Sección de Informática Gráfica VALENCIA | Computer Graphics Group

UNIVERSIDAD POLITECNICA DE VALENCIA

Técnicas de compresión

Introducción

Técnicas de compresión

Compresión sin pérdida

Compresión con pérdida

Introducción

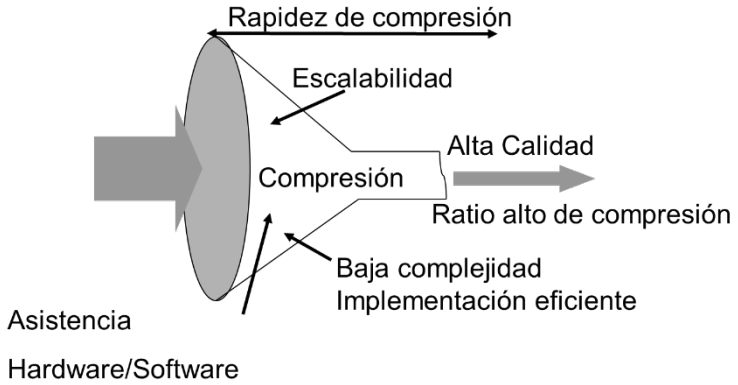
- Hablar de imagen, vídeo y audio digital es hablar de compresión
- Los conceptos fundamentales son:
  - Requerimientos de almacenamiento y ancho de banda: medios discretos y continuos
  - Compresión básica: Entropía, Fuente, Híbrido
  - Técnicas de compresión:
    - Imagen: JPEG
    - Vídeo H.26x, MPEG 1/2/4
    - Audio G.7xx

Introducción

- ¿Por qué comprimir?
  - La información sin comprimir necesita una capacidad de almacenamiento considerable
  - Útil para almacenar imágenes estáticas
  - Imprescindible para enviar audio y vídeo, sin compresión no hay suficiente ancho de banda para enviar 30 imágenes por segundo

Introducción

- Conceptos:
  - Ratio de compresión: tamaño del fichero original dividido por el del comprimido
  - Calidad de datos: hay dos tipos de compresión, con pérdida y sin pérdida
  - Velocidad de compresión: tiempo necesario para comprimir/descomprimir
  - Requerimientos:



Introducción

- Requerimientos de almacenamiento de un A4:

Resolución (dpi)	Bitonal (MB)	Escala grises (MB)	Color (MB)
Bits por pixel	1	4-6	32-128
200	0.48	1.9-7.7	15-61
300	1.09	4.4-17.4	35-140
400	1.93	7.7-30.9	62-247

Introducción

- Medios discretos – Tamaño por página

Medio	Tamaño
Texto	9.4KB
Gráficos	2.8KB
Bitmap	300-900KB
A4	15-247MB

Introducción

- Medios continuos – Ancho de Banda

Medio	Ancho de Banda
Audio digital telefonía	64Kb/s
Audio estéreo calidad CD	1.34Mb/s
Video PAL	176Mb/s
Video HDTV	936Mb/s

Introducción

- Ejemplo de compresión de vídeo:
- Si creamos un vídeo a pantalla completa de 10”:
  - a 25 frames/seg * 10 = 250 frames
  - a 1280x720 = 0.88 Mega pixels per frame
  - True color = 3 bytes por pixel
  - 250 * 0.88 * 3 = 659MB
  - Ocupa demasiado espacio.
  - Transferir 659MB en 10” ocupa mucho ancho de banda
- Además si fuera 1’-> 3,95 GB y 1 hora-> 237 GB

Introducción

- Por lo tanto, es necesario comprimir: se realiza tanto por hardware como por software
- El vídeo digital es muy propicio para ser comprimido:
  - En general, se realizan pocos cambios entre fotogramas consecutivos
  - Se consiguen muy buenos ratios de compresión
- En la práctica, 10" de vídeo ocupan 14 MB o menos, 1h30' en FullHD 4 GB

Técnicas de compresión

- Entropía: Utiliza la redundancia, sin pérdida
- Fuente: Utiliza el contexto semántico, generalmente con pérdida
- Híbrido: Combina ambas técnicas

Compresión sin pérdida

- Compresión sin pérdida:
  - Siempre es posible comprimir o descomprimir los datos obteniendo una copia exacta del original
  - Los datos se almacenan de forma más eficiente, sin eliminar información
  - Hay 4 tipos:
    - Codificación Run-length (RLE)
    - Codificación Huffman
    - Codificación Aritmética
    - Esquemas basados en diccionarios

Compresión sin pérdida: Entropía

- Entropía
  - Los datos se consideran como una secuencia digital, ignorando la semántica
  - De acuerdo con el teorema de Shannon (1948) la longitud óptima de un código para un símbolo es:

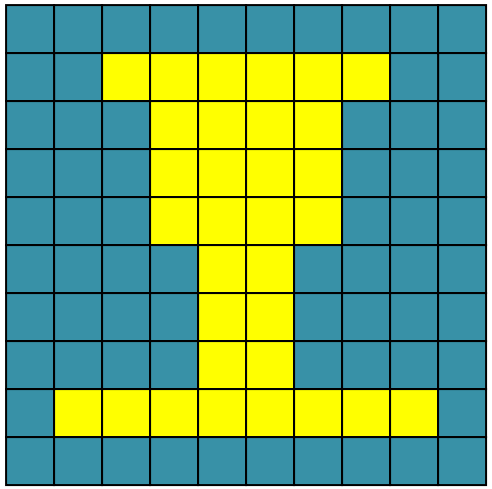
    $-log_b P$
  - donde **b** es el número de símbolos empleados para crear los códigos de salida y **P** es la probabilidad del símbolo de entrada
  - Se utilizan códigos más cortos para los símbolos más frecuentes y códigos más largos para los más infrecuentes
    - Por ejemplo: en Español la A se utiliza mucho, pues se le asigna un código más corto que para la K

Técnicas de compresión: RLE

- Run Length Encoding (RLE)
  - Generalización de la técnica Zero Supression
  - Las secuencias de datos se almacenan como un único valor y un contador, en vez de una secuencia
- Ejemplo:
  - WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWW
WWWWWWWWWWWWWWWWWWWWBWWWWWWWWWWWWWW
  - Se codifica como: 12WB12W3B24WB14W
- Para evitar confusiones se utilizan flags antes del contador
- Ejemplo: ABCCCCCCCCCDEFGGG
  - Se codifica como: ABC!8DEFGGG (! Es una flag)

Compresión sin pérdida: RLE

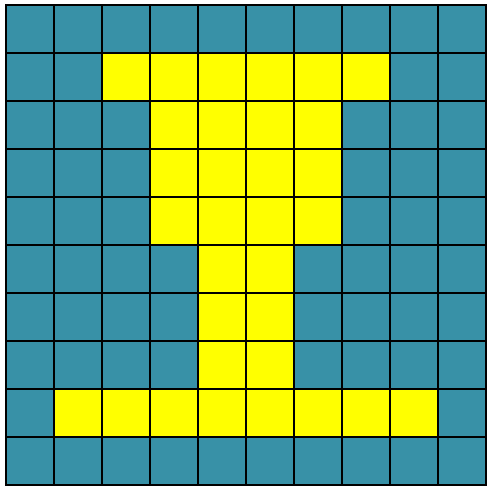
- RLE de una dimensión:



10'0'		
2'0'	6'1'	2'0'
3'0'	4'1'	3'0'
3'0'	4'1'	3'0'
3'0'	4'1'	3'0'
4'0'	2'1'	4'0'
4'0'	2'1'	4'0'
4'0'	2'1'	4'0'
1'0'	8'1'	1'0'
10'0'		

Compresión sin pérdida: RLE

- RLE de dos dimensiones:



10'0'		
2'0'	6'1'	2'0'
3'0'	4'1'	3'0'
AGAIN		
AGAIN		
4'0'	2'1'	4'0'
AGAIN		
AGAIN		
1'0'	8'1'	1'0'
10'0'		

Compresión sin pérdida: Huffman

- Codificación Huffman asigna códigos óptimos en función de:
  - Número de símbolos diferentes
  - Probabilidad de cada símbolo
- Los códigos más cortos a los más probables
- La compresión de este algoritmo varía con el algoritmo particular y el tipo de imagen, pero pocas veces pasa la compresión de 8 a 1.
- Tiende a comportarse peor con ficheros que contienen cadenas largas de píxeles idénticos, que suelen comprimirse mejor con el RLE u otro tipo.
- Necesita estadísticas precisas de cuantas veces aparece un símbolo en el fichero original
- Por ello, normalmente da dos pasadas, en la primera se crea el modelo estadístico y en la segunda los códigos de compresión variable.

Compresión sin pérdida: Huffman

- Tomamos un texto corto como ejemplo:

"ata la jaca a la estaca"

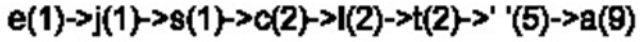
1) Contamos las veces que aparece cada carácter y hacemos una lista enlazada:

' '(5), a(9), c(2), e(1), j(1), l(2), s(1), t(2)

2) Ordenamos por frecuencia de menor a mayor

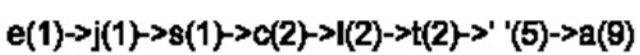
e(1), j(1), s(1), c(2), l(2), t(2), ' '(5), a(9)

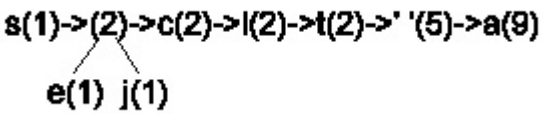
3) Consideremos ahora que cada elemento es el nodo raíz de un árbol.

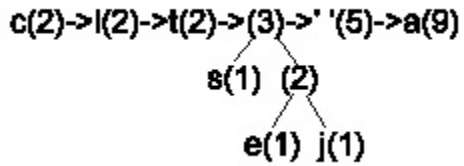


Compresión sin pérdida: Huffman

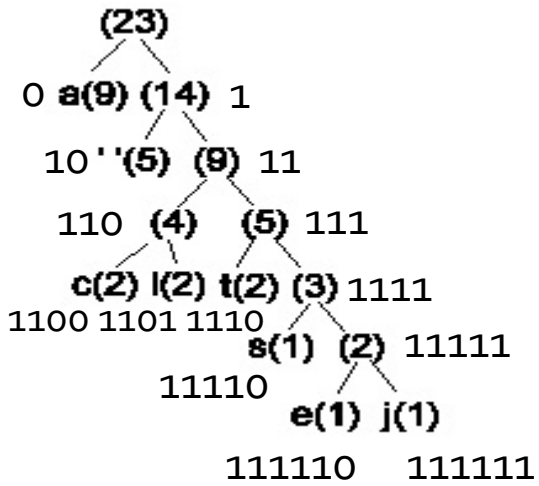
4) Fundimos los dos primeros nodos (árboles) en un nuevo árbol, sumamos sus frecuencias y lo colocamos en el lugar correspondiente:







- El resultado final es:



Compresión sin pérdida: Huffman

5) Asignamos los códigos, es una regla arbitraria:

a	' '	c	l	t	s	e	j
0	10	1100	1101	1110	11110	111110	111111

6) Codificamos el texto:

a	t	a	' '	l	a	' '	j	a	c	a	' '	a	' '	l	a	' '	e	s	t	a	c	a
0	1110	0	10	1101	0	10	111111	0	1100	0	10	0	10	1101	0	10	111110	11110	1110	0	1100	0

7) Agrupamos en bytes:

01110010	11010101	11111011	00010010	11010101	11110111	10111001	10000000
0x72	0xD5	0xFB	0x12	0xD5	0xF7	0xb9	0x80

- En total ocho bytes, y el texto original tenía 23

Compresión sin pérdida: codificación aritmética

- La codificación aritmética:
  - Al igual que Huffman emplea códigos cortos para modelizar lo que ocurre frecuentemente y códigos más largos para lo infrecuente
  - La principal debilidad de Huffman está en que codifica los símbolos de entrada de uno en uno
  - Sin embargo, la codificación aritmética codifica secuencias de símbolos de entrada a la vez:
    - No hay una correspondencia uno a uno entre los símbolos de entrada y las palabras codificadas
  - Por ello, es más lento que Huffman pero con mejores ratios de compresión

Compresión sin pérdida: codificación aritmética

- Codifica cada símbolo utilizando el anterior
- Mapea cada secuencia diferente de píxeles de una región en una línea de números imaginaria que va entre 0 y 1
- Método general:
  - Cada símbolo divide el intervalo previo
  - Los intervalos se escalan
- No permite la decodificación en cualquier parte del fichero
- Puede obtener compresiones de 100 a 1

Compresión sin pérdida: codificación aritmética

- Ejemplo:
- redes:

Símbolo	Frecuencia	Probabilidad	Rango
d	1	1/5=0.2	[0.0,0.2[
e	2	2/5=0.4	[0.2,0.6[
r	1	1/5=0.2	[0.6,0.8[
s	1	1/5=0.2	[0.8,1.0[

- Codificación:
  - Inferior = AI + (AS-AI)*NI
  - Superior = AI + (AS-AI)*NS

A=Antiguo N=Nuevo I=Inferior S=Superior

Compresión sin pérdida: codificación aritmética

- Inferior = AI + (AS-AI)*NI - Superior = AI + (AS-AI)*NS
- redes:

r [0.6,0.8[

e [0.2,0.6[I=0.6+(0.8-0.6)*0.2=0.64

S= 0.6+(0.8-0.6)*0.6=0.72 [0.64,072]

d [0.0,0.2[I=0.64+(0.72-0.64)*0.0=0.64

S=0.64+(0.72-0.64)*0.2=0.656 [0.64,0.656]

e [0.2,0.6[I=0.64+(0.656-0.64)*0.2=0.6432

S=0.64+(0.656-0.64)*0.6=0.6496 [0.6432, 0.6496]

s [0.8,1.0[I=0.6432+(0.6496-0.6432)*0.8=0.64832

S=0.6432+(0.6496-0.6432)*1.0=0.64960 [0.64832, 0.64960]

Convertir a binario para finalizar la codificación

Compresión sin pérdida: codificación aritmética

- Decodificación:

[0.64832, 0.64960]
Límite inferior cae en la r
(LIA – LIN)/RangoN
(0.64832-0.6)/0.2 = 0.2416 -> e
(0.2416-0.2)/0.4 = 0.104 -> d
(0.104-0.0)/0.2 = 0.52 -> e
(0.52-0.2)/0.4 = 0.8 -> s
(0.8-0.8)/0.2 = 0.0 FIN

Símbolo	Frecuencia	Probabilidad	Rango
d	1	1/5=0.2	[0.0,0.2[
e	2	2/5=0.4	[0.2,0.6[
r	1	1/5=0.2	[0.6,0.8[
s	1	1/5=0.2	[0.8,1.0[

Compresión sin pérdida: Diccionarios

- No requiere conocimiento previo sobre las probabilidades de los símbolos
- Asigna códigos de longitud fija a secuencias de símbolos de longitud variable: No hay una correspondencia uno a uno entre los símbolos de entrada y los códigos.
- La clave de este método es que es posible construir automáticamente un diccionario de secuencias previamente vistas en el texto que va a ser comprimido.
- Empezando con una tabla simple construye una tabla más efectiva, que es adaptativa.
- El esquema explota la redundancia de los patrones incluso no contiguos que encuentra en el análisis de la imagen.

Compresión sin pérdida: Diccionarios

- El diccionario no tiene que ser transmitido con el texto comprimido, puesto que el descompresor puede construirlo de la misma manera que lo hace el compresor
- El diccionario comienza con 256 entradas, si asumimos escala de grises
- Cada vez que se encuentra una secuencia no vista en el diccionario, se almacena una secuencia más larga que consiste en esa secuencia seguida de un nuevo carácter.
- La salida será un índice a esta tabla.

Compresión sin pérdida: Diccionarios

- Ejemplo: tenemos una imagen de 4x4 píxeles en escala de grises (8 bits por pixel)

IMAGEN

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Índice Dicccionario	Valor
0	0
1	1
.	.
255	255
256	-
511	-

Compresión sin pérdida: Diccionarios

- Conforme se van examinando los píxeles, las secuencias de grises que no están en el diccionario se asignan a nuevos índices:
  - 39 está en el diccionario? Sí
  - 39-39 está en el diccionario? No, se añade 39-39 a la posición 256

IMAGEN

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Índice Dicccionario	Valor
0	0
1	1
.	.
255	255
256	39-39
511	-

Compresión sin pérdida: Diccionarios

- Secuencia actual: CR (1ª columna)
- Pixel siendo procesado: P (2ª columna)
- Secuencia concatenada: CS=CR+P

CR=vacío
Repetir
  P=siguiente pixel
  CS=CR+P
  Si CS está en el diccionario
    Salida vacía
    CR=CS
  Sino
    Salida D(CR) (3ªcol)
    Añadir CS a D (4ª-5ª)
    CR=P

IMAGEN

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Currently Recognized Sequence	Pixel Being Processed	Encoded Output	Dictionary Location (Code Word)	Dictionary Entry
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

Compresión sin pérdida: Diccionarios

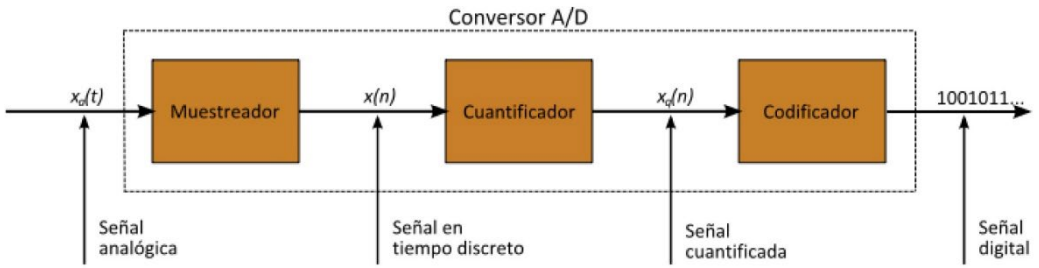
- Tiene unos ratios de compresión entre 1:1 y 3:1, aunque en imágenes de patrones muy marcados puede llegar a 10:1
- En imágenes con ruido se comporta mal, ya que el ruido es lo contrario a los patrones. Se pueden emplear entonces técnicas de supresión del ruido como eliminar los bits menos significativos o hacer un promedio local
- Este esquema lo emplean el PDF, GIF y TIFF

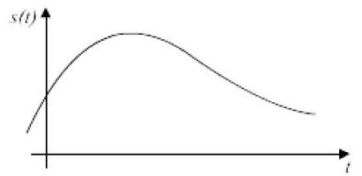
Compresión con pérdida

- La compresión de la fuente:
  - Tiene en cuenta la semántica de los datos
  - La cantidad de compresión depende del contenido
  - Con pérdida
  - Se utiliza mucho para realizar streaming
- Hay cuatro tipos principales:
  - Predicción: DPCM, ADPCM, DM, MC
  - Transformación: FFT, DCT
  - Por capas (progresivo): Posición de bit, subsampling, sub-band coding
  - Cuantificación vectorial

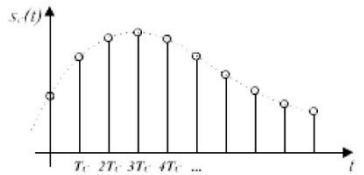
Compresión con pérdida

- La información a comprimir ya sea audio o vídeo, se considera en el espacio de la frecuencia
- Muestreo es la cantidad de veces por segundo que se mide el valor de la señal





Señal Analógica



Señal En tiempo Discreto

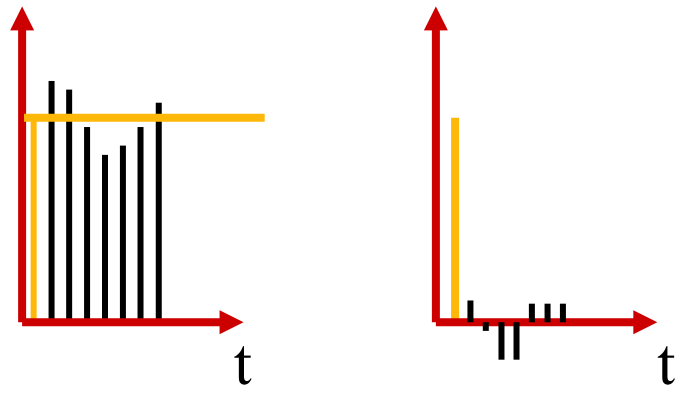
Compresión con pérdida: Predicción

- Predicción:
  - La señal actual se puede predecir en función de las anteriores
  - Audio
    - PCM: Pulse Code Modulation
    - DPCM: Differential PCM
    - ADPCM: Adaptive DPCM
    - DM: Delta Modulation
  - Vídeo
    - MC: Motion Compensation

Compresión con pérdida: Predicción

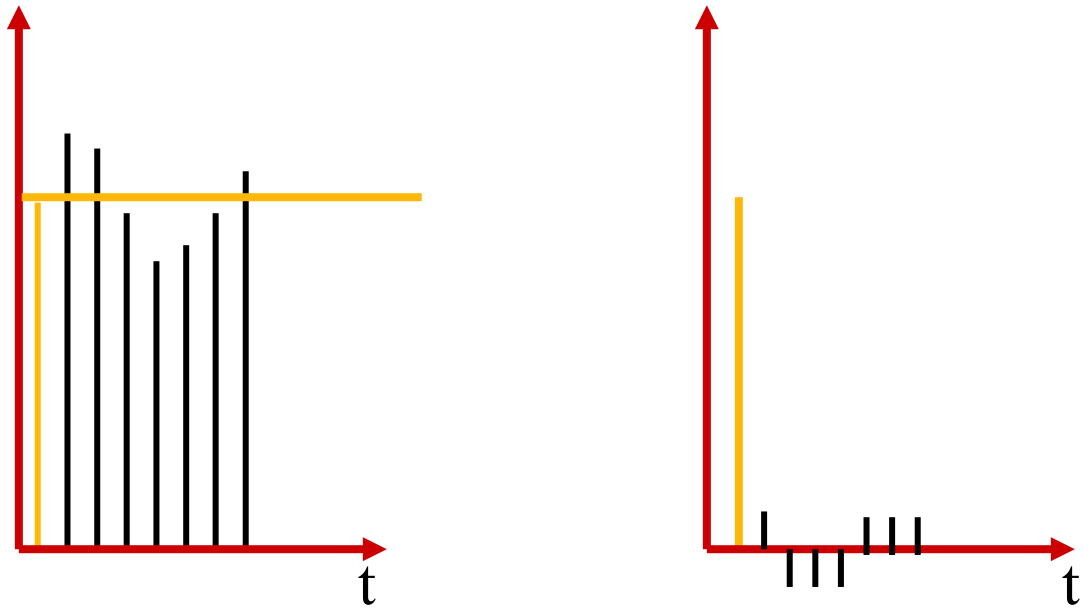
- DPCM
  - Calcula el valor para el siguiente muestreo, almacena la diferencia entre el valor calculado y el real
  - Utilizado en sistemas de telefonía digital, audio de ordenadores y algunos formatos de CD's
- ADPCM
  - Varía dinámicamente el tamaño utilizado para almacenar las diferencias
- El valor calculado =

Última muestra+diferencia



Compresión con pérdida: Predicción

- Delta Modulation (DM)
  - Las diferencias de codifican con 1 bit

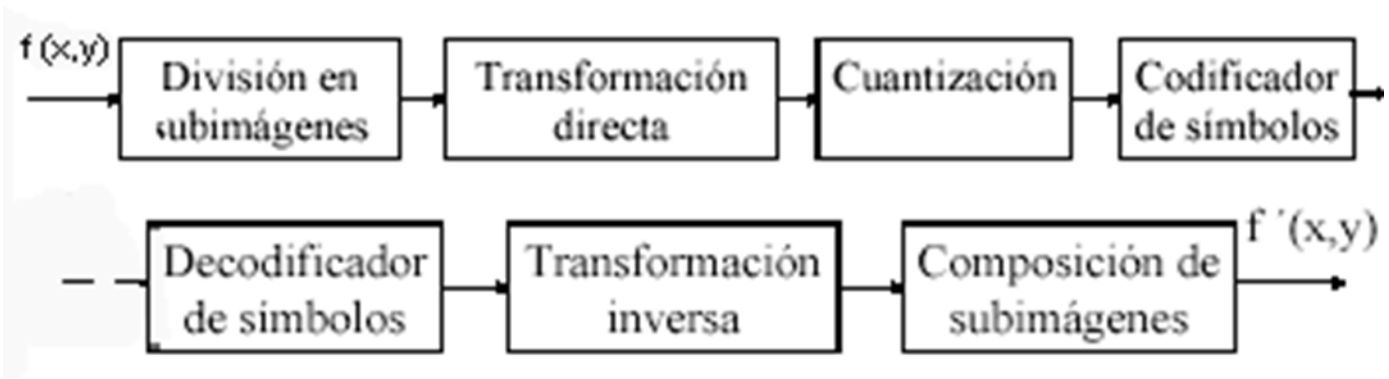


Compresión con pérdida: Transformación

- Codificación por Transformación:
  - FFT – Fast Fourier Transform
  - DCT – Discrete Cosine Transform
- Se realiza una transformación antes de la codificación de la Entropía y se aplica una transformación inversa después de decodificar.
- Tiene la ventaja de que los coeficientes resultantes tienen una distribución estocástica significativa y pueden ser modelados y comprimidos más fácilmente.
- Es decir después de la transformación algunos coeficientes son más predecibles, otros menos. Ello implica que algunos coeficientes pueden ser cuantificados (quantifier) (con pérdida) o/y codificados con entropía (sin pérdida).

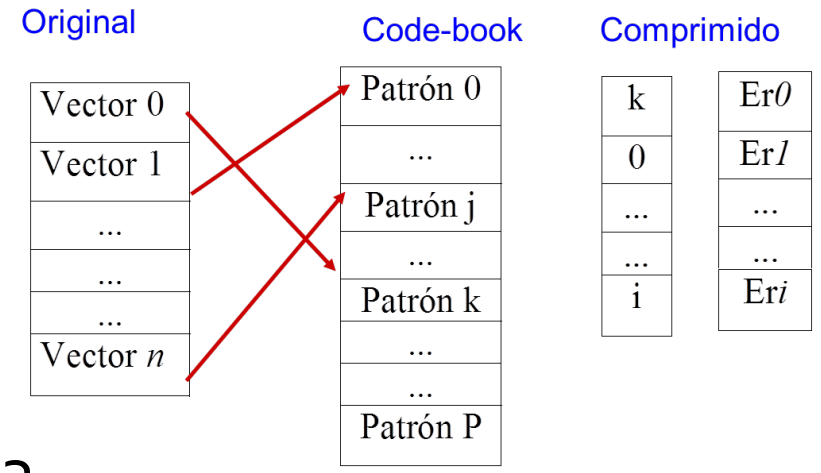
Compresión con pérdida: Transformación

- Metodología: "mapear" la imagen mediante una transformación lineal y reversible, de manera que los valores transformados aporten una información progresiva de la imagen, facilitando con ello su cuantificación



Compresión con pérdida: Cuantificación

- Cuantificación vectorial:
  - El flujo de datos se divide en bloques llamados vectores
  - Utiliza una tabla, denominada code-book
    - Contiene un conjunto de patrones
    - Pueden estar predefinidos o construidos dinámicamente
  - Buscar el patrón más similar al bloque en la tabla
  - Enviar la entrada en la tabla en vez del bloque
  - Se puede incluir un valor del error



Técnicas de compresión

- Métodos híbridos:
  - Imágenes: JPEG
  - Video/Audio: MJPEG, MPEG (1, 2, 4), otros, H.26x

Técnicas de compresión

- Resumen:
  - Entropía: Utiliza la redundancia, sin pérdida
    - RLE:
    - Huffman
    - Aritmética
    - Basada en diccionarios: GIF, TIFF
  - Fuente: Utiliza el contexto semántico, generalmente con pérdida
    - Predictiva: DPCM, ADPCM, DM, MC
    - Transformación: FFT, DCT
    - Por niveles (progresiva)
    - Cuantificación vectorial
  - Híbrido: Combina ambas técnicas