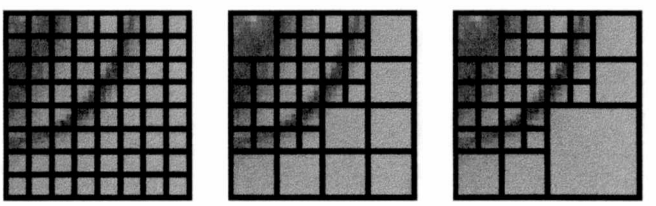
Bloque del sombrero



Bloque de espejo



Bloque del omoplato



En síntesis esta técnica de compresión cuenta con los siguientes parámetros que pueden ajustarse: Límite de decisión para decidir si particionar o no un bloque de 8x8, 16x16, 32x32; factor de pérdida para bloques de 4x4, 8x8, 16x16 y 32x32.

Observaciones y algunos resultados del método adaptivo

Por razones de espacio, en este artículo se analizan los resultados con 2 imágenes de características muy diferentes. Los resultados obtenidos de la imágen VLSI.BMP y LENA.BMP se lograron con un particionamiento de 10 y 20 respectivamente para cada uno de los tres limites.

Con el particionamiento de la primer imágen se logró que las zonas de la fórmula fueran tratadas con bloques mas chicos (4x4 y 8x8), mientras la mayor parte de la zona oscura fuera procesada con bloques de 32x32 y en menor cantidad con los de 16x16. Con esta distribución de bloques solo se produjeron distorsiones en los lugares muy cercanos a la fórmula y solo al aplicar factores muy altos. En las siguientes tablas se muestran algunos resultados obtenidos para las imágenes LENA y VLSI.

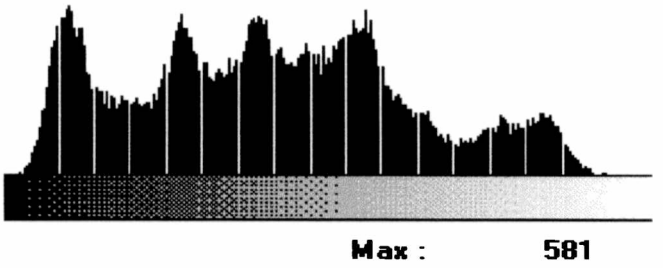
Si bien se puede aumentar el valor de los factores de pérdida para los bloques de 8x8, 16x16 y 32x32, sin distorsión visible para las zonas donde estos caen, estos incrementos no se ven reflejados en los tamaños de las imágenes comprimidas.

Tabla para VLSI.BMP

Descomprimida	Radio	Factor 4x4	Factor 8x8	Factor 16x16	Factor 32x32
pv1_1.bmp	6.4	2	2	2	2
pv1_2.bmp	9.9	9	2	2	2
pv1_3.bmp	9.98	9	9	2	2
Pv1_4.bmp	11	15	9	2	2
Pv1_5.bmp	13	20	9	2	2
Pv1_6.bmp	14.4	25	9	2	2
Pv1_7.bmp	14.5	25	9	9	9

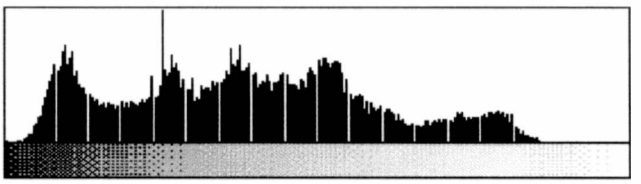
Tabla para LENA.BMP (con valores de 20 para los límites de decisión en el particionamiento)

Descomprimida	Radio	Radio Factor 4x4	Radio Factor 8x8	Radio Factor 16x16	Radio Factor 32x32
Plena4_1.bmp	4	2	2	2	2
Plena4_2.bmp	8.3	9	2	9	9
Plena4_3.bmp	6.4	5	3	5	5
Plena4_4.bmp	9	10	3	10	10
Plena4_5.bmp	10.8	15	2	15	15
Plena4_6.bmp	11.18	15	3	15	15
Plena4_7.bmp	11.4	15	4	15	15
Plena4_8.bmp	13.16	20	4	15	15
Plena4_9.bmp	13.3	20	5	15	15

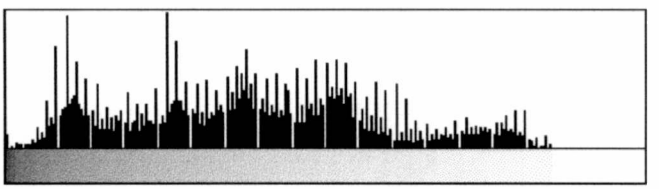


Lena.bmp (original ,66614 bytes) Histograma Original

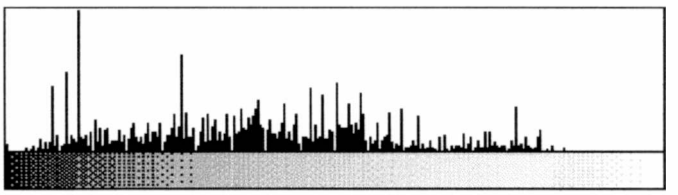
A continuación se muestran con orden de perdida producida algunas de las imágenes de la tabla anterior junto con sus histogramas de brillo.



plena4_1.bmp



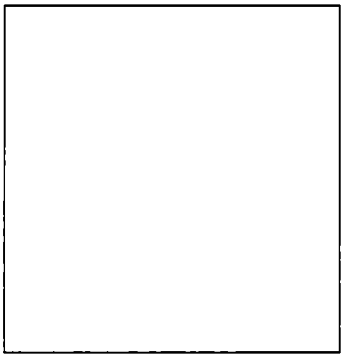
plena4_5.bmp



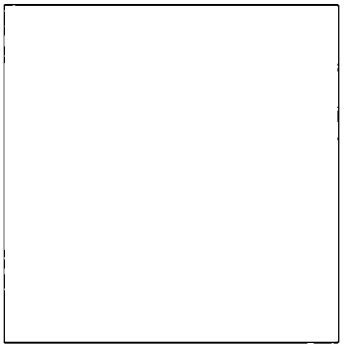
plena4_9.bmp

Siguiendo la secuencia de pérdidas vemos en los histogramas que ciertos niveles de grises tienden a aumentar considerablemente su cantidad de ocurrencias al mismo tiempo que otros disminuyen, esto se debe a que luego del procesamiento de un bloque cada uno de sus pixels tiende a tener el valor promedio de ese bloque, es decir el brillo del mismo

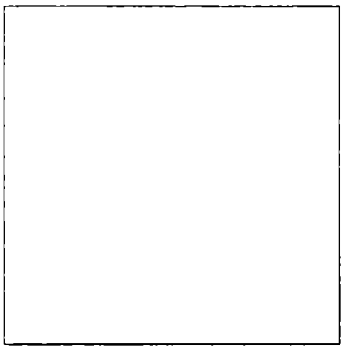
A continuación se muestran la perdida visual producida como la diferencia entre la imágen original y la descomprimida:



Error de plena4_1.bmp



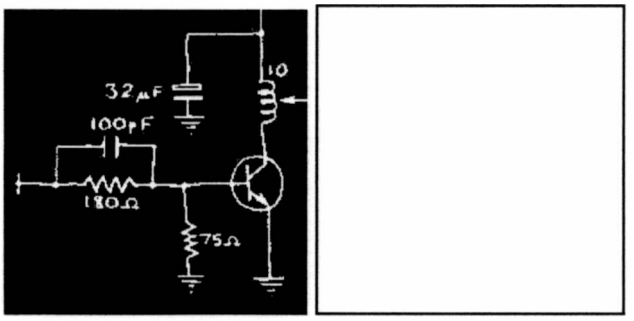
Error de plena4_5.bmp



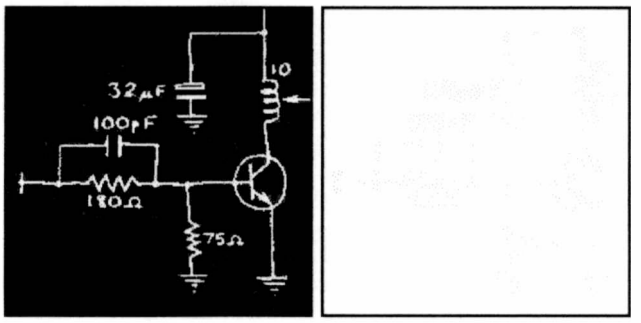
Error de plena4_9.bmp

Las zonas mas oscuras de las imágenes representan areas de mayor distorción que coinciden con las de mayor frecuencia como las del pelo, los ojos, contornos de los hombros, bordes del sombrero y el espejo, es decir en zonas donde se producen variaciones rápidas en el brillo .

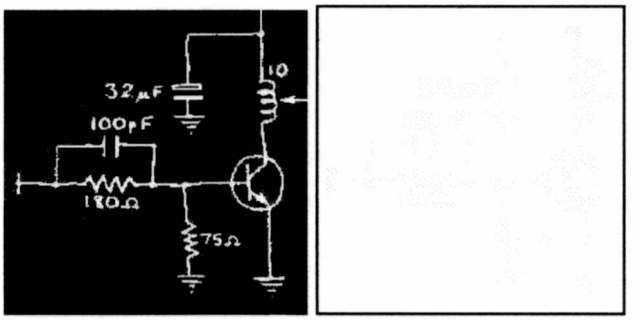
Algunos otros resultados con particionamiento variable para vlsi.bmp



pv1_1.bmp



pv1_4.bmp



pv1_7.bmp

Aparentemente en las tres imágenes reconstruídas tenemos la misma información visual, sin embargo las imágenes de la derecha indican que se produjeron distorciones en las zonas de los bordes que al no haber sido procesados con bloques grandes se pudo conservar un buen grado de fidelidad.

Criterios de fidelidad

Algo natural para hacer cuando estamos interesados en la fidelidad de una secuencia reconstruida es observar en las diferencias entre los valores originales y reconstruidos; en otras palabras la distorsión introducida en el proceso de compresión. Dos medidas conocidas de distorsión o diferencia entre las secuencias original y reconstruida es el error cuadrático medio y la de la diferencia absoluta, las cuales son llamadas *medidas de distorsión diferencia.*

Si $\{x_n\}$ es la salida de la fuente y $\{y_n\}$ es la secuencia reconstruida entonces la medida del error cuadrado es la siguiente:

$$d(x,y) = (x-y)^2$$

mientras que la medida de la diferencia absoluta esta dada por:

$$d(x,y) = |x-y|$$

En general es dificultoso examinar la diferencia en una base termino a termino. Es por eso que un conjunto de medidas del promedio es usado para resumir la información en la secuencia de diferencias.

La medida del promedio mas comúnmente utilizada es la del promedio de los errores cuadrados llamado *error cuadrático medio* (mean squared error)

$$\sigma^2 = \frac{1}{N}\sum_{n=1}^{N}(x_n - y_n)^2$$

Si estamos interesados en el tamaño del error relativo a la señal podemos hallar el radio del valor cuadrado promedio de la salida da la fuente y el *mse,* lo que se llama *signal-to-noise* (SNR):

$$SNR = \frac{\sigma_x^2}{\sigma_d^2}$$

donde es el cuadrado promedio de la salida de la fuente o señal y es el *mse* .El *SNR* es frecuentemente medido en una escala logarítmica y las unidades de medidas son los decibeles (dB):

$$SNR(dB) = 10\log_{10}\frac{\sigma_x^2}{\sigma_d^2}$$

Otras veces estamos más interesados en el tamaño de error relativo al valor máximo de la señal. Este radio es llamado peak signal to noise ratio (PSNR) y esta dado por

$$PSNR(dB) = 10\log_{10}\frac{X^2_{peak}}{\sigma_d^2}$$

Otra medida de distorsión diferencia que es muy usada aunque no tanto como el *mse* es el promedio de la diferencia absoluta:

$$\sum_i^n |x_i - y_i|$$

También suele utilizarse la raíz cuadrada del *mse* si se quiere saber en cuanto difieren en promedio los valores originales de los reconstruidos.

De acuerdo a la bibliografía utilizada y artículos publicados sobre optimizaciones de JPEG, el SNR y PSNR son utilizados casi como una medida standard de la eficiencia.

Entropía

El contenido de información en las señales en un sistema de comunicación(en nuestro caso una imágen a ser codificada) está caracterizada por la entropía H:

$$H=-\sum_{image} p(i)\log_2(p(i))$$

Donde i representa el valor de brillo de cada elemento en la imágen y p(i) su probabilidad de ocurrencia.

Si X representa la imágen original e Y la descomprimida la definición de entropía condicional de X dado Y es la cantidad de información que se puede conocer de la imágen original dado que conozco la reconstruida. Con esta definición que se encuentra mas detallada en la bibliografía y la sexta columna de la siguiente tabla se puede interpretar que a medida que nos alejamos cada vez mas de la imágen original mas incertidumbre tenemos con respecto de la imágen original a pesar de conocer la reconstruida. Mas allá de que no tengamos una medida concreta de la pérdida las tablas se encuentran ordenadas por el factor aplicado a las matrices de cuantificación.

Imágenes con Particionamiento de 8x8						
Imagen	ems	$\sqrt{mse}$	SNR(dB)	entropía de la imágen descomp. (bits)	entropía condicional (bits)	entropía de la imágen resta(bits)
lenf16.bmp	33,01	5,74	25.80	7,59	4.17	3,49
lenf2.bmp	56,28	7,5	23.48	7,54	4.48	3,84
lenf9.bmp	75,13	8,66	22.23	7,45	4.67	4,04
lenf12.bmp	92,73	9,62	21.32	7,31	4.79	4,2
lenf13.bmp	110,49	10,51	20.56	7,15	4.90	4,34
lenf14.bmp	129,29	11,37	19.87	7,04	4.98	4,46
lenf15.bmp	147,66	12,15	19.30	6,86	5.06	4,56

Cuantificación del vector

Estructura y perfomance

Introducción

La cuantificación del vector es una generalización de la cuantificación escalar a la cuantificación de un vector, un conjunto ordenado de números reales.

Con el salto de una dimensión a múltiples dimensiones surgen nuevas ideas, conceptos, técnicas y aplicaciones que pueden no tener su contraparte en el caso simple de la cuantificación escalar. Mientras la cuantificación escalar es usada principalmente para conversión analógico_digital, la cuantificación del vector(VQ) es usada con procesamiento de señales digitales sofisticadas donde en la mayoría de los casos la señal de entrada ya tiene alguna forma de representación digital y la salida deseada es una versión comprimida de la señal original. VQ es generalmente pero no exclusivamente usado para compresión de datos. Hay intereses paralelos con la cuantificación escalar y muchos de los modelos estructurales y analíticos y técnicas de diseño usadas en VQ son generalizaciones naturales del caso escalar.

Un vector puede ser utilizado para describir ciertos patrones como un segmento de una waveform de un speech o de una imágen simplemente formando un vector de muestras de waveform o imágen.

VQ puede ser vista como una forma de reconocimiento de patrones donde un patrón de entrada es aproximado por alguno de un conjunto predeterminado de patrones estandards o dicho de otra manera, el patrón de entrada es matcheado con alguno de un conjunto almacenado o codewords.

Algunas de las definiciones básicas pueden generalizarse desde la cuantificación escalar mientras que existen otras que no.

Presentaremos primero las definiciones básicas y las propiedades estructurales que son independientes de cualquier consideración estática o medida de distorsión.

Definiciones básicas

Un cuantificador del vector Q de dimensión k y tamaño N es un mapeo de un vector(o punto) en el espacio Euclidiano k_dimensional, R^k,dentro de un conjunto finito C conteniendo N puntos de salida o reproducción llamados vectores de código o codewords.

Es decir

$$Q : R^k \rightarrow C$$

donde $C = (y^1, y^2 \ldots y^n)$ con $y^i \in R^k$ para $i \in I = \{1,2,3..\ N\}$

El conjunto C es llamado el codebook o el código y el tamaño N significa que tiene N elementos distintos los cuales son vectores en R^k. La resolución, code rate, o simplemente rate de un cuantificador del vector es $r = (\log_2 N)/k$ que mide el número de bits por componente de vector utilizados para representar el vector de entrada y dá una indicación de la exactitud o precisión que es alcanzable con un VQ si el codebook está bien diseñado. Es decir que esta resolución está determinada por el tamaño N del codebook dada una dimensión k del VQ.

Una implementación típica del VQ es a través de una tabla en memoria y el número de bits empleados para representar cada componente de cada code vector no afecta a la resolución o bit rate del VQ pero influirá en la cuestión de las limitaciones del espacio de almacenamiento.

Asociado con cada cuantificador del vector de N puntos existe una partición de R^k con N regiones o celdas.

La celda i_ésima esta definida por:

$$R_i = \{x \in R^k : Q(x) = y_i\}$$

o también llamada la imágen inversa o preimágen de y_i con mapeo Q y denotado mas precisamente por $R_i = Q^{-1}(y_i)$.

De la definición de celdas tenemos:

$$\bigcap_i R_i = R_k \text{ y } R_i \bigcap R_j = Q$$

para i<>j así las celdas forman una partición de R^k.

Una celda sin límite es llamada celda sobrecargada y la colección de celdas sobrecargadas se denomina región sobrecargada . Una celda con límites es decir con volumen k_dimensional finito es llamada celda granular. La colección de celdas granulares se llama región granular.

Un cuantificador del vector se dice limitado si está definido sobre un dominio limitado, $B \subset R^k$,de manera que cualquier vector de entrada x cae en este conjunto.

El volumen de este conjunto B esta dado por $V(B) = \int_B dx$, y por lo tanto es finito.

Un cuantificador del vector se puede descomponer en dos operaciones:

Codificador del vector

Decodificador del vector

El Codificador E es el mapeo de R^k al conjunto índice J y el decodificador D mapea el conjunto índice J en el conjunto de reproducción C.

$$E : R^k \rightarrow J \quad \text{y} \quad D: J \rightarrow R^k$$

Con lo anterior vemos que una partición determina completamente como el codificador asignará un índice a un vector de entrada dado. Por otro lado un codebook determina por completo como el decodificador generará un vector de salida dado un índice.

La tarea del codificador es determinar en cual de las N regiones especificadas geométricamente el vector de entrada cae. El codificador no necesita precisamente conocer el codebook para realizar su función. Por otro lado el decodificador es simplemente una tabla look_up que está especificada completamente con la especificación del codebook.

El decodificador no necesita conocer la geometría de la partición para realizar su tarea.

Se verá que para la mayoría de los VQ de interés práctico el codebook provee información suficiente para caracterizar la partición y en este caso la operación del codificador puede ser realizada utilizando el codebook como el conjunto de datos que especifica completamente la partición.

La operación global del VQ. puede ser descripta como la cascada o composición de 2 operaciones :

$$Q(x)= D \circ E(x)=D(E(x))$$

Ocasionalmente es conveniente considerar al cuantificador como generador de un índice i y un valor de salida cuantificado Q(x).

El decodificador es considerado algunas veces como un cuantificador inverso .La cascada de un codificador y decodificador define un cuantificador.

Dentro de un sistema de comunicación digital el decodificador de un VQ realiza la tarea de seleccionar (implícita o explícitamente) un code vector(y_i) que mejor matchee y que aproximará, describirá o representará a un vector de entrada x.

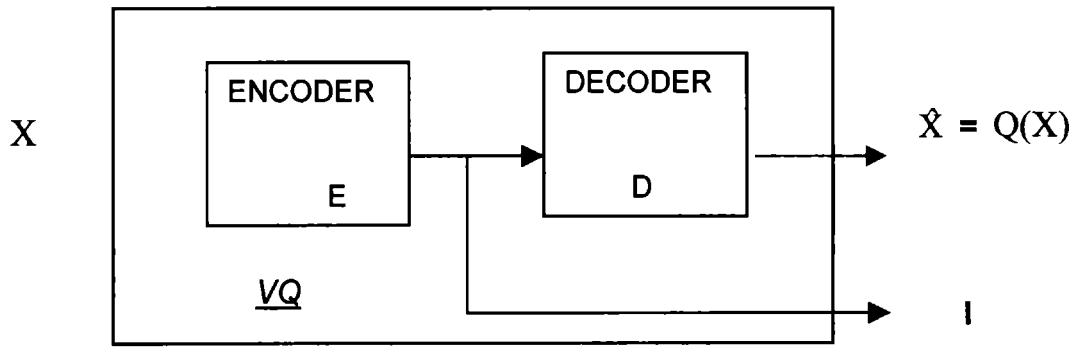


Fig 10

El índice y del code vector seleccionado es transmitido como una palabra binaria al receptor donde este realiza un acceso directo a una tabla y genera una reproducción y_i, una aproximación del vector de entrada original.

Para una secuencia de vectores de entrada que serán cuantificados y luego transmitidos el costo de bits o costo de transmisión R, en bits por vector está dado por R=k.r donde r es la resolución y k la dimensión del vector. Si llamamos a fv el costo por vector o el número de vectores de entrada a ser codificados por segundo entonces el costo de bits, R, en bits por segundo está dado por Rs= k.r.fv

Es conveniente ver la operación de un VQ en forma geométrica viendo intuitivamente el caso bidimensional y tridimensional: Por ejemplo un cuantificador bidimensional asigna un punto de entrada en el plano a algún otro en particular de un conjunto de N puntos o ubicaciones en el plano.

Cuantificacion del vector y JPEG

Los conceptos expuestos previamente nos ayudan a pensar en una futura posible mejora de la etapa de cuantificación de la técnica de codificación de la transformada.

Como vimos en esta etapa la técnica JPEG utiliza cuantificación escalar sobre cada coeficiente resultante de la transformación. Una posible variante podría consistir en tratar a cada bloque de nxn coeficientes como un vector y cuantificarlo como tal, en vez de hacerlo con cada uno de sus componentes en forma escalar. De esta manera para cada vector debería realizarse una búsqueda en una tabla que contenga los vectores más representativos. Una vez encontrado codificaríamos ese vector de la misma forma en que lo hacíamos con la técnica de JPEG es decir asignando un código de longitud variable Huffman a una secuencia de ceros y una categoría de coeficiente seguido por la codificación del valor del coeficiente .

Cabe aclarar que para formar la tabla de vectores representativos debemos construir las N regiones o celdas lo cual implica un proceso de recorrida sobre el conjunto total de vectores resultantes.

Conclusiones y Trabajos Futuros

Se observó que en imágenes con grandes zonas con poca actividad en niveles de grises permitían radios de compresión mayores si eran procesadas con bloques más grandes produciendo distorsiones poco visibles en las zonas poco activas, pero con el costo de pérdidas notables de calidad en zonas con poca actividad adyacentes a otras con mucho nivel de detalle. Con la reducción de los tamaños de bloques esas zonas conflictivas se fueron restringiendo pero con la desventaja de la disminución en los radios de compresión.

Concluimos además que el particionamiento elegido nos permite un mayor grado de compresión en imágenes con paisajes o retratos con grandes áreas que visualmente representan una misma tonalidad de gris, ya que en esas zonas pueden aplicarse bloques de tamaño mas grande y con mayor pérdida . Lo vimos en la imágen Lena, donde areas como el fondo, el homóplato y parte del espejo eran procesadas con bloques de 16x16 o 32x32.

Vimos que en imágenes relacionadas a paisajes, retratos u otras que son observadas
por el ser humano nos permiten un grado de compresión mayor. Los excelentes resultados comparativos del esquema adaptivo respecto del fijo, nos llevan a la idea de la paralelización del algoritmo adaptivo, a fin de tener tiempos razonables para procesamiento en tiempo real. Un estudio más profundo de la etapa de cuantificación podría llevarnos a una optimización de esta etapa, aunque nos alejaríamos de la técnica aplicada por JPEG, motivo de este trabajo.

Tabla default JPEG para los coeficientes AC

Corrida de ceros/Categoría	*Código base*	*Longitud*
0 / 0	**1010(=EOB)**	4
0 / 1	00	3
0 / 2	01	4
0 / 3	100	6
0 / 4	1011	8
0 / 5	11010	10
0 / 6	111000	12
0 / 7	1111000	14
0 / 8	1111110110	18
0 / 9	1111111110000010	25
0 / A	1111111110000011	26
1 / 1	1100	5
1 / 2	111001	8
1 / 3	1111001	10
1 / 4	111110110	13
1 / 5	11111110110	16
1 / 6	1111111110000100	22
1 / 7	1111111110000101	23
1 / 8	1111111110000110	24
1 / 9	1111111110000111	25
1 / A	1111111110001000	26
2 / 1	11011	6
2 / 2	11111000	10
2 / 3	1111110111	13
2 / 4	1111111110001001	20
2 / 5	1111111110001010	21
2 / 6	1111111110001011	22
2 / 7	1111111110001100	23
2 / 8	1111111110001101	24

Corrida de ceros/Categoría	*Código base*	*Longitud*
2 / 9	1111111110001110	25
2 / A	1111111110001111	26
3 / 1	111010	7
3 / 2	111110111	11
3 / 3	11111110111	14
3 / 4	1111111110010000	20
3 / 5	1111111110010001	21
3 / 6	1111111110010010	22
3 / 7	1111111110010011	23
3 / 8	1111111110010100	24
3 / 9	1111111110010101	25
3 / A	1111111110010110	26
4 / 1	111011	7
4 / 2	1111111000	12
4 / 3	1111111110011111	19
4 / 4	1111111110011000	20
4 / 5	1111111110011001	21
4 / 6	1111111110011010	22
4 / 7	1111111110011011	23
4 / 8	1111111110011100	24
4 / 9	1111111110011101	25
4 / A	1111111110011110	26
5 / 1	1111010	8
5 / 2	1111111001	12
5 / 3	1111111110011111	19
5 / 4	1111111110100000	20
5 / 5	1111111110100001	21
5 / 6	1111111110100010	22
5 / 7	1111111110100011	23

Corrida de ceros/Categoría	*Código base*	*Longitud*
5 / 8	1111111110100100	24
5 / 9	1111111110100101	25
5 / A	1111111110100110	26
6 / 1	1111011	8
6 / 2	11111111000	13
6 / 3	11111111101000111	19
6 / 4	11111111101001000	20
6 / 5	11111111101001001	21
6 / 6	11111111101001010	22
6 / 7	11111111101001011	23
6 / 8	11111111101001100	24
6 / 9	11111111101001101	25
6 / A	11111111101001110	26
7 / 1	11111001	9
7 / 2	11111111001	13
7 / 3	1111111110101111	19
7 / 4	1111111110110000	20
7 / 5	1111111110110001	21
7 / 6	1111111110110010	22
7 / 7	1111111110110011	23
7 / 8	1111111110110100	24
7 / 9	1111111110110101	25
7 / A	1111111110110110	26
8 / 1	11111010	9
8 / 2	111111111000000	17
8 / 3	1111111110110111	19
8 / 4	1111111110111000	20
8 / 5	1111111110111001	21
8 / 6	1111111110111010	22

Corrida de ceros/Categoría	*Código base*	*Longitud*
8 / 7	1111111110111011	23
8 / 8	1111111110111100	24
8 / 9	1111111110111101	25
8 / A	1111111110111110	26
9 / 1	111111000	10
9 / 2	1111111110111111	18
9 / 3	1111111111000000	19
9 / 4	1111111111000001	20
9 / 5	1111111111000010	21
9 / 6	1111111111000011	22
9 / 7	1111111111000100	23
9 / 8	1111111111000101	24
9 / 9	1111111111000110	25
9 / A	1111111111000111	26
A / 1	111111001	10
A / 2	1111111111001000	18
A / 3	1111111111001001	19
A / 4	1111111111001010	20
A / 5	1111111111001011	21
A / 6	1111111111001100	22
A / 7	1111111111001101	23
A / 8	1111111111001110	24
A / 9	1111111111001111	25
A / A	1111111111010000	26
B / 1	111111010	10
B / 2	1111111111010001	18
B / 3	1111111111010010	19
B / 4	1111111111010011	20
B / 5	1111111111010100	21

Corrida de ceros/Categoría	*Código base*	*Longitud*
B / 6	1111111111010101	22
B / 7	1111111111010110	23
B / 8	1111111111010111	24
B / 9	1111111111011000	25
B / A	1111111111011001	26
C / 1	1111111010	11
C / 2	1111111111010	18
C / 3	1111111111011	19
C / 4	1111111111100	20
C / 5	1111111111101	21
C / 6	1111111111110	22
C / 7	1111111111111	23
C / 8	1111111100000	24
C / 9	1111111100001	25
C / A	1111111100010	26
D / 1	11111111010	12
D / 2	1111111111100011	18
D / 3	1111111111100100	19
D / 4	1111111111100101	20
D / 5	1111111111100110	21
D / 6	1111111111100111	22
D / 7	1111111111101000	23
D / 8	1111111111101001	24
D / 9	1111111111101010	25
D / A	1111111111101011	26
E / 1	111111110110	13
E / 2	1111111111101100	18
E / 3	1111111111101101	19
E / 4	1111111111101110	20

Corrida de ceros/Categoría	*Código base*	*Longitud*
E / 5	1111111111101111	21
E / 6	1111111111110000	22
E / 7	1111111111110001	23
E / 8	1111111111110010	24
E / 9	1111111111110011	25
E / A	1111111111110100	26
F / 0	**111111110111**	12
F / 1	1111111111110101	17
F / 2	1111111111110110	18
F / 3	1111111111110111	19
F / 4	1111111111111000	20
F / 5	1111111111111001	21
F / 6	1111111111111010	22
F / 7	1111111111111011	23
F / 8	1111111111111100	24
F / 9	1111111111111101	25
F / A	1111111111110110	26

Resultados de radios de compresión para particionamientos fijos

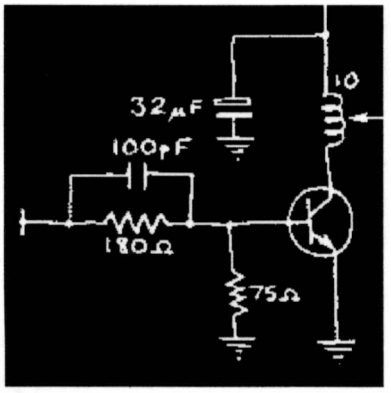


Tabla para la imágen anterior (vlsi.bmp)

N=4		N=8		N=16		N=32	
Factor	Radio	Factor	Radio	Factor	Radio	Factor	Radio
2	6	2	13	2	6	5	13
8	8	7	24	6	11	9	21
18	10.4	9	27			10	23
25	10.8						

Resultados para las medidas de pérdida(mse y $\sqrt{mse}$)

Imagen original : Lena.bmp

Particionamiento de 4x4				*Particionamiento de 8x8*			
Imágen	ems	$\sqrt{mse}$	Radio	imágen	mse	$\sqrt{mse}$	radio
Lenf17.bmp	11,27	3,35	4,4	lenf16.b	33,01	5,74	9,1
lenf19.bmp	25,06	5,00	6,6	lenf2.b	56,28	7,5	13,8
lenf21.bmp	38,53	6,20	8,3	lenf9.b	75,13	8,66	17,7
lenf23.bmp	51,69	7,19	9,7	lenf12.b	92,73	9,62	20,9
lenf24.bmp	64,64	8,04	10,9	lenf13.b	110,4	10,51	23,9
lenf25.bmp	76,79	8,76	11	lenf14.b	129,2	11,37	26,7
lenf26.bmp	88,18	9,39	13	lenf15.b	147,6	12,15	29,4
Lenf27.bmp	97,98	9,89	13				

Particionamiento de 16x16				*Particionamiento de 32x32*			
Imágen	mse	$\sqrt{mse}$	Radio	Imágen	mse	$\sqrt{mse}$	Radio
Lenf3.bmp	24,66	4,96	8	Lenf4.bmp	51,28	7,16	12
Lenf10.bm	57,81	7,6	14,5	Lenf11mp	104,3	10,21	23
Lenf29.bm	66,89	8,17	16	Lenf7	153,2	12,38	36
Lenf6.bmp	92,04	9,59	22	Lenf28bmp	181,1	13,45	44

Particionamiento Variable			
Imágen	mse	$\sqrt{mse}$	Radio
Plen4_1.bm	13,98	3,73	4
Plen4_2.bm	33,21	5,76	8,3
Plen4_3.bm	24,9	4,99	6,4
Plen4_4.bm	39,07	6,25	9
Plen4_5.bm	49,89	7,06	10,8
Plen4_6.bm	53,10	7,28	11,1
plen4_7.bm	55,91	7,47	11,4
plen4_8.bm	69,46	8,33	13,1
plen4_9.bm	72,66	8,52	13,3

Resultados para las medidas de pérdida, perfomance y entropía(mse y $\sqrt{mse}$)

Imágen original : Lena.bmp

Imágenes con Particionamiento de 4x4						
Imagen	ems	$\sqrt{mse}$	SNR(dB)	entropía de la imágen descomp. (bits)	Entropía condicional (bits)	entropía de la imágen resta(bits)
Lenf17.bmp	11,27	3,35	30.47	7,56	3.65	2,9
Lenf19.bmp	25,06	5,00	27.00	7,43	4.08	3,38
Lenf21.bmp	38,53	6,20	25.13	7,13	4.28	3,64
Lenf23.bmp	51,69	7,19	23.85	6,8	4.41	3,81
Lenf24.bmp	64,64	8,04	22.88	6,57	4.52	3,95
Lenf25.bmp	76,79	8,76	22.14	6,35	4.60	4,05
Lenf26.bmp	88,18	9,39	21.53	6,08	4.68	4,4
Lenf27.bmp	97,98	9,89	21.08	5,9	4.74	4,2

Imágenes con Particionamiento de 8x8						
Imagen	ems	$\sqrt{mse}$	SNR(dB)	entropía de la imágen descomp. (bits)	entropía condicional (bits)	entropía de la imágen resta(bits)
lenf16.bmp	33,01	5,74	25.80	7,59	4.17	3,49
lenf2.bmp	56,28	7,5	23.48	7,54	4.48	3,84
lenf9.bmp	75,13	8,66	22.23	7,45	4.67	4,04
lenf12.bmp	92,73	9,62	21.32	7,31	4.79	4,2
lenf13.bmp	110,49	10,51	20.56	7,15	4.90	4,34
Lenf14.bmp	129,29	11,37	19.87	7,04	4.98	4,46
Lenf15.bmp	147,66	12,15	19.30	6,86	5.06	4,56

Imágenes con Particionamiento de 16x16						
imágen	ems	$\sqrt{mse}$	SNR(dB)	entropía de la imágen descomp. (bits)	entropía condicional (bits)	entropía de la imágen resta(bits)
lenf3.bmp	24,66	4,96	27.072221	7,59	4.083058	3,37
lenf10.bmp	57,81	7,6	23.373444	7,61	4.528039	3,88
lenf29.bmp	66,89	8,17	22.739913	7,61	4,61	3,98
lenf6.bmp	92,04	9,59	21.353333	7,61	4.7874	4,18

Imágenes con Particionamiento de 32x32						
Imágen	ems	$\sqrt{mse}$	SNR(dB)	entropía de la imág descomp. (bits)	entropía condicional (bits)	entropía de la imágen resta(bits)
lenf4.bmp	51,28	7,16	23.893925	7,61	4.505544	3,84
lenf11.bmp	104,37	10,21	20.807579	7,62	4.9126	4,29
lenf7.bmp	153,28	12,38	19.138584	7,63	5.132177	4,55
lenf28.bmp	181,127	13,45	18.413694	7,63	5.24267	4,68

Imágenes con Particionamiento Variable						
Imagen	ems	$\sqrt{mse}$	SNR(dB)	entropía de la imág descomp. (bits)	entropía condicional (bits)	entropía de la imágen resta(bits)
Plen4_1.bmp	13,98	3,73	29.537773	7,57	3.62	2,9
Plen4_2.bmp	33,21	5,76	25.780164	7,56	4.27	3,56
Plen4_3.bmp	24,9	4,99	27.030027	7,55	4.09	3,36
Plen4_4.bmp	39,07	6,25	25.07497	7,54	4.37	3,67
Plen4_5.bmp	49,89	7,06	24.01287	7,48	4.48	3,8
Plen4_6.bmp	53,10	7,28	23.742412	7,47	4.53	3,85
Plen4_7.bmp	55,91	7,47	23.518611	7,38	4.56	3,89
Plen4_8.bmp	69,46	8,33	22.575681	7,41	4.66	4,02
Plen4_9.bmp	72,66	8,52	22.380217	7,33	4.69	4,06

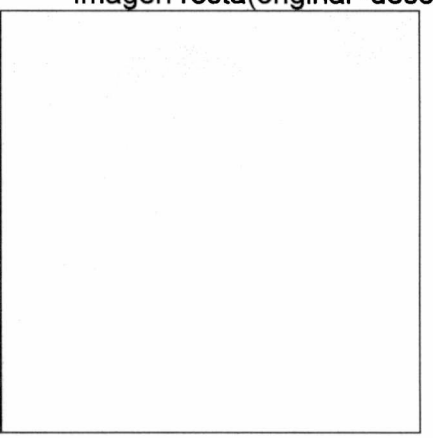
Podemos comparar más concretamente estos resultados si vemos las imágenes resultantes en la sección de resultados, sus radios de compresión y factores de pérdida dados en tablas anteriores. Por ejemplo con plen4_8 y plen4_9 vemos que al incementar la pérdida en los bloques de 8x8 obtuvimos un radio de compresión levemente mayor.

Resultados con particionamiento de 4x4

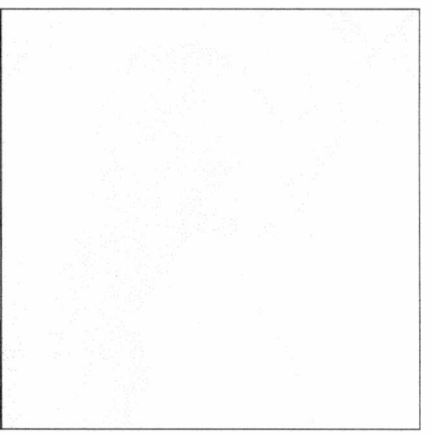
Lena.bmp (original ,66614 bytes)

Las siguientes secuencias representan la imágen descomprimida, la imágen resta(original -descomprimida). Las zonas más oscuras de la imágen resta representan zonas de mayor distorsión.

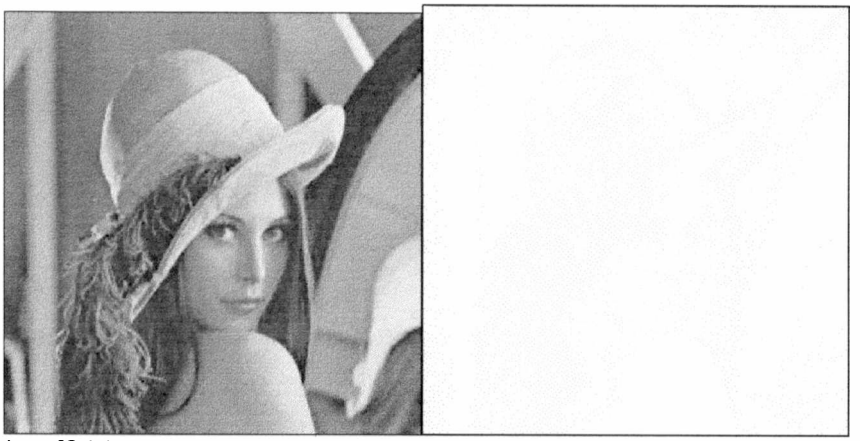
Imágen descomprimida Imágen resta(original -descomprimida)



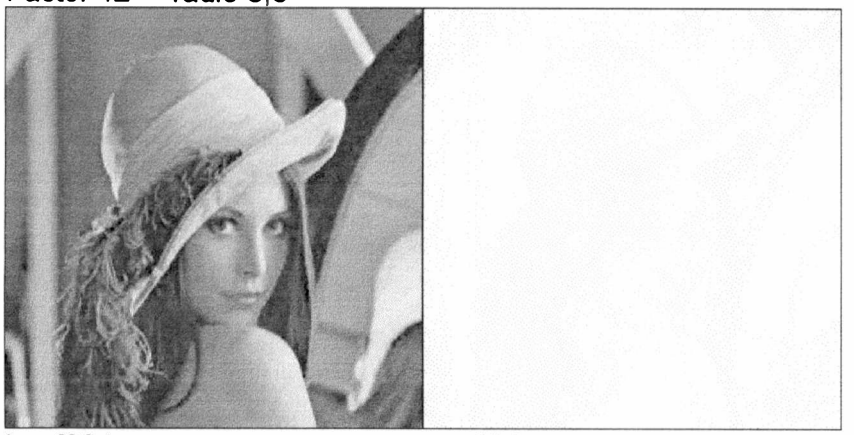
Lenf17.bmp res17.bmp
Factor 4 radio 4,4



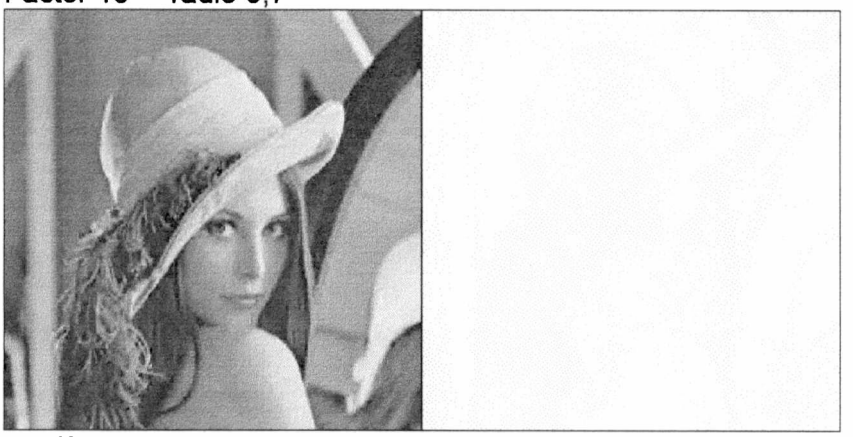
Lenf19.bmp res19.bmp
Factor 8 radio 6,6



Lenf21.bmp res21.bmp
Factor 12 radio 8,3

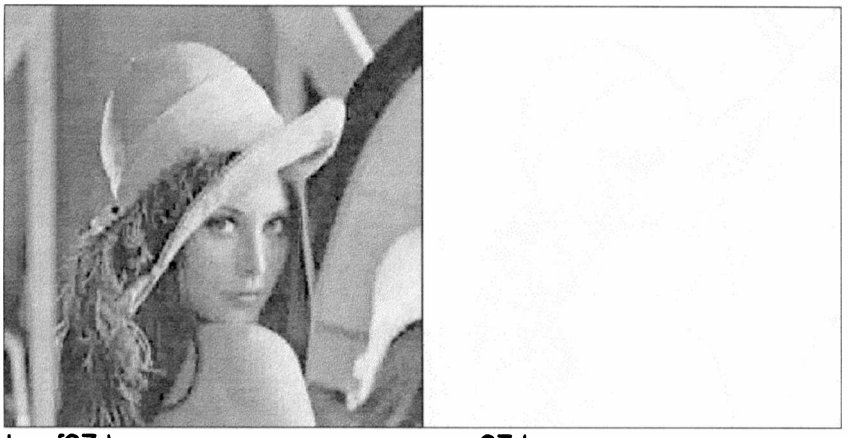


Lenf23.bmp res23.bmp
Factor 16 radio 9,7



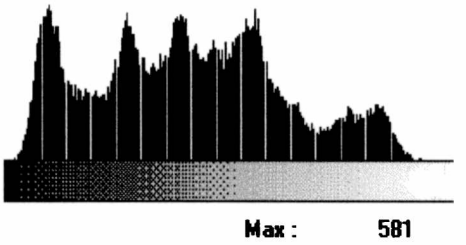
Lenf24.bmp res24.bmp
Factor 20 radio 10,9

Lenf26.bmp res26.bmp
Factor 28 radio 13

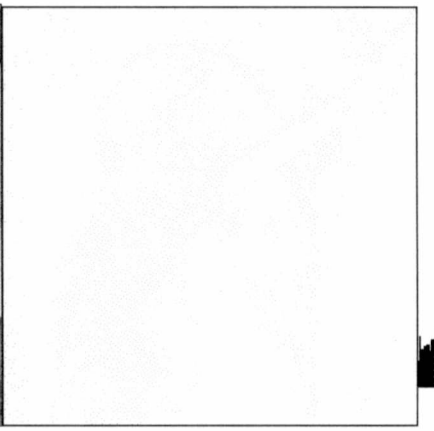


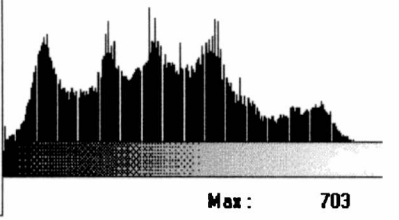
Lenf27.bmp res27.bmp
Factor 32 radio 13

Resultados con particionamiento de 8x8

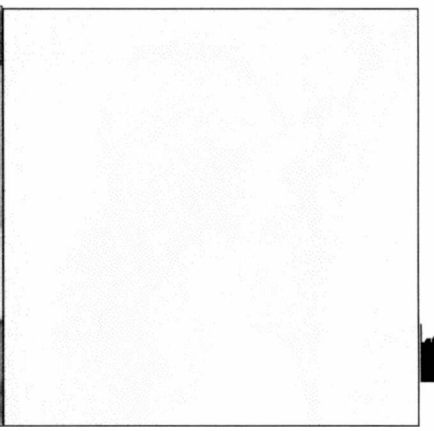


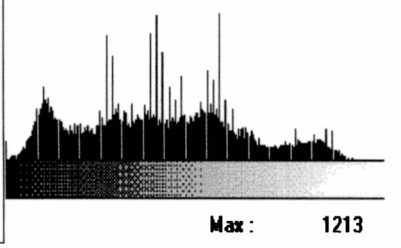
Lena.bmp (original ,66614 bytes) Histograma Original



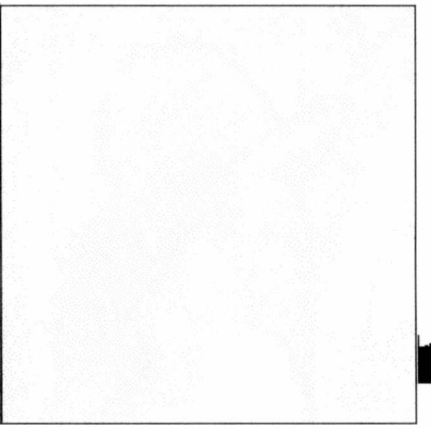


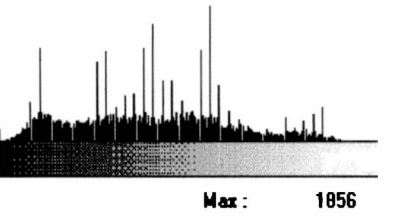
Lenf16.bmp factor 1 res16.bmp
Radio 9,1



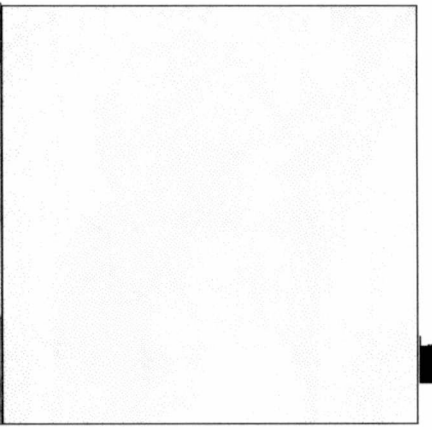


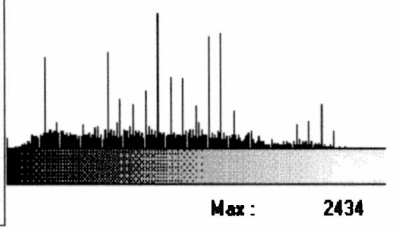
Lenf2.BMP factor 2 res2.bmp
Radio 13,8



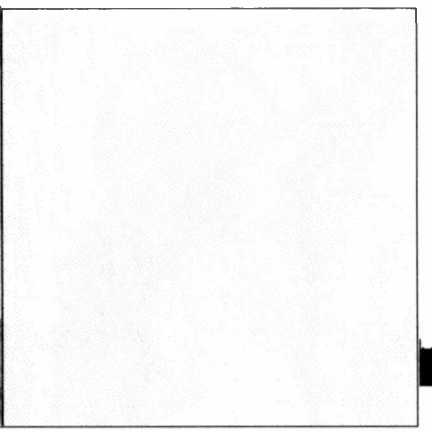


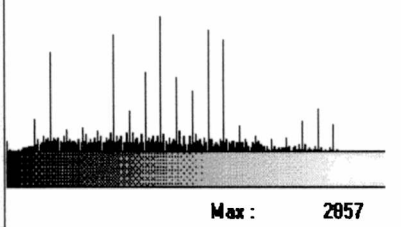
Lenf9.bmp factor 3 res9.BMP
Radio 17,7



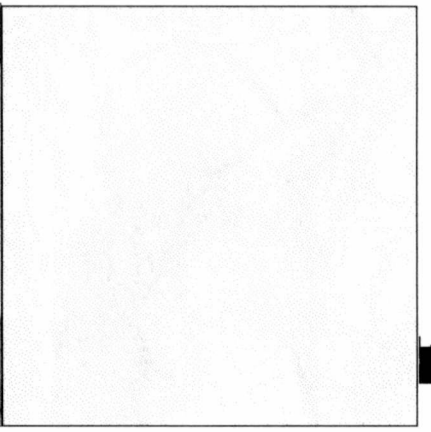
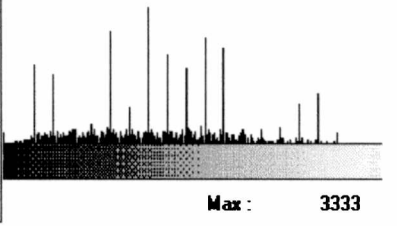


Lenf12.bmp factor 4 res12.BMP
Radio 20,9

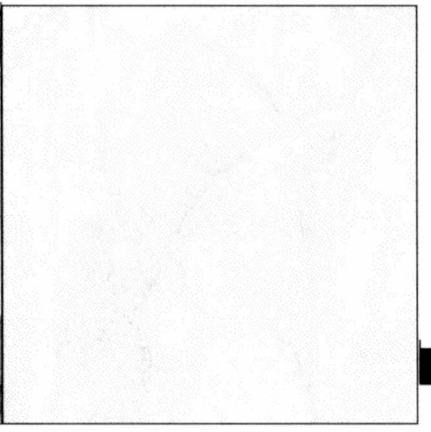
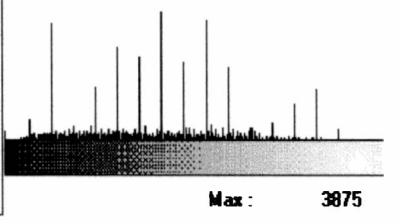




Lenf13.bmp factor 5 res13BMP
Radio 23,9

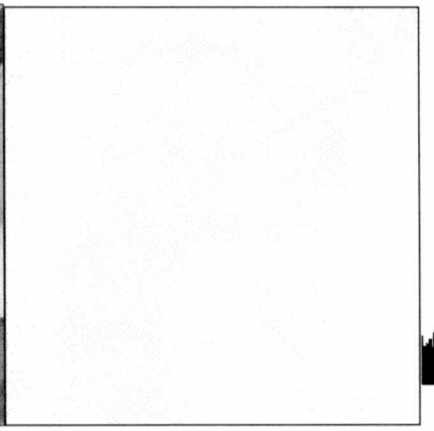
Lenf14.BMP Factor 6 Res14.BMP
Radio 26,7

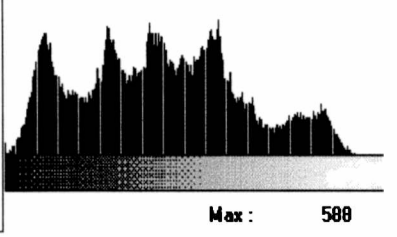




Lenf15.bmp factor 7 Res15.bmp
Radio 29,4

Resultados con particionamiento de 16x16

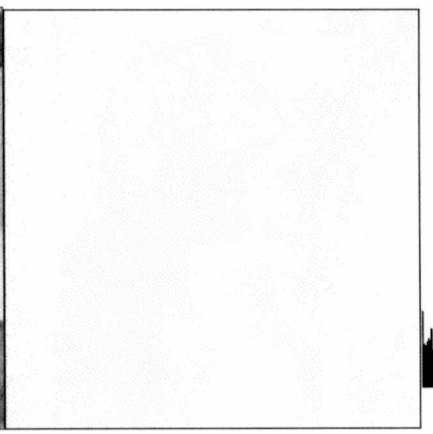
Imágen descomprimida | Imágen resta(original -descomprimida) | Histograma de la imag.desc.

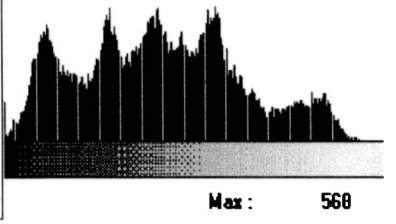




Lenf3.bmp
Factor 2 radio 8

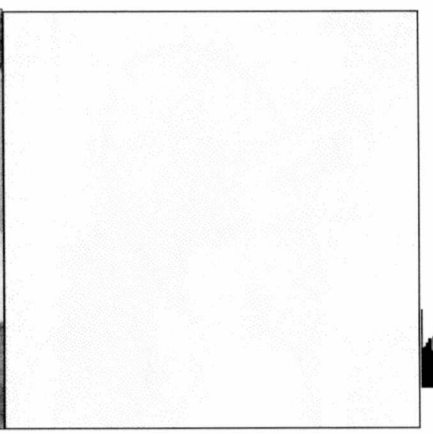
Res3.bmp

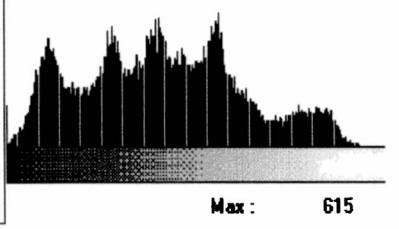




Lenf10.bmp
Factor 5 radio 14,5

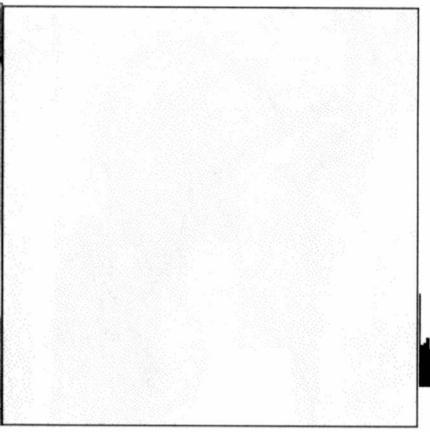
Res10.bmp

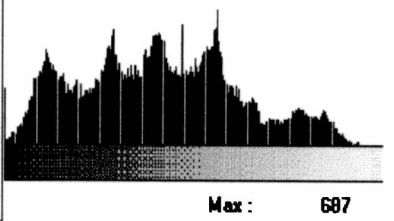




Lenf29.bmp
Factor 6 radio 16

Res29.bmp



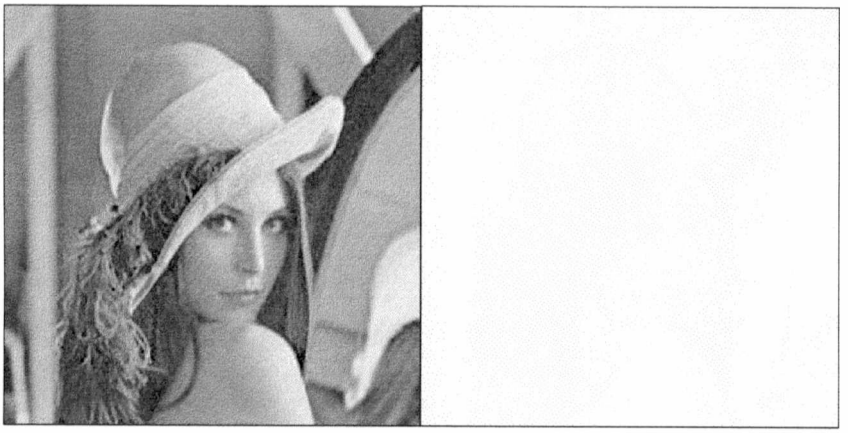


Lenf6.bmp Res6.bmp
Factor 9 radio 22

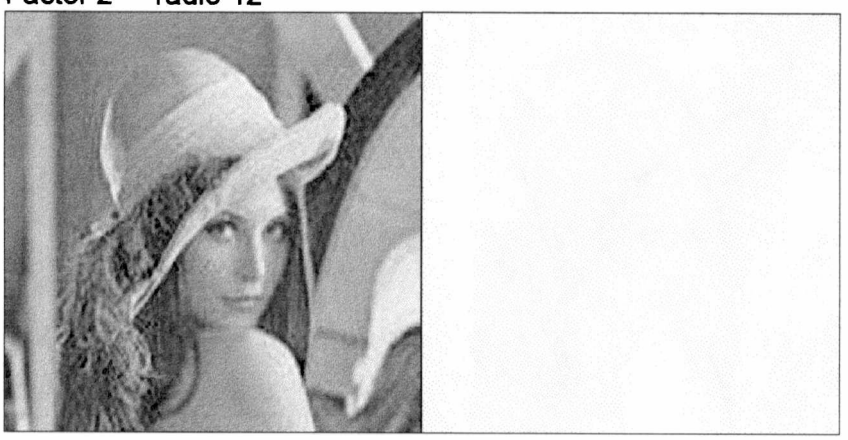
Resultados con particionamiento de 32x32

Imágen descomprimida

Imágen resta(original -descomprimida) Histograma de la imag.desc



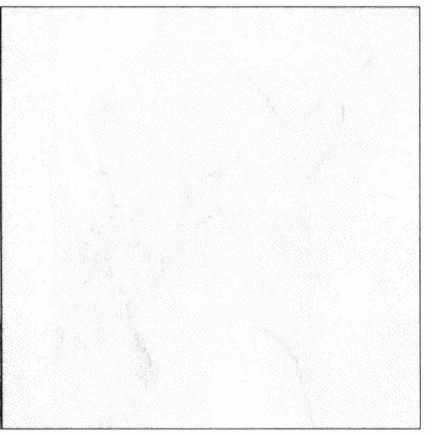
Lenf4.bmp res4.bmp
Factor 2 radio 12



Lenf11.bmp res11.bmp
Factor 5 radio 23

Lenf7.bmp res7.bmp
Factor 9 radio 36

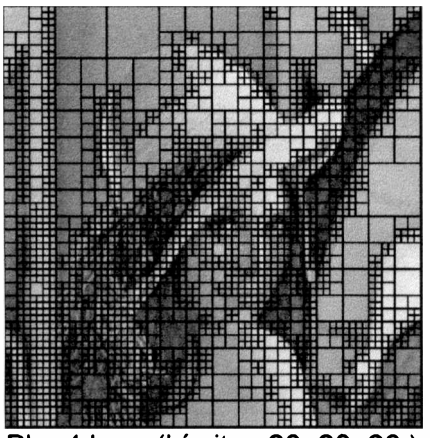
Lenf28.bmp



res28.bmp

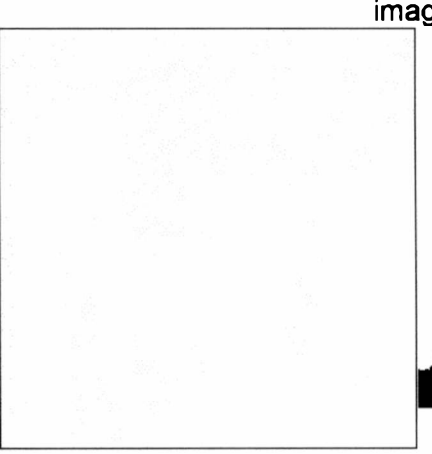
Factor 12 radio 44

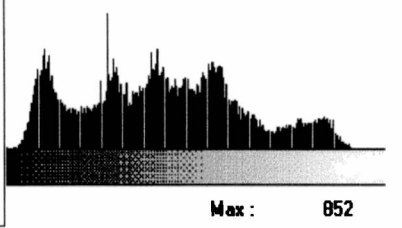
Resultados con particionamiento variable



Plen4.bmp (Límites 20 ,20, 20)

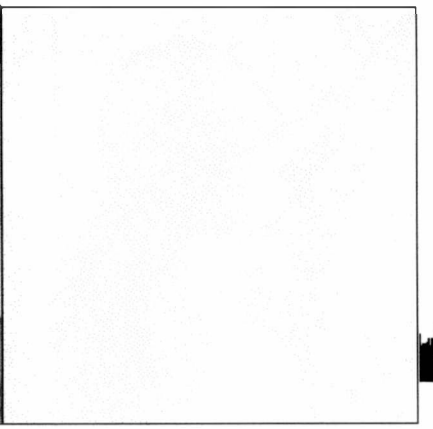
Imágen descomprimida | Imágen resta(original -descomprimida) imag. | Histograma de la desc.

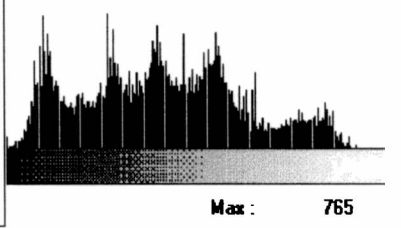




Plen4_1.bmp Dlen4_1.bmp

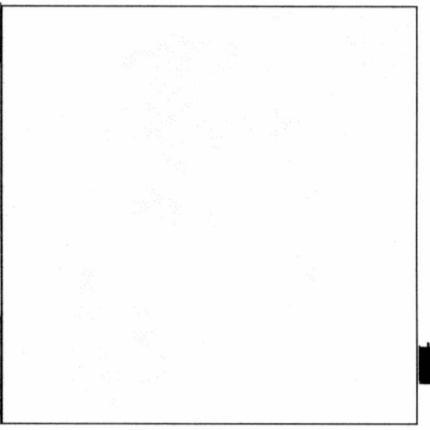
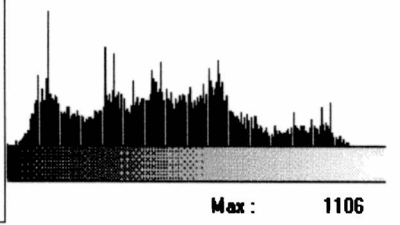
Radio 4





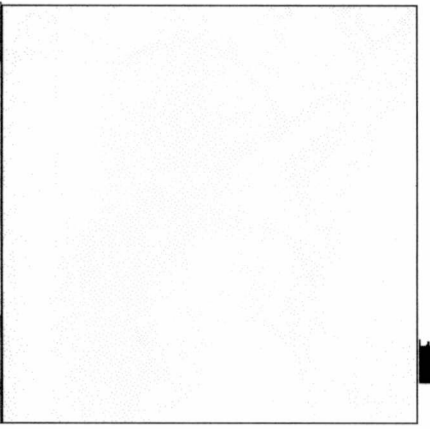
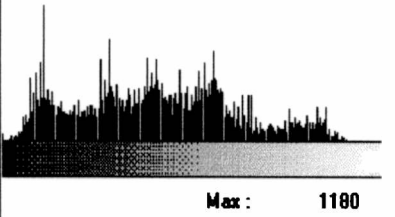
Plen4_2.bmp Dlen4_2.bmp

Radio 8,3

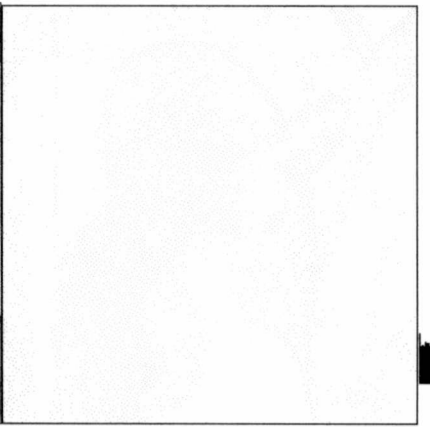
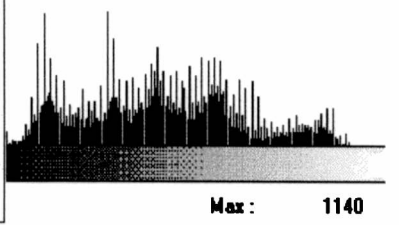
Plen4_3.bmp Dlen4_3.bmp

Radio 6,4

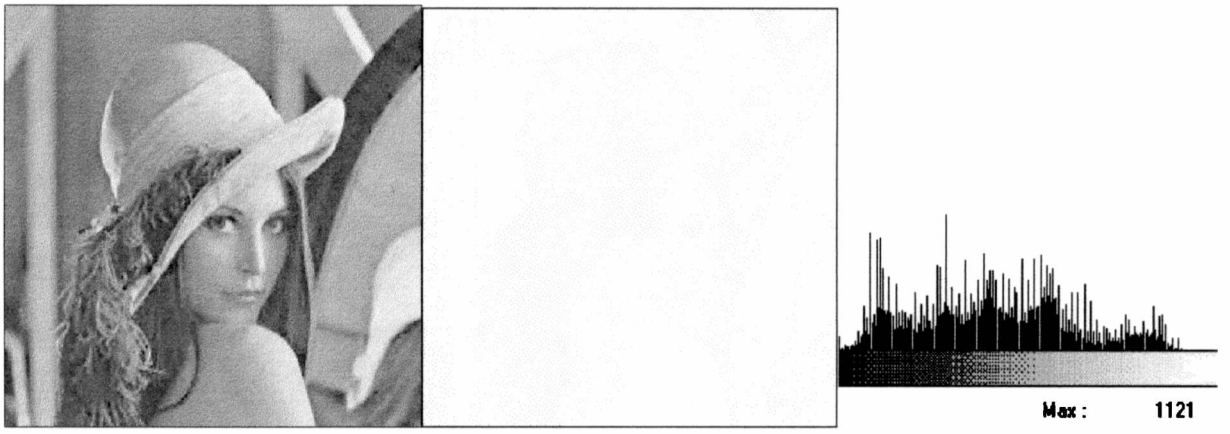
Plen4_4.bmp Dlen4_4.bmp

Radio 9

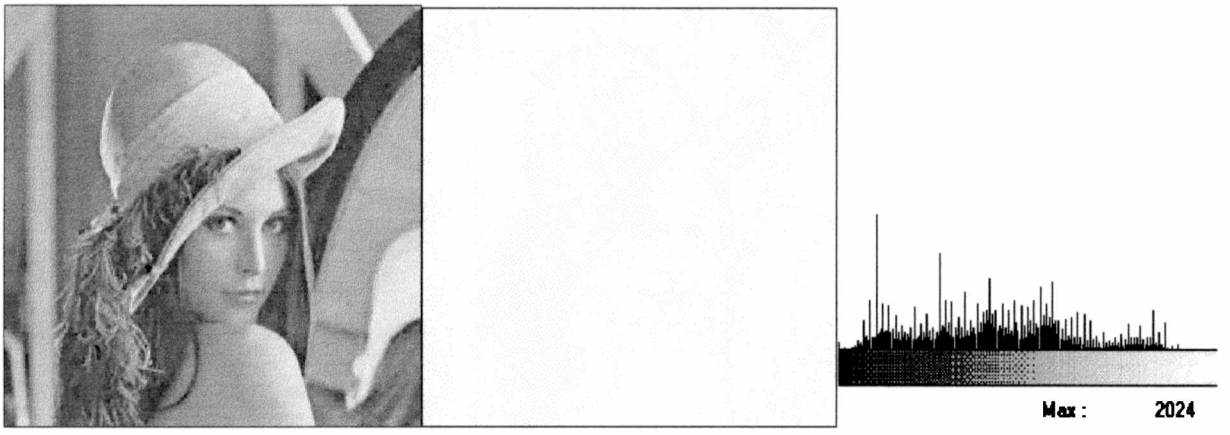
Plen4_5.bmp Dlen4_5.bmp

Radio 10,8



Plen4_6.bmp Dlen4_6.bmp

Radio 11,1

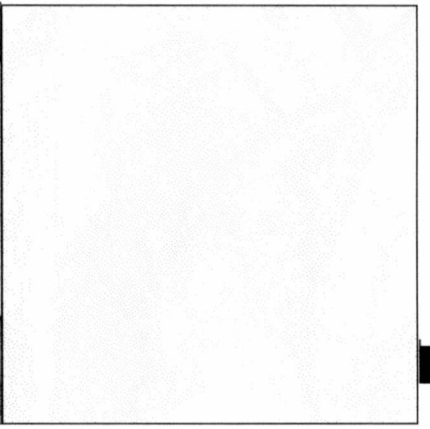


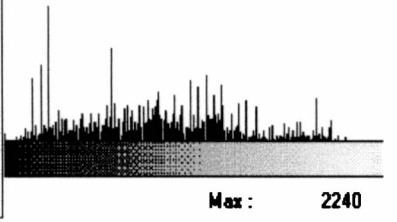
Plen4_7.bmp Dlen4_7.bmp

Radio 11,4

Plen4_8.bmp Dlen4_8.bmp

Radio 13,1





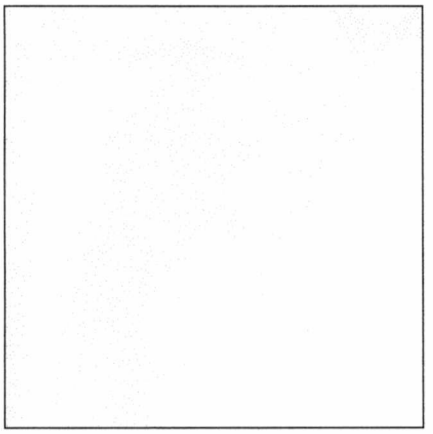
Plen4_9.bmp Dlen4_9.bmp

Radio 13,3

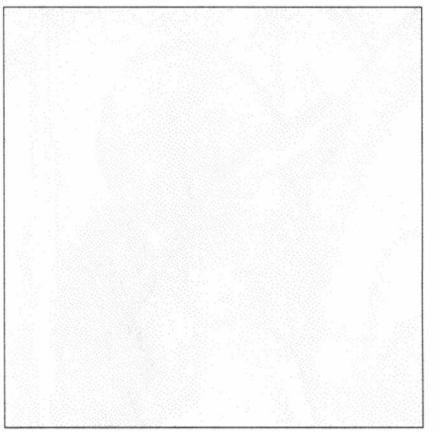
Comparación de las imágenes resta para LENA.BMP

En ambas columnas se muestran los peores y mejores casos de las imágenes resta obtenidas para Lena.bmp, conjuntamente con el radio de compresión de la imágen descomprimida correspondiente. Se presenta una comparación visual

Particionamiento fijo de 4x4

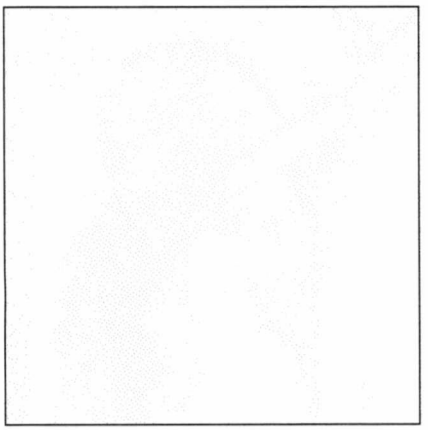


res17.bmp
radio 4,4

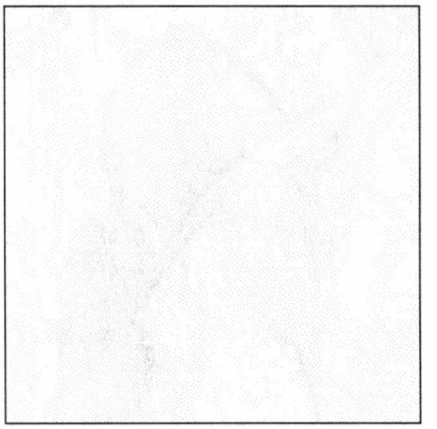


res27.bmp
radio 13

Particionamiento fijo de 8x8

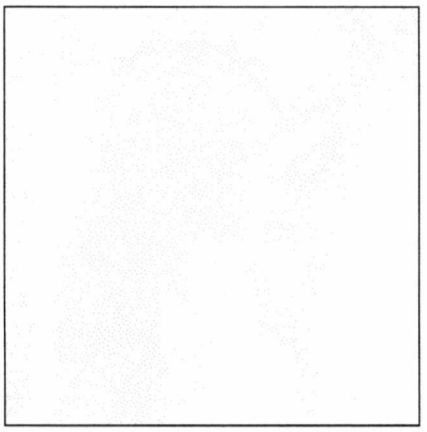


res16.bmp
radio 9,1



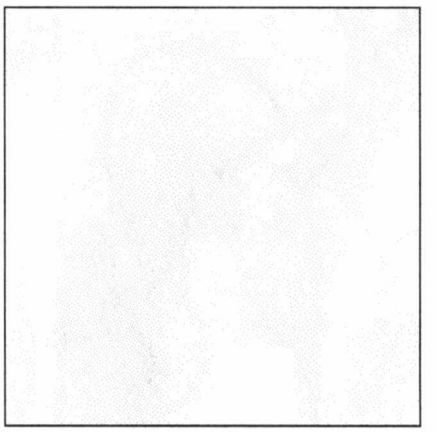
res15.bmp
radio 29,4

Particionamiento fijo de 16x16



res3.bmp

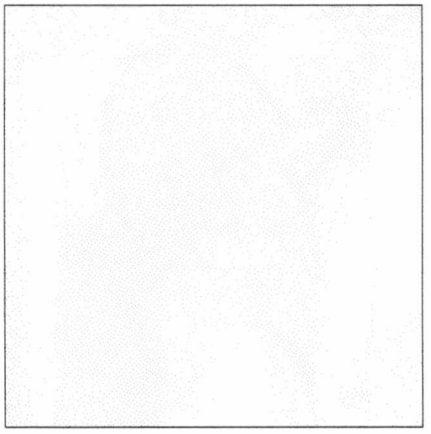
radio 8



res8.bmp

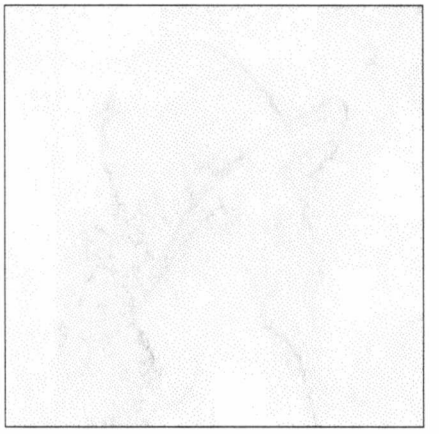
radio 22

Particionamiento fijo de 32x32



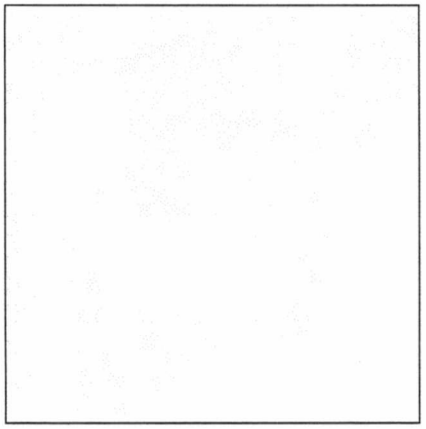
res4.bmp

radio 12



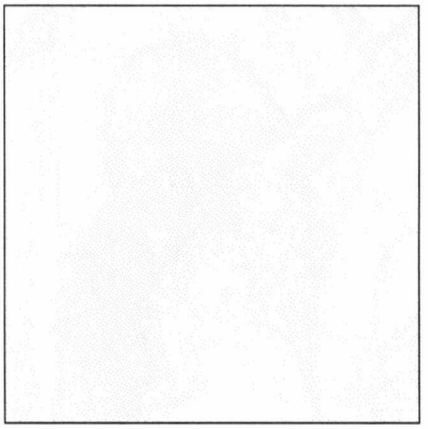
res28.bmp

radio 44



res4.bmp

radio 12



res28.bmp

radio 44

Imágen Tierra.bmp

Particionamiento fijo de 4x4

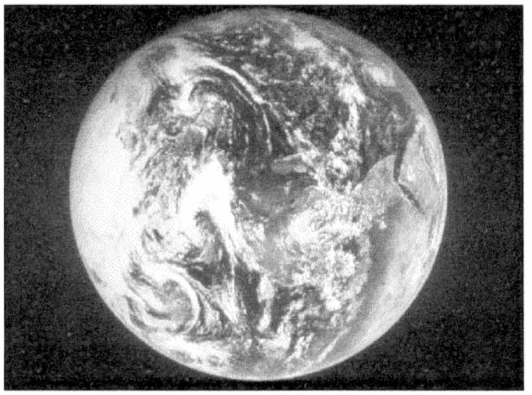
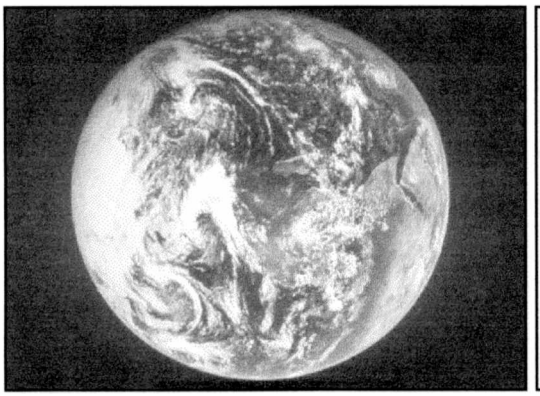


Imagen original

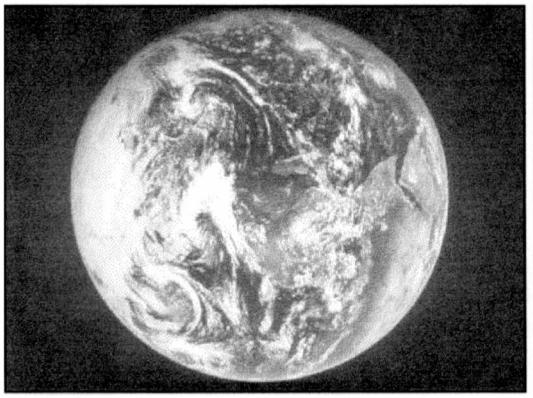
tie1.bmp

Radio 4



tie2.bmp

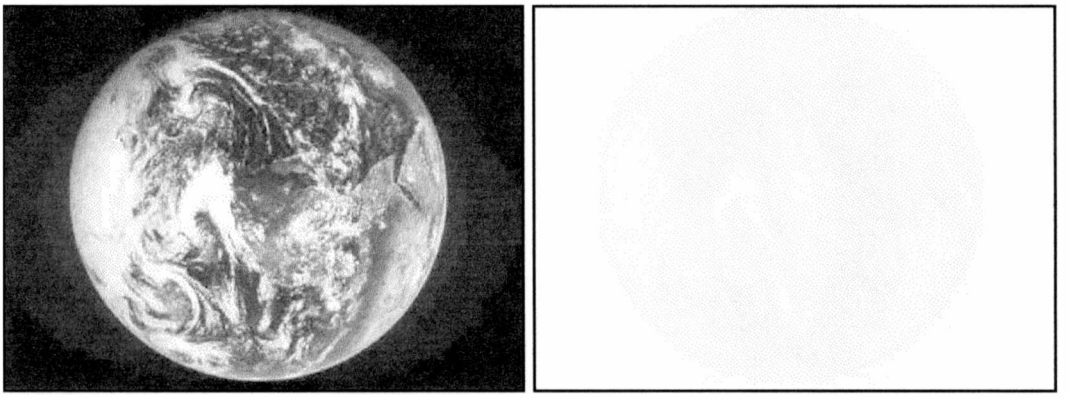
Radio 5,8



tie3.bmp

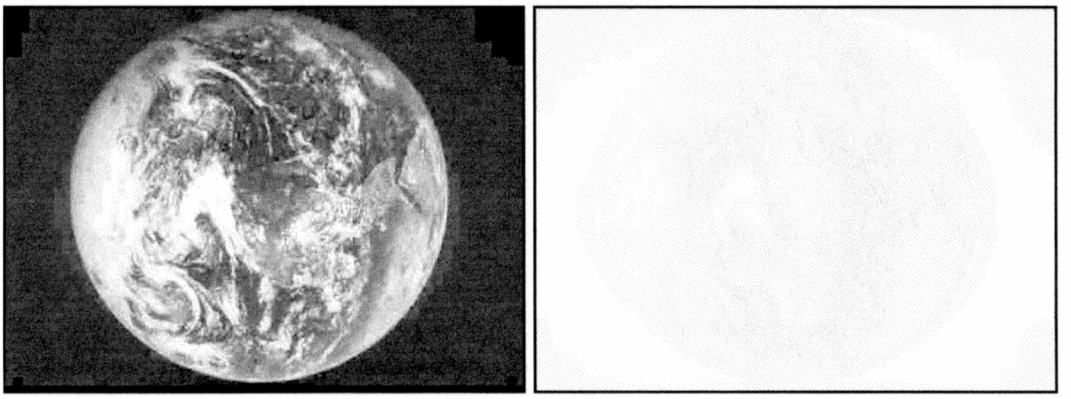
Radio 7,2

Particionamiento fijo de 8x8



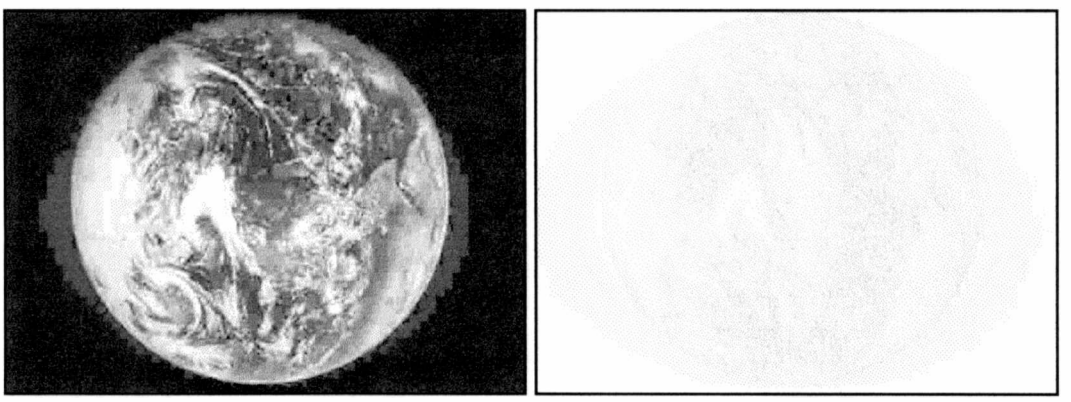
tie4.bmp

radio 18,4



tie5.bmp

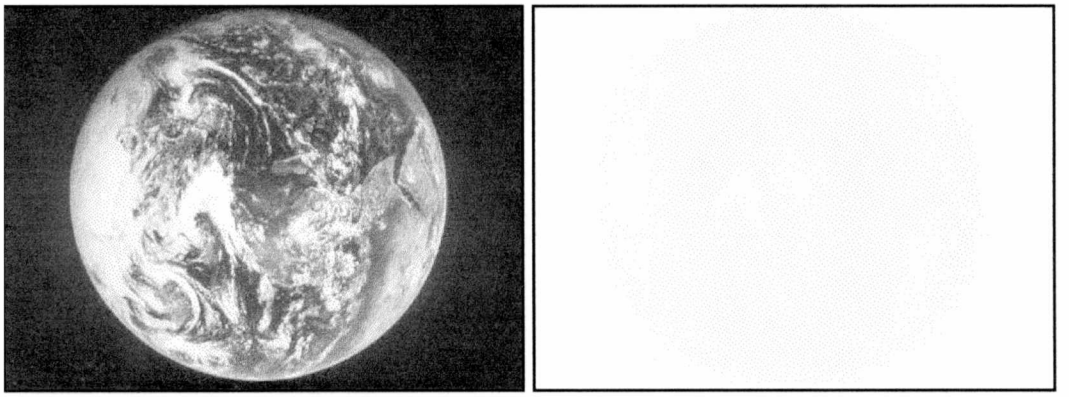
radio 28



tie6.bmp

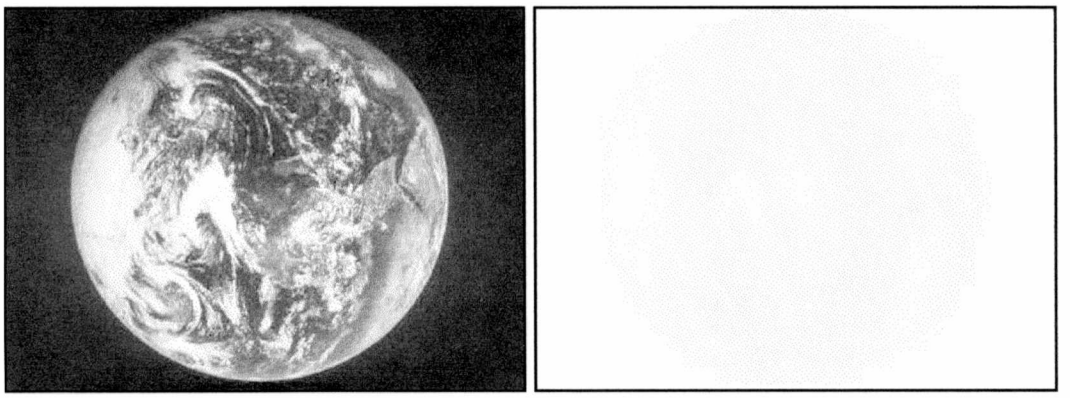
radio 36

Particionamiento fijo de 16x16



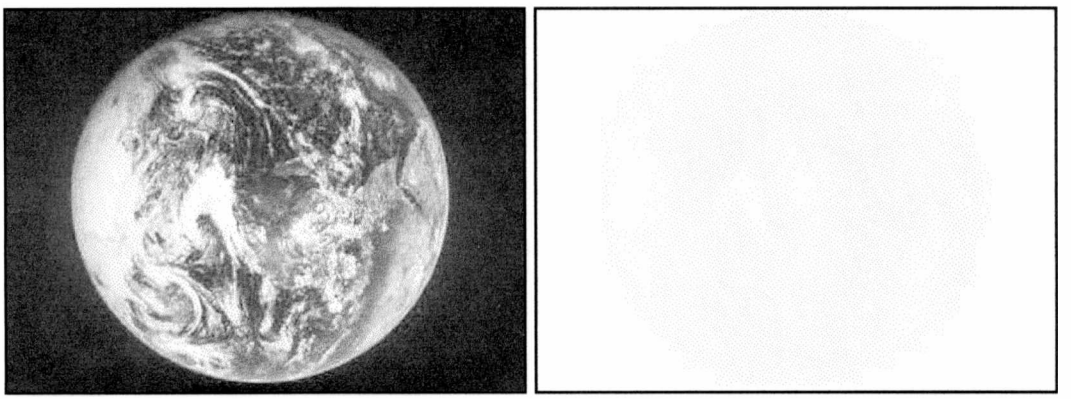
tie7.bmp

radio 11



tie8.bmp

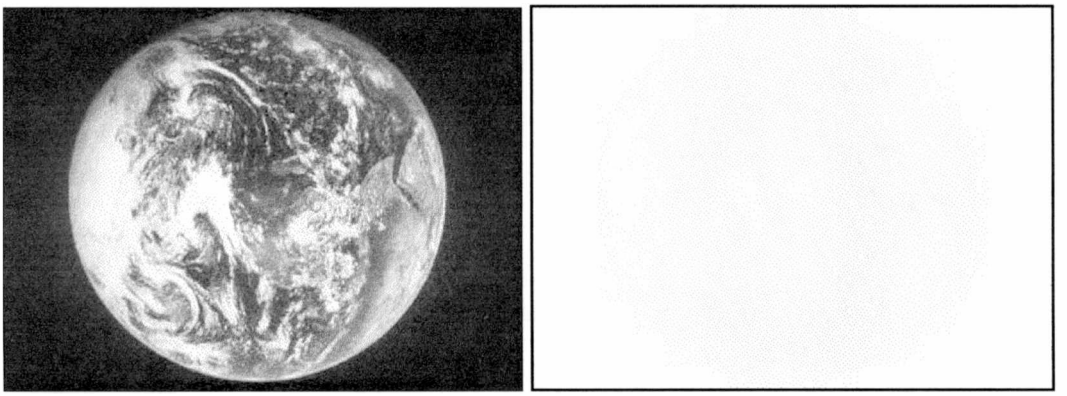
radio 18,3



tie9.bmp

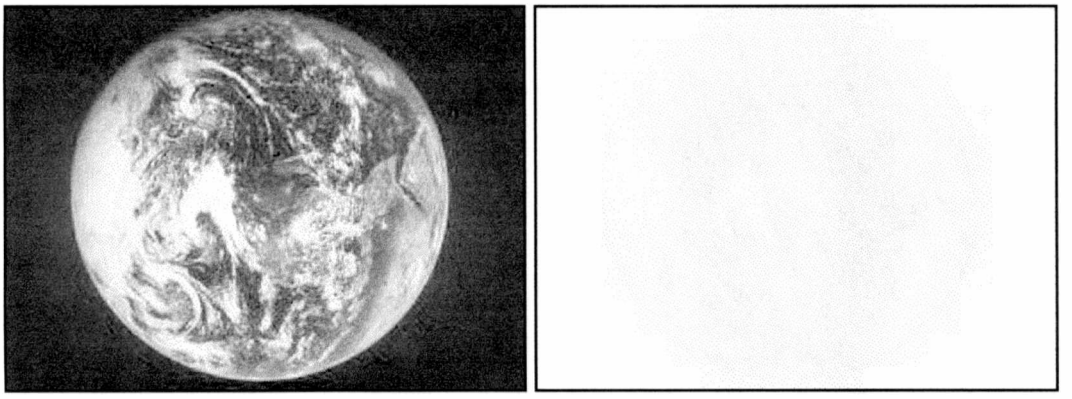
radio 24,8

Particionamiento fijo de 32x32



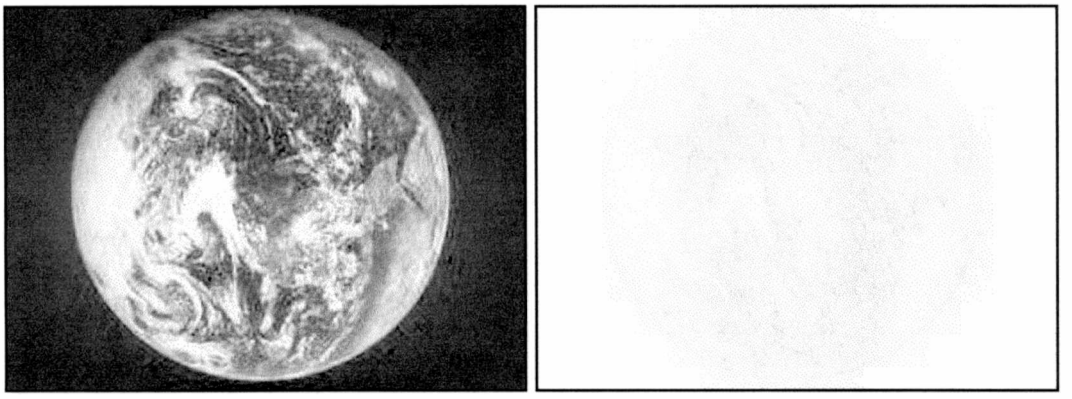
tie10.bmp

radio 18



tie11.bmp

radio 31

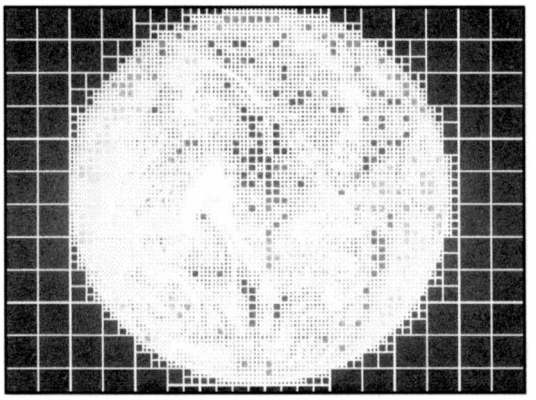


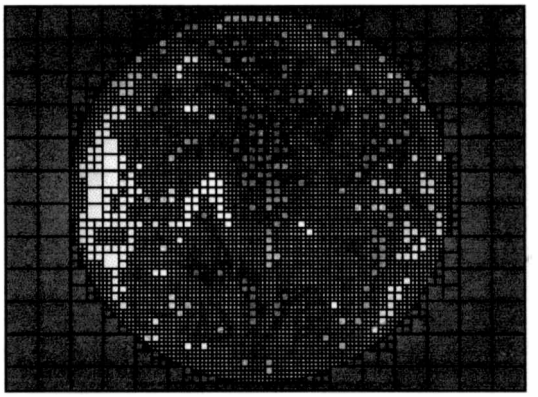
tie12.bmp

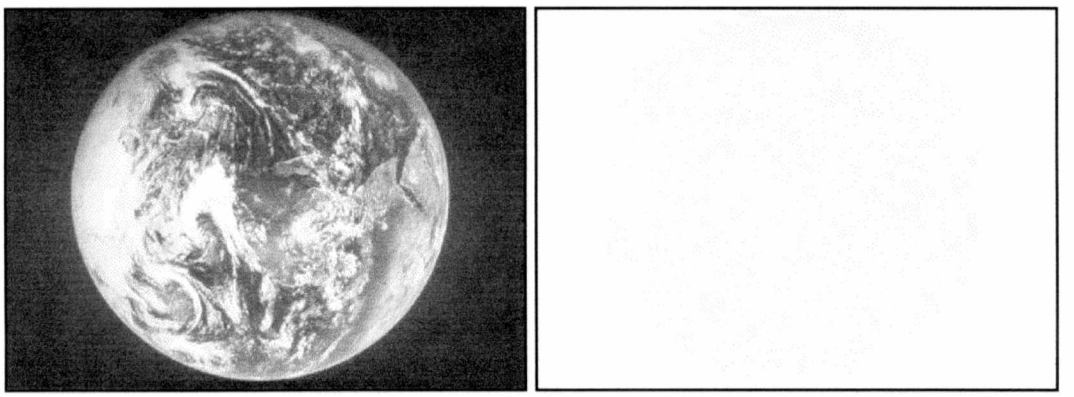
radio 43

Particionamiento variable con límites de 20,20,20

El particionamiento para esta imágen se logró estableciendo límites de 20,20,20 para los límites. Ambas imágenes diagramadas representan el mismo particionamiento pero se muestra el trazado en distinto color para poder apreciar los bloques del planeta y los del fondo.

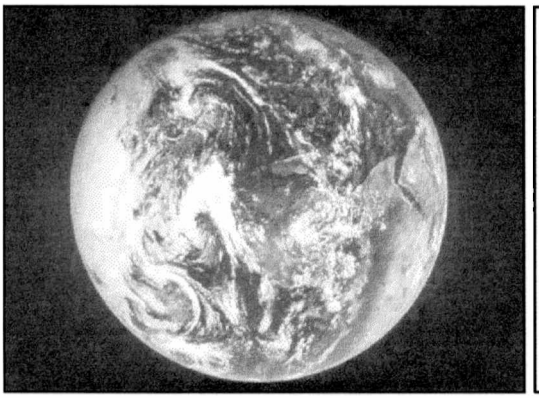






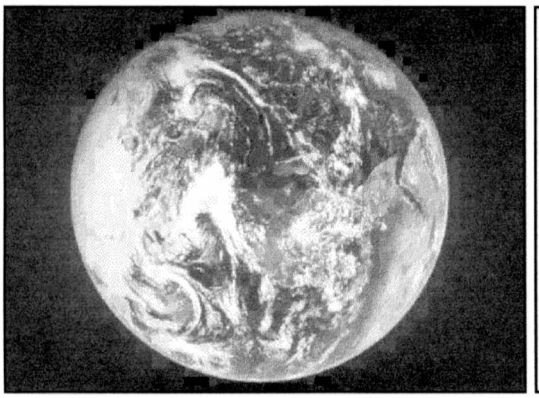
tie13.bmp

radio 4,8



tie14.bmp

radio 6,9

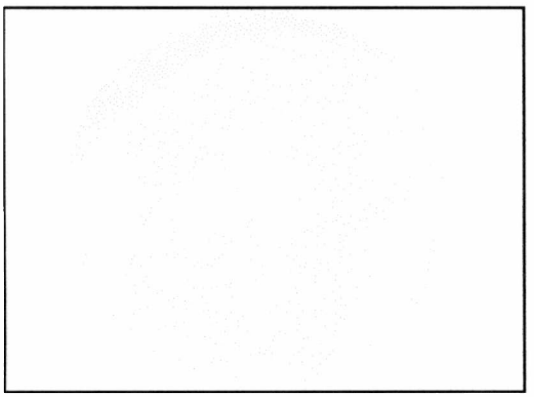


tie15.bmp

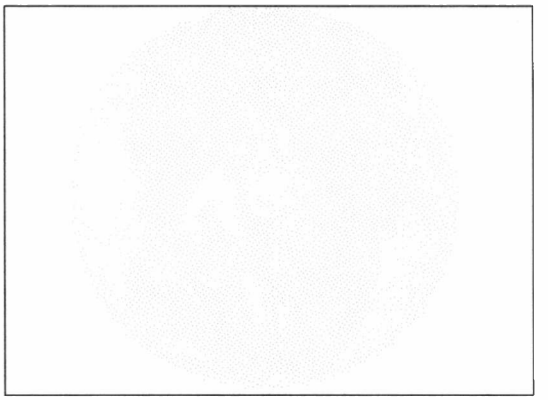
radio 10,57

Mejores y peores casos de las imágenes resta para tierra.bmp

Particionamiento 4x4

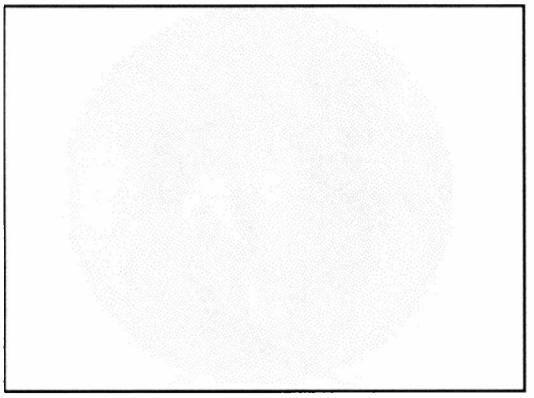


radio 4

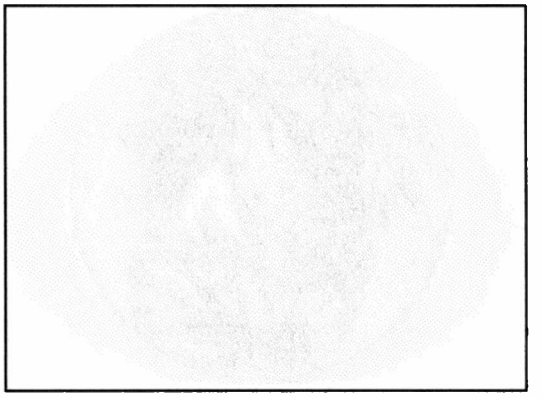


radio 7,2

Particionamiento 8x8

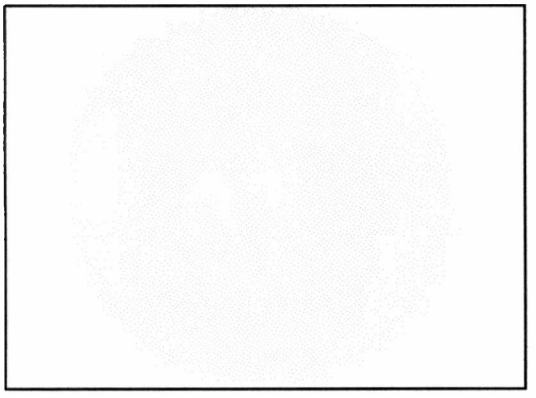


radio 18,4

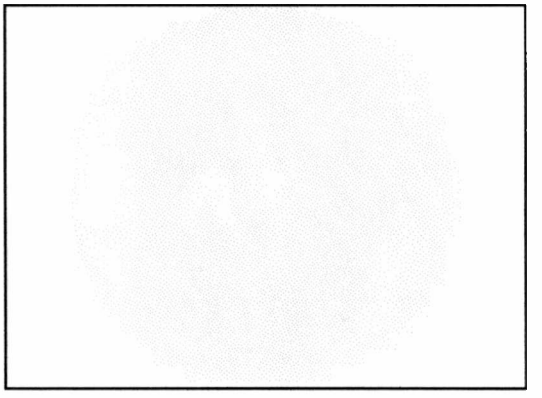


radio 36

Particionamiento 16x16

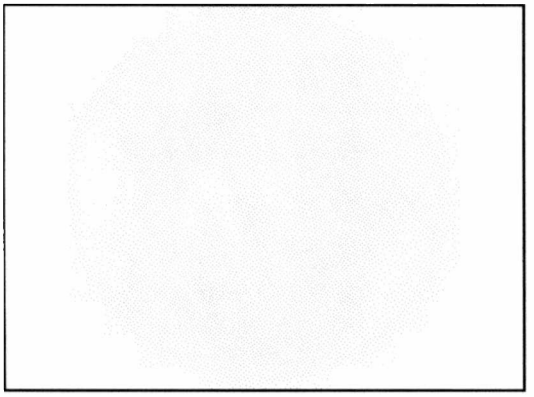


radio 11

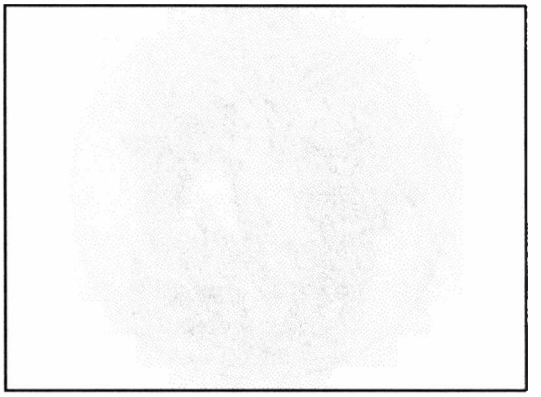


radio 24

Particionamiento 32x32

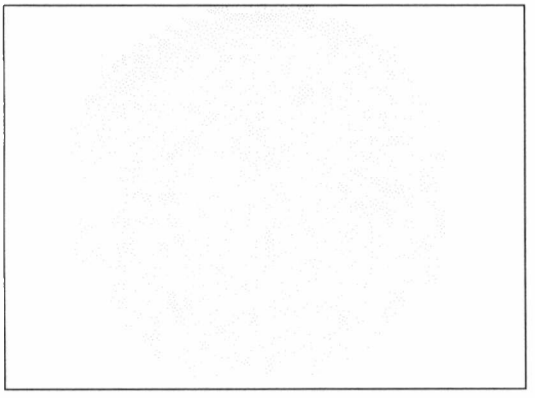


radio 18

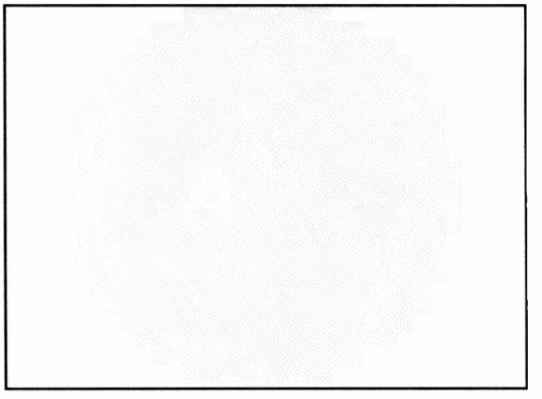


radio 43

Particionamiento variable



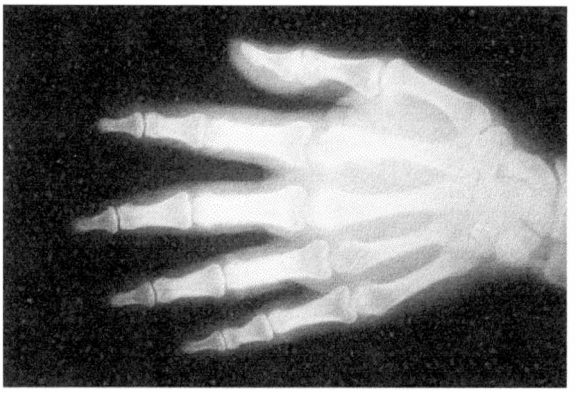
radio 4,8



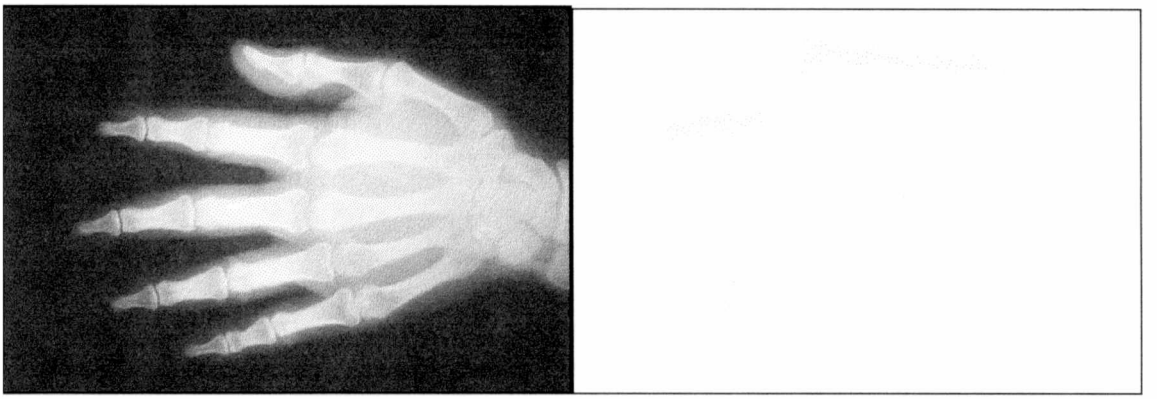
radio 10,5

Imágen Radiogr.bmp

Particionamiento fijo de 4x4

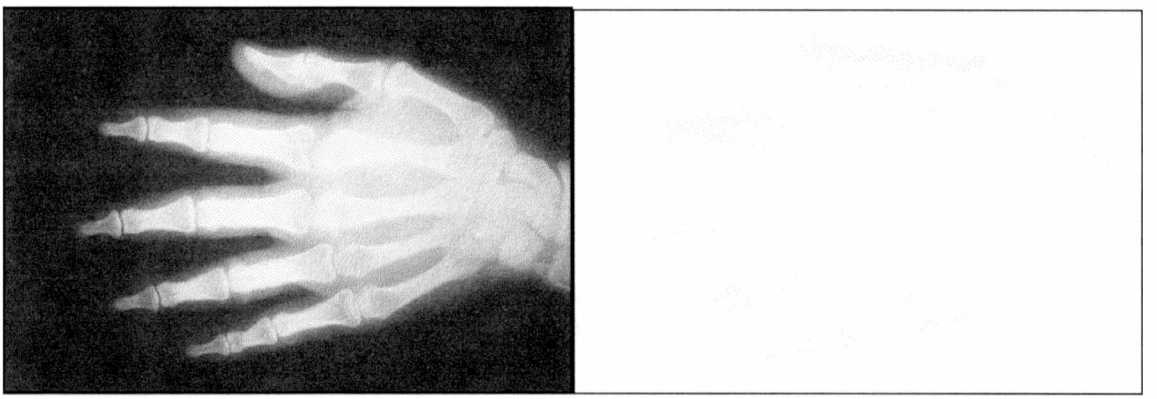


Imágen original



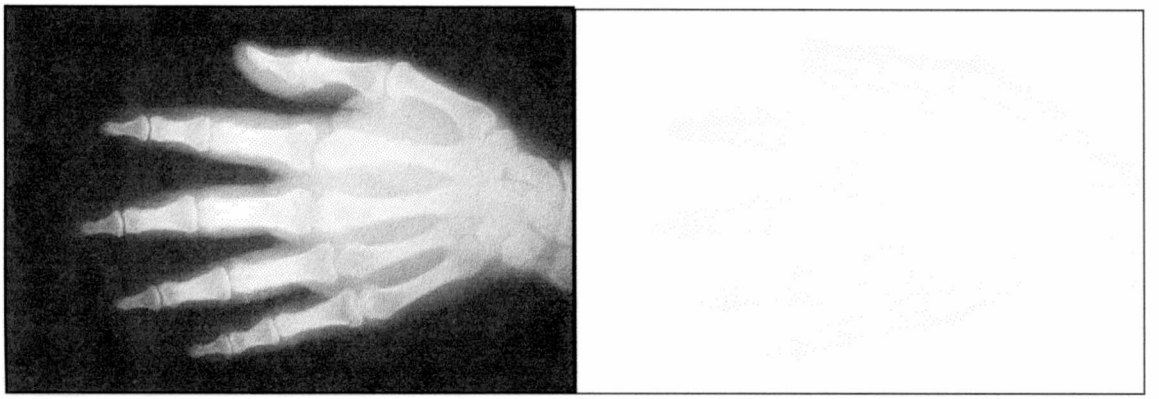
radi1.bmp

radio 10,2



radi2.bmp

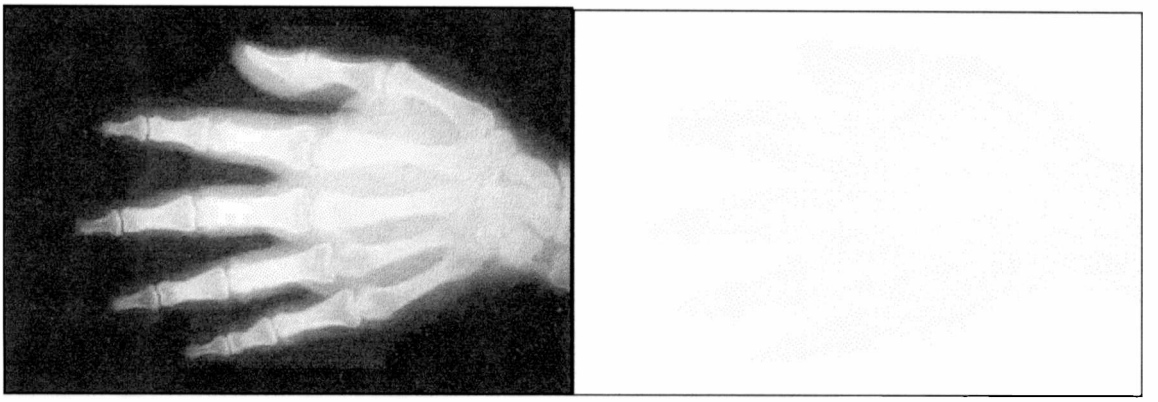
radio 13.4



radi3.bmp

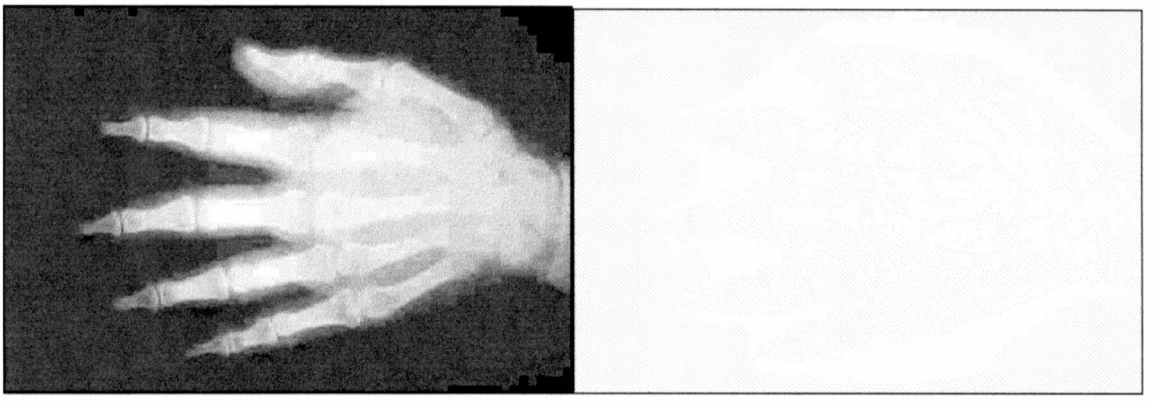
radio 15,4

Particionamiento fijo de 8x8



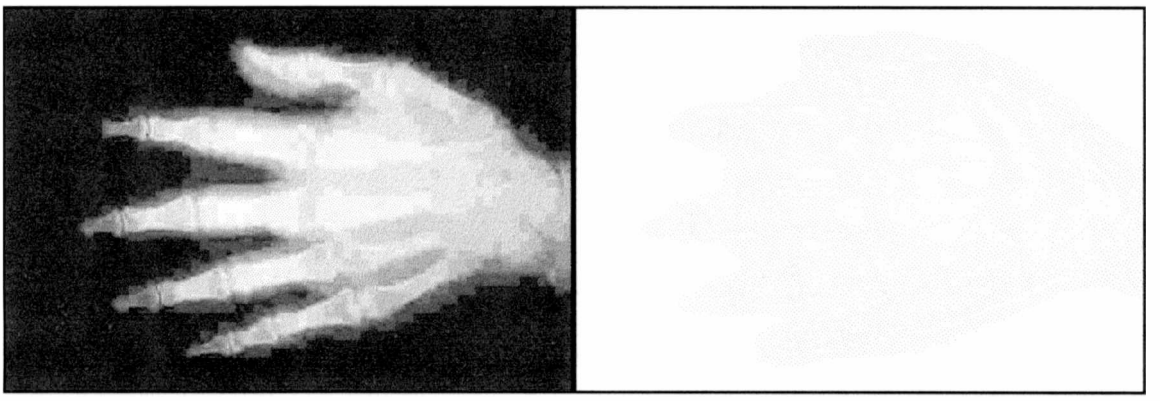
radi4.bmp

radio 40



radi5.bmp

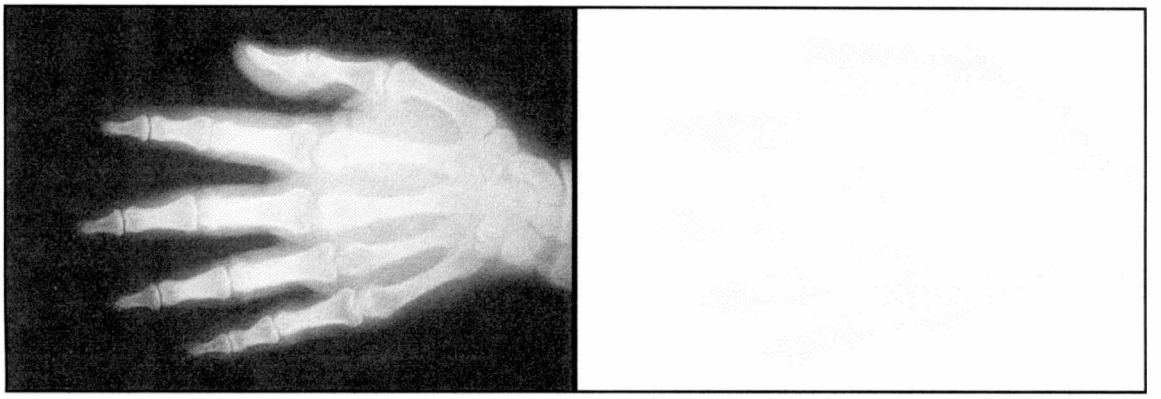
radio 51



radi6.bmp

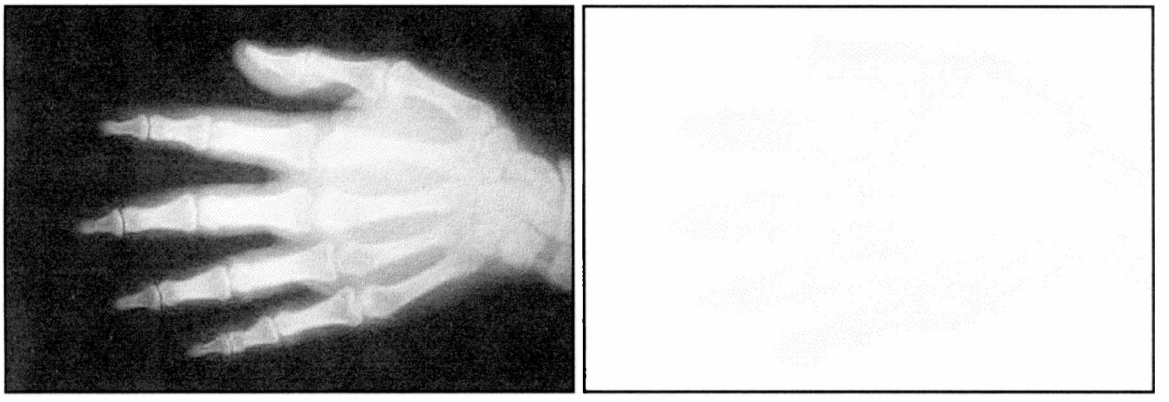
radio 57

Particionamiento fijo de 16x16



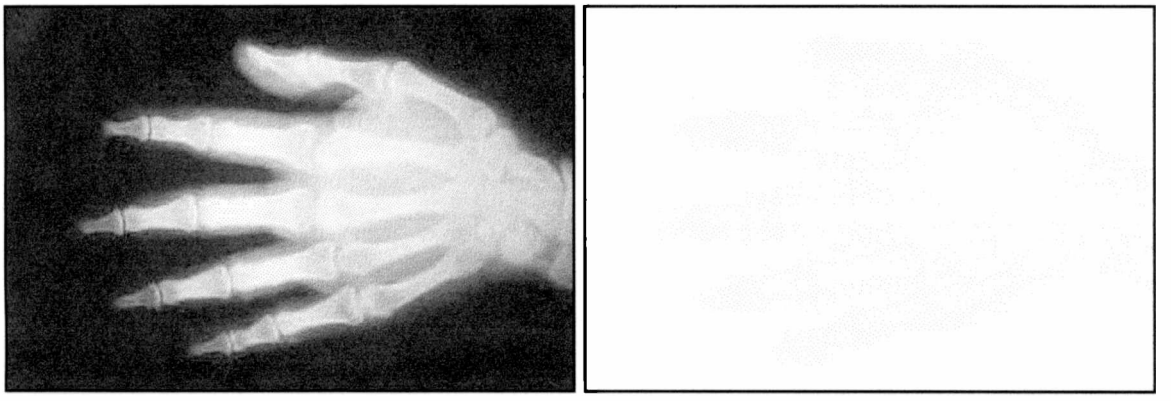
radi7.bmp

radio 32



radi8.bmp

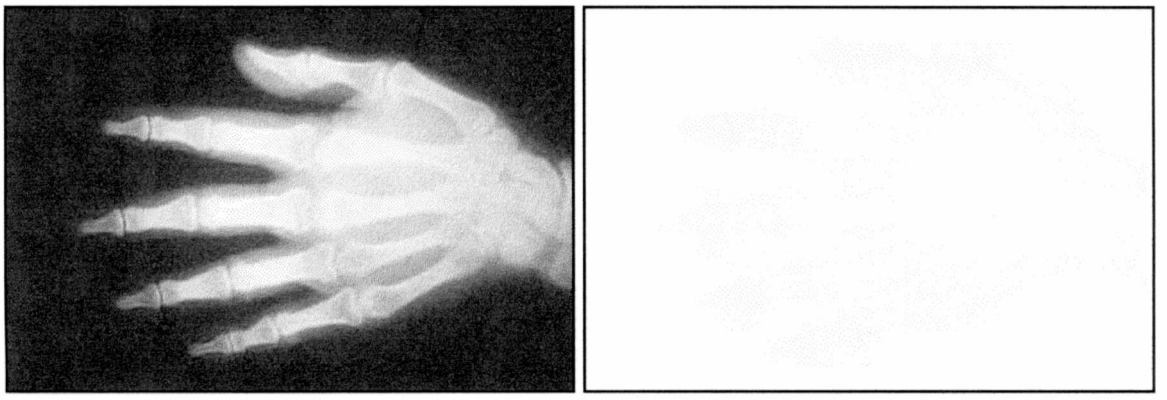
radio 48



radi9.bmp

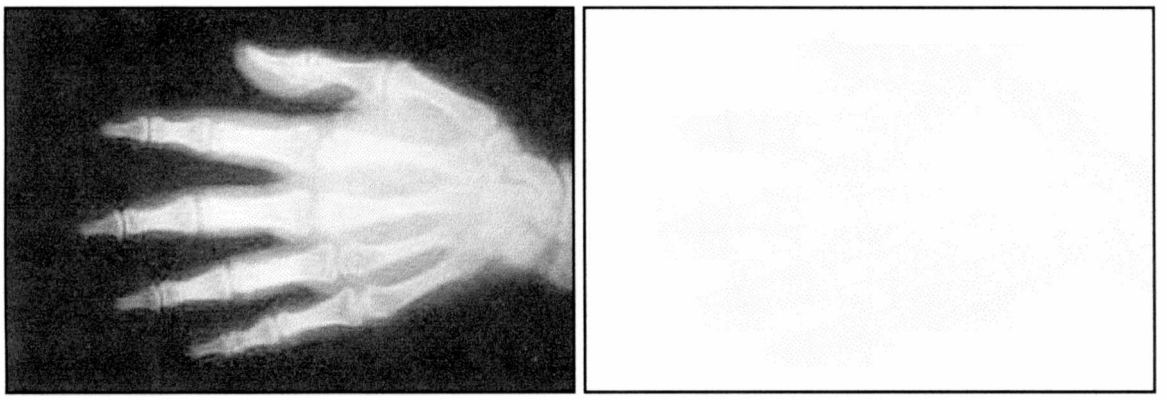
radio 61

Particionamiento fijo de 32x32



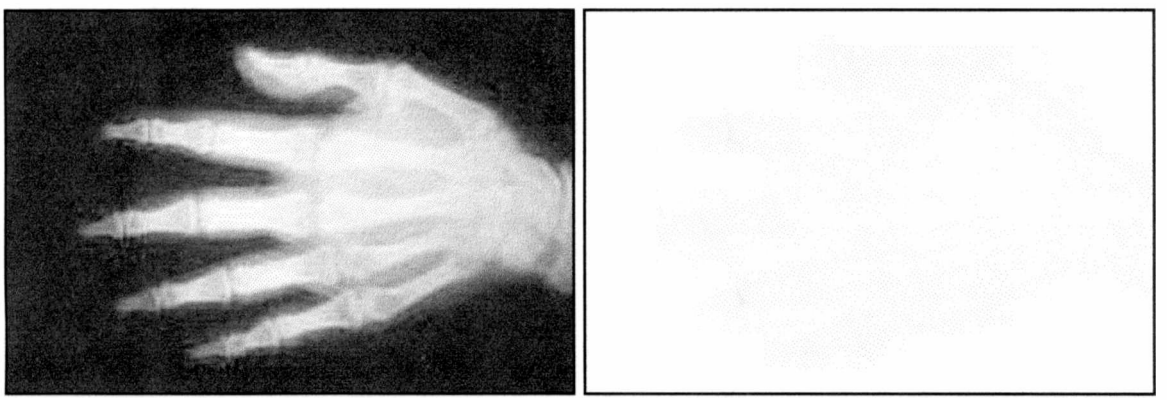
radi10.bmp

radio 50



radi11.bmp

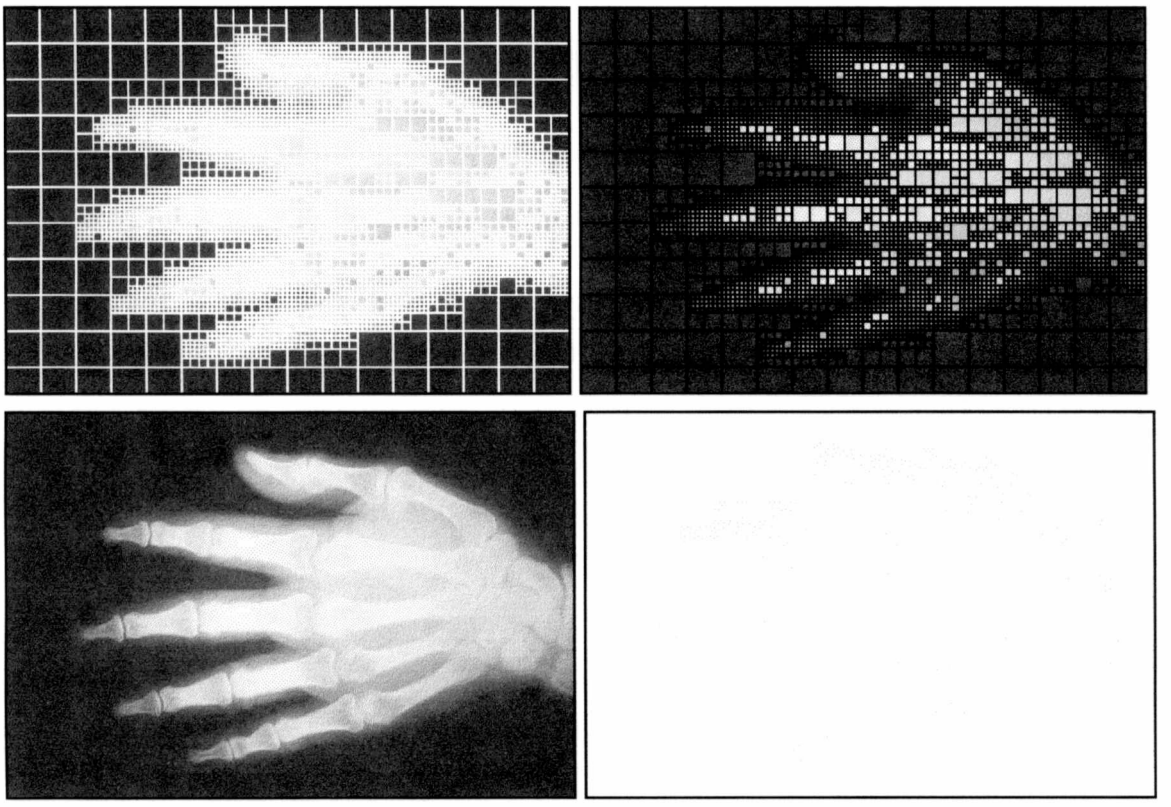
radio 78



radi12.bmp

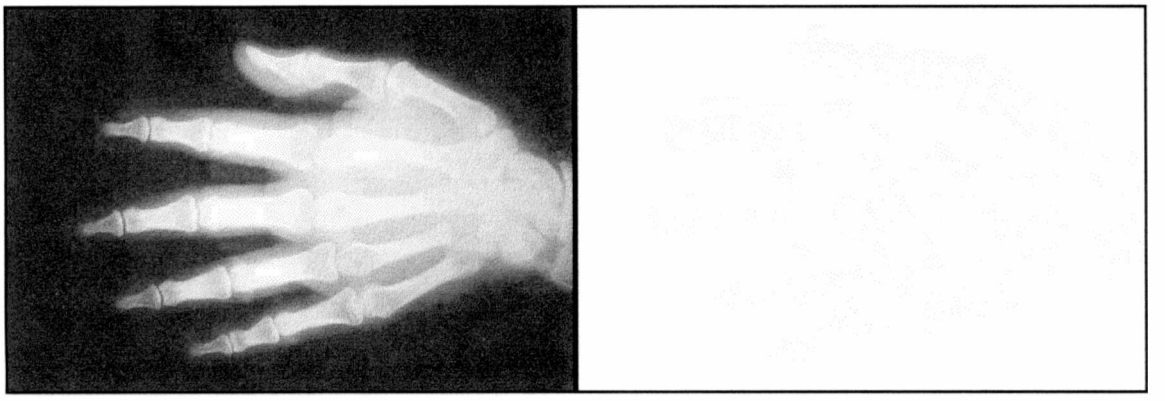
radio 101

Límites de particionamiento (10,10,10)



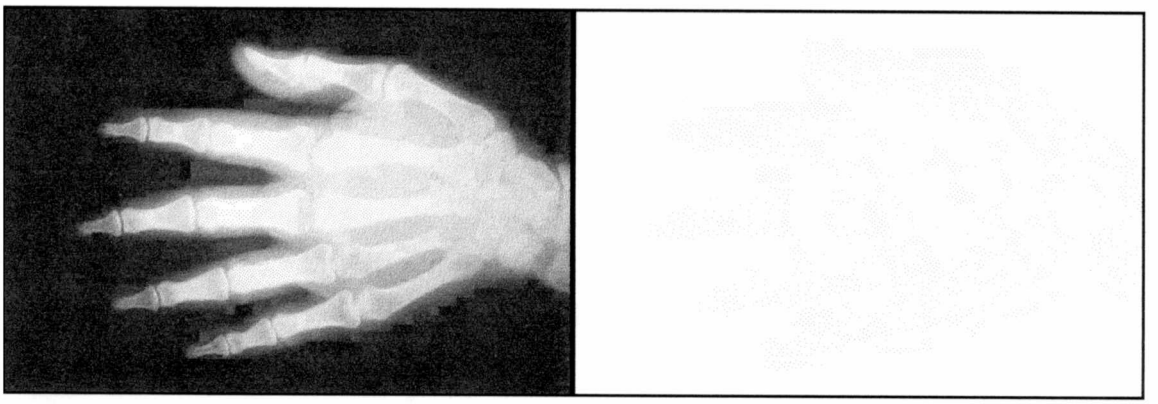
radi13.bmp

radio 14



radi14.bmp

radio 20

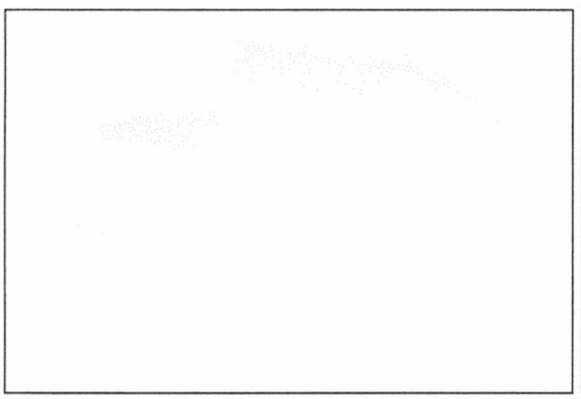


radi15.bmp

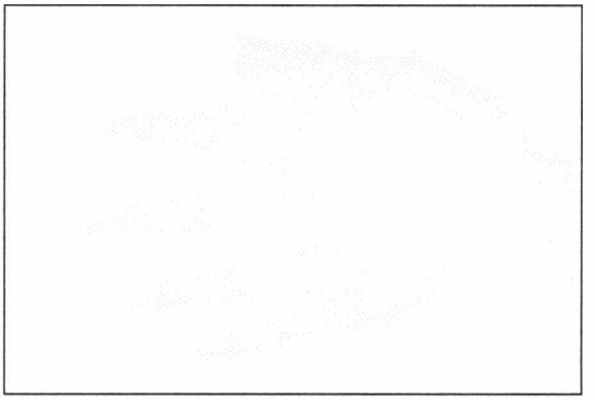
radio 24

Mejores y peores casos de las imágenes resta para radiogr.bmp

Particionamiento de 4x4

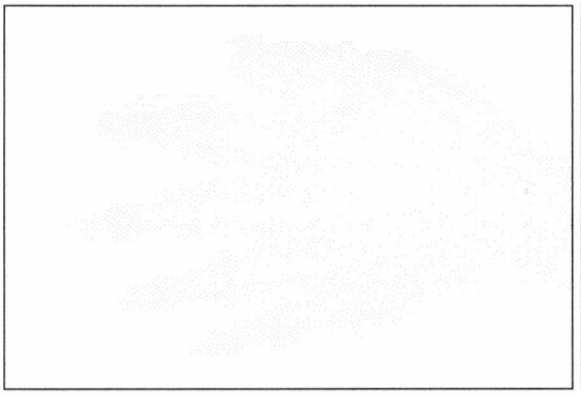


radio 10,2



radio 15,4

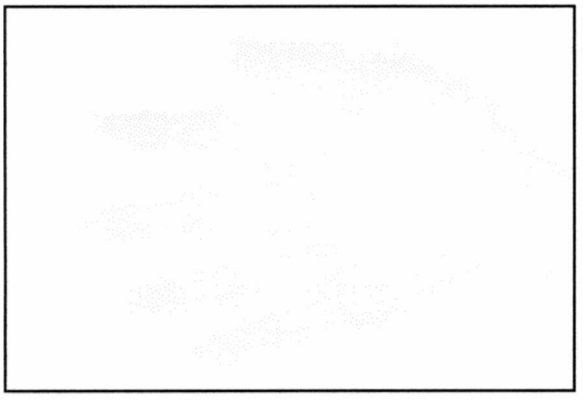
Particionamiento de 8x8



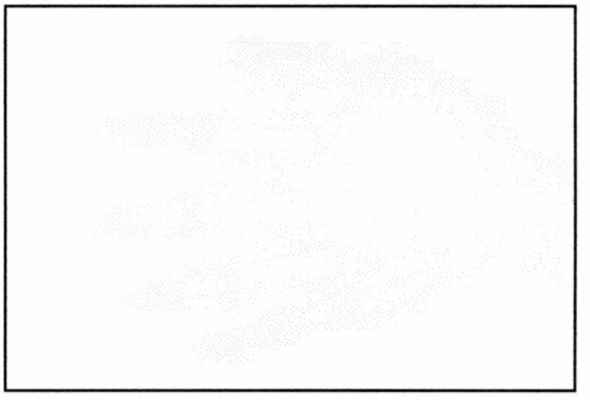
radio 40

radio 57

Particionamiento de 16x16

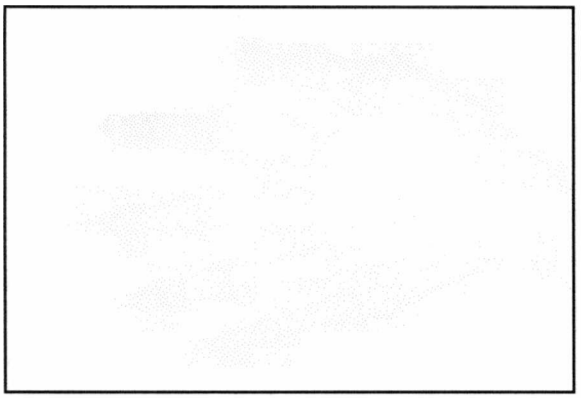


radio 32

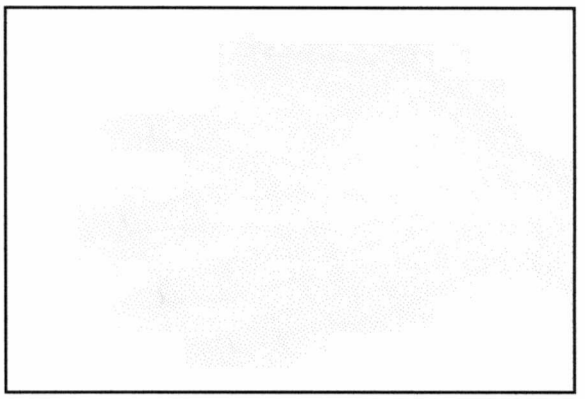


radio 61

Particionamiento de 32x32

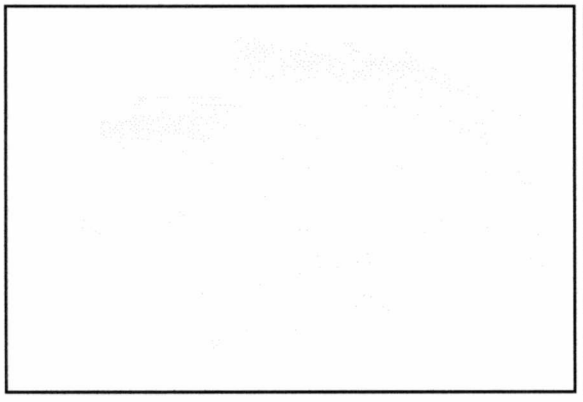


radio 50

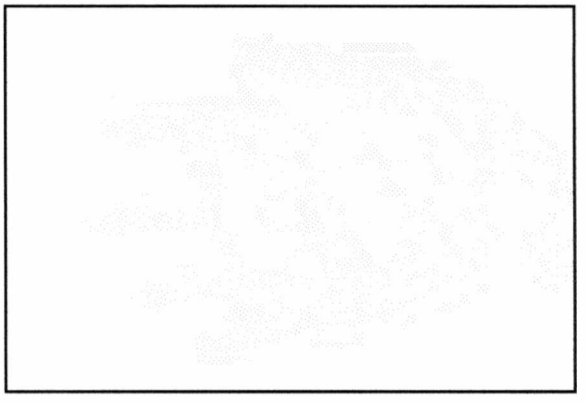


radio 101

Particionamiento variable



radio 14



radio 24

Programas entregados

Programas ejecutables

Cjpegn.exe y Djpegn.exe realizan la compresión y descompresión de una imagen con particionamiento fijo.

Cjpegn requiere :

Primer parámetro :nombre de la imagen BMP fuente.

Segundo parámetro :nombre del archivo resultante.

Tercer parámetro :longitud de bloque(4,8,16,32).

Cuarto parámetro :factor de calidad.

Djpegn requiere

Primer parámetro : nombre del archivo fuente (resultante de la compresión anterior).

Segundo parámetro : nombre de la imagen BMP resultante (imágen bmp con pérdida).

Cjv.exe y djv.exe realizan la compresión y descompresión de una imagen con particionamiento variable.

Cjv requiere

Primer parámetro : nombre de la imágen fuente (BMP).

Segundo parámetro : nombre del archivo resultante.

Tercer parámetro : threshold de decisión para 4 bloques de 4x4.

Cuarto parámetro : threshold de decisión para 4 bloques de 8x8.

Quinto parámetro : threshold de decisión para 4 bloques de 16x16.

Sexto parámetro : factor de pérdida para bloques de 4x4.

Séptimo parámetro : factor de pérdida para bloques de 8x8.

Octavo parámetro : factor de pérdida para bloques de 16x16.

Noveno parámetro : factor de pérdida para bloques de 32x32.

El factor de pérdida es un número que no tiene una relación proporcional con la pérdida producida pero que se utiliza para dar un peso a la matriz de cuantificación. Es decir que para aumentar la pérdida debemos aumentar este factor pero usar como medidas de distorción las ya estudiadas.

Djv requiere

Primer parámetro : nombre del archivo fuente.

Segundo parámetro : nombre de la imágen resultante.

Pqvar2.exe realiza un particionamiento de una imagen de acuerdo a los parámetros especificados.

Requiere

Primer parámetro : nombre de la imágen fuente (BMP).

Segundo parámetro : nombre del archivo resultante(un archivo con extención BMP).

Tercer parámetro : threshold de decisión para 4 bloques de 4x4.

Cuarto parámetro : threshold de decisión para 4 bloques de 8x8.

Quinto parámetro : threshold de decisión para 4 bloques de 16x16.

Sexto parámetro : 1 o 2 para el ancho en pixels del particionamiento.

Resta.exe realiza la resta entre imágenes en escalas de grises con las mismas dimensiones.

Requiere

Primer parámetro : nombre de la primer imagen fuente

Segundo parámetro : nombre de la segunda imagen fuente

Tercer parámetro : nombre de la imagen resultante

Bibliografía

[Cla95] Roger Clarke, *Digital Compression of Still Images and Video*, Academic Press, 1995.

[Dou87] E. Dougherty, C. Giardina, *Matrix Structured Image Processing*, Prentice-Hall Inc., 1987.

[Fol90] Foley, van Dam, Feiner, Huges, *Computer Graphics*, Addison-Wesley,1990.

[Glas95-1] Andrew Glassner, *Principles of Digital Image Synthesis. Vol. 1*, Morgan Kaufmann Publishers, 1996.

[Glas95-2] Andrew Glassner, *Principles of Digital Image Synthesis. Vol. 2*, Morgan Kaufmann Publishers, 1996.

[Gon92] R. Gonzales, R. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[Jai89] Anil K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall Inc., 1989.

[Nel91] *The Data Compresión Book*. Mark Nelson, Prentice Hall, 1991.

[Say96] Khalid Sayood, *Introduction to Data Compression*, Morgan Kaufmann Publishers, 1996.

[Baxe94-] Gregory A. Baxes, Digital Image Processing, John Wiley, 1994.

[Doug87] E.R.Dougherty,C.R. Giardina,"Matrix Structured Image Processing" Prentice-Hall Inc., 1987.

[Fole90] J. Foley, A. van Dam, S. Feiner, J.Hughes, "Computers Graphics ", Addison-Welsey Publishing Comp, 1990.

[Geva87] William Gevarter ,"Máquinas inteligentes",Ediciones Diaz Santos 1987

[Gonz92] Rafael C: Gonzáles, Richard E. Woods, "Digital Image Procesing", Addison-Wesley Publishing Comp , 1992.

[Heer91] D.W. Heermann , A..N. Burkitt "Paralell Algorithms in Computational Science", Springer -Verlag, 1991.

[Huss91] Zahid Hussain,"Digital Image Processing", Ellis Horwood Limited, 1991.

[IEEE] Colección de "Computers Graphics and Applications", IEEE

[IEEE] Colección de "IEEE Transaccions on Computers", IEEE

[Jain89] Anil K. Jain, "Fundamentals of Digital Image Procesing", Prentice Hall Inc. 1989.

[Jain95] Anil K. Jain, R. Kasturi, B. Schunk, "Machine Vision", MacGrawHill,. 1995.

[Niem90]Heinrich Niemann,"Pattern Analysis and Understanding", Springer-Verlag, 1990.

[Pao89] Yoh-Han Pao. Adaptive Pattern Recognition and Neural Networks", Addison-Wesley,1989

[Prat78] W.Pratt, "Digital Image Processing",Wiley,1992

[Rimm92]Rimmer, "Supercharged bitmapped graphics"

[Scha92] R. Schalkoff,"Patern Recognition ",Wiley,1992.

[Zavi91] B. Zavidovique, P. Wendel (Ed)," Computer Architecture for Machine Perception".,CAMP1 ,1991