



TV: TeleVisión – Plan 2010

MPEG-1

Contenido (1)

1. Introducción
 1. Estructuras de datos
 2. Modos de predicción
2. Tipos de imágenes
3. Definición de GOP
4. Cuantificación
5. Compensación de movimiento
6. *Video buffering verifier*
7. Control de la tasa binaria
8. Decodificación
9. Edición

Contenido (2)

10. Estructura del flujo binario

1. Nivel de secuencia
2. Nivel de GOP
3. Nivel de imagen
4. Nivel de tira
5. Nivel de macrobloque
6. Nivel de bloque

11. Sintaxis: nivel de sistema y nivel de compresión

1. Introducción

- **MPEG-1:**

- Se diseñó con el objetivo de proporcionar calidad de vídeo similar a VHS y calidad de audio similar a la de un CD (a 1,5 Mbit/s) en los dispositivos domésticos de almacenamiento digital.
- Consta de las siguientes partes:
 - Parte 1 (Sistema) → Multiplexación, sincronización y transmisión de vídeo y audio.
 - Parte 2 (Vídeo) → Compresión de señales de vídeo.
 - Parte 3 (Audio) → Compresión de señales de audio (Layer 3 → MP3).
 - Parte 4 (Pruebas de conformidad).
 - Parte 5 (Modelo de referencia).
- Permite resoluciones de hasta 4095×4095 píxeles a 60 cuadros por segundo.
- Pero lo más habitual es utilizar un subconjunto de parámetros para los que los algoritmos están optimizados. Este formato se denomina SIF (*Source Input Format*):
 - Imágenes de 396 macrobloques y una tasa binaria máxima de 1856 Mb/s.
 - Imágenes progresivas y estructura de muestreo 4:2:0.
 - SIF-525 → 29,97 cuadros por segundo de 240×352 píxeles en EEUU.
 - SIF-625 → 25 cuadros por segundo de 288×352 píxeles (=CIF) en Europa.

1. Introducción

- **Características principales:**

- Acceso aleatorio → Cualquier imagen debe poder ser decodificada en un tiempo limitado.
- Búsqueda rápida → Existen puntos de acceso que permiten el avance rápido.
- Reproducción hacia atrás → Algunas aplicaciones pueden requerirla.
- Robustez frente a errores → Debe permitir al decodificador recuperarse de errores de transmisión.
- Edición → Incluye algunas características para editar el vídeo codificado.

1.1. Estructuras de datos

- **Secuencia:**

- Conjunto de cuadros consecutivos en el tiempo.
- Se identifica con:
 - Cabecera $\rightarrow$ 0x000001B3
 - Dimensión horizontal y vertical de los cuadros $\rightarrow$ 12 bits por dimensión.
 - Relación de aspecto $\rightarrow$ 4 bits.
 - N° de cuadros por segundo $\rightarrow$ 4 bits.
 - Velocidad del canal $\rightarrow$ 18 bits.
 - Tamaño del *buffer* $\rightarrow$ 10 bits.
 - Tablas empleadas en la cuantificación.

- **GOP (*Group Of Pictures* = Grupo de imágenes):**

- Subgrupo de imágenes consecutivas dentro de la secuencia.
- Tiene como función principal el permitir el acceso aleatorio a distintas partes de la secuencia.
- Se identifica con un código de cabecera y el código de tiempos correspondiente al primer cuadro del GOP.

1.1. Estructuras de datos

- **Cuadro:**

- Imagen compuesta por una componente de luminancia y dos de crominancia.
- Constituye la unidad primaria de codificación.
- Se identifica con:
 - Cabecera $\rightarrow$ 0x00000100
 - Información particular del cuadro: referencia temporal, dimensiones máximas de los vectores de movimiento, etc.
 - Modo de codificación.

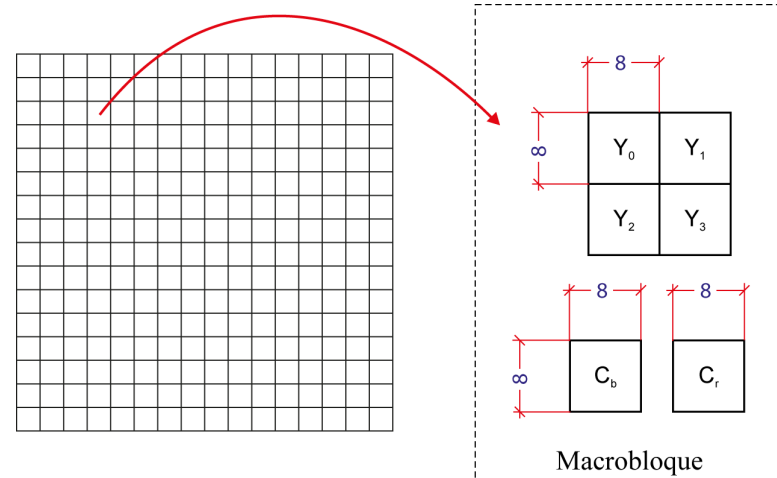
- **Tira ($\cong$ GOB de H.261):**

- Conjunto de MB (MacroBloques) consecutivos en una misma imagen.
- La longitud de las tiras es variable (desde un único MB a toda la imagen).
- Su función principal es recuperar errores y permitir la resincronización del decodificador cuando hay pérdidas $\rightarrow$ Constituye la unidad de resincronización.
- Se identifica con:
 - Cabecera $\rightarrow$ 0x000001...
 - ... 01-AF: posición vertical del primer MB de la tira en la imagen (entre 1 y 15 para SIF-525).
 - Parámetros de codificación que afectan a los MB de la tira.

1.1. Estructuras de datos

• MB (MacroBloque):

- Conjunto de 4 bloques de luminancia y 2 de crominancia, que ocupan la misma posición espacial (igual que en H.261).
- A este nivel se decide si se aplica o no la compensación de movimiento.
- Se define con:
 - El tipo de predicción empleada.
 - El vector de movimiento asociado.
 - Cuantificación aplicada sobre los bloques.
 - Etc.

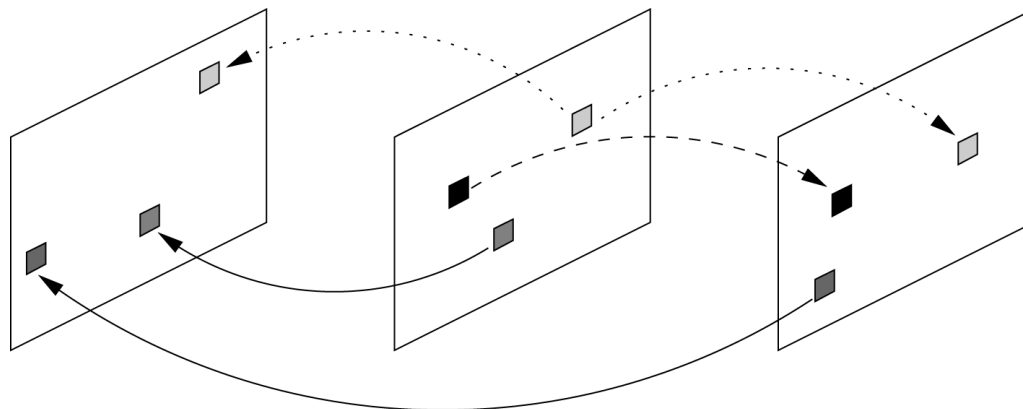


• Bloque:

- Grupo de 8×8 muestras de luminancia o crominancia.
- Constituye la unidad de codificación DCT.
- La codificación DCT se realiza de forma similar a la estudiada en H.261.

1.2. Modos de predicción

- Se puede distinguir entre 2 modos de predicción que se definen a nivel de MB.
- **Intracuarto:**
 - Cuando un MB se codifica sin utilizar predicciones basadas en otros cuadros.
- **Intercuarto:**
 - Cuando un MB se codifica utilizando una predicción obtenida a partir de otro cuadro.
 - Hacia delante → Se usan predicciones obtenidas de cuadros anteriores.
 - Hacia atrás → Se usan predicciones obtenidas de cuadros posteriores.
 - Por interpolación → Se usan predicciones obtenidas de un cuadro anterior y un cuadro posterior.

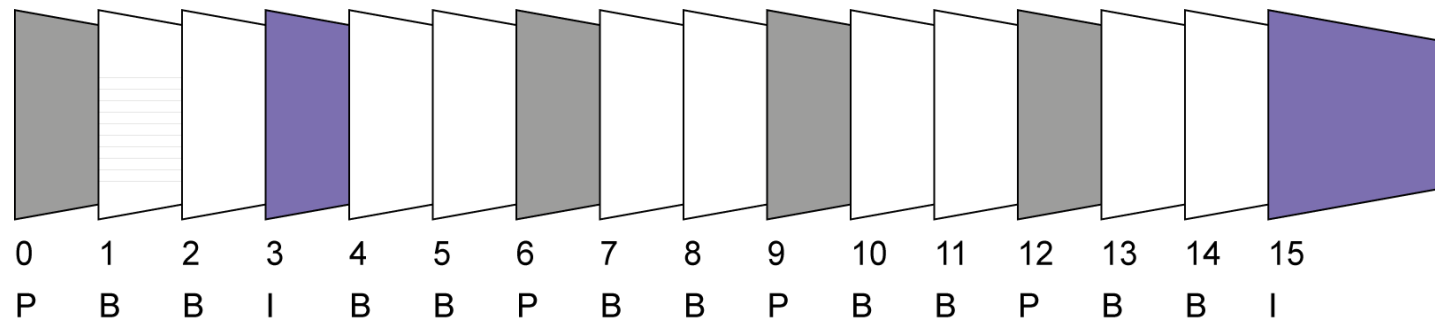


- > predicción hacia atrás (backward)
- ←--- predicción hacia delante (forward)
- ←...→ predicción por interpolación

2. Tipos de imágenes

• Además de ser capaz de proporcionar una codificación eficiente, MPEG-1 debe permitir el acceso aleatorio dentro de la secuencia de vídeo, el avance rápido y el retroceso. Para lograrlo, las secuencias están formadas por distintos tipos de imágenes.

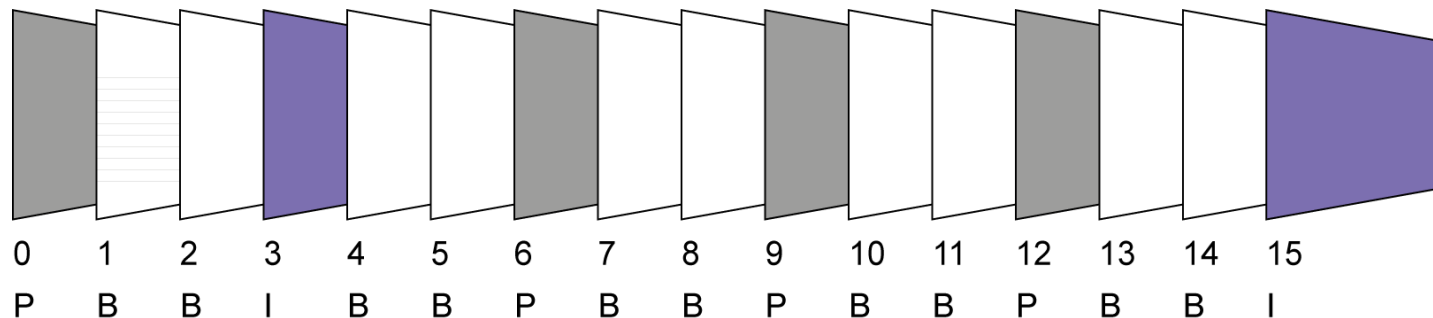
- Imágenes tipo I.
- Imágenes tipo P.
- Imágenes tipo B.
- Imágenes tipo D.



2. Tipos de imágenes

• Imágenes tipo I:

- No dependen de otras imágenes → Todos sus MBs se codifican en modo intra.
- Dependiendo del grado de ocupación del *buffer*, el codificador decide qué nivel de cuantificación aplicar sobre los coeficientes transformados de cada MB (el mismo nivel para los 6 bloques).
- MB intra-d:
 - Utilizan el cuantificador en curso (no cambian el nivel de cuantificación).
- MB intra-q:
 - Definen un nuevo nivel de cuantificación.
 - La cabecera incluye un campo de 5 bits para indicar el nuevo índice de cuantificación (entre 1 y 31).
 - Pueden aparecer en cualquier posición de la imagen (en H.261 el nivel de cuantificación sólo podía modificarse al comienzo de un GOB).



2. Tipos de imágenes

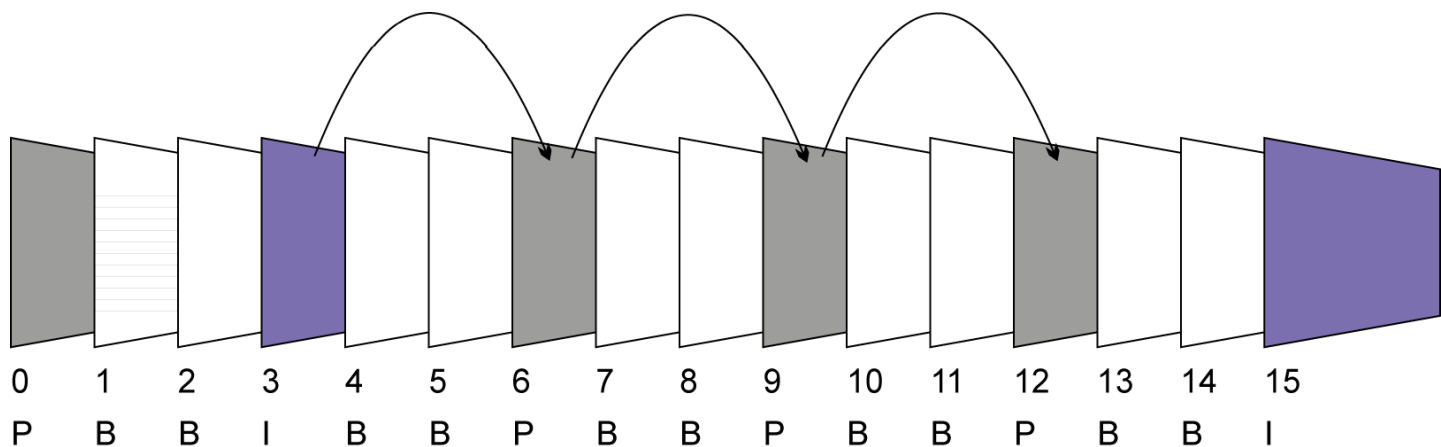
- **Imágenes tipo I (ventajas y desventajas):**

- ✓ Facilitan en comienzo de la decodificación → Ya que no requieren la previa decodificación de ninguna otra imagen.
- ✓ Eliminan la propagación de errores de transmisión → Estos errores sólo se propagan con las imágenes que utilizan predicción.
- ✓ Permiten la existencia de GOPs cerrados → En edición es necesario extraer segmentos de vídeo que no requieran decodificar otras partes de la secuencia.
- ✓ Permiten la reproducción hacia atrás → Seleccionando únicamente las imágenes de tipo I es posible visualizar la secuencia en orden inverso.
- ✗ Al no utilizar predicciones temporales, su codificación es poco eficiente.
- ✗ Al ocupar más bits que otros tipos de imágenes, producen retardos en el *buffer*.

2. Tipos de imágenes

• Imágenes tipo P:

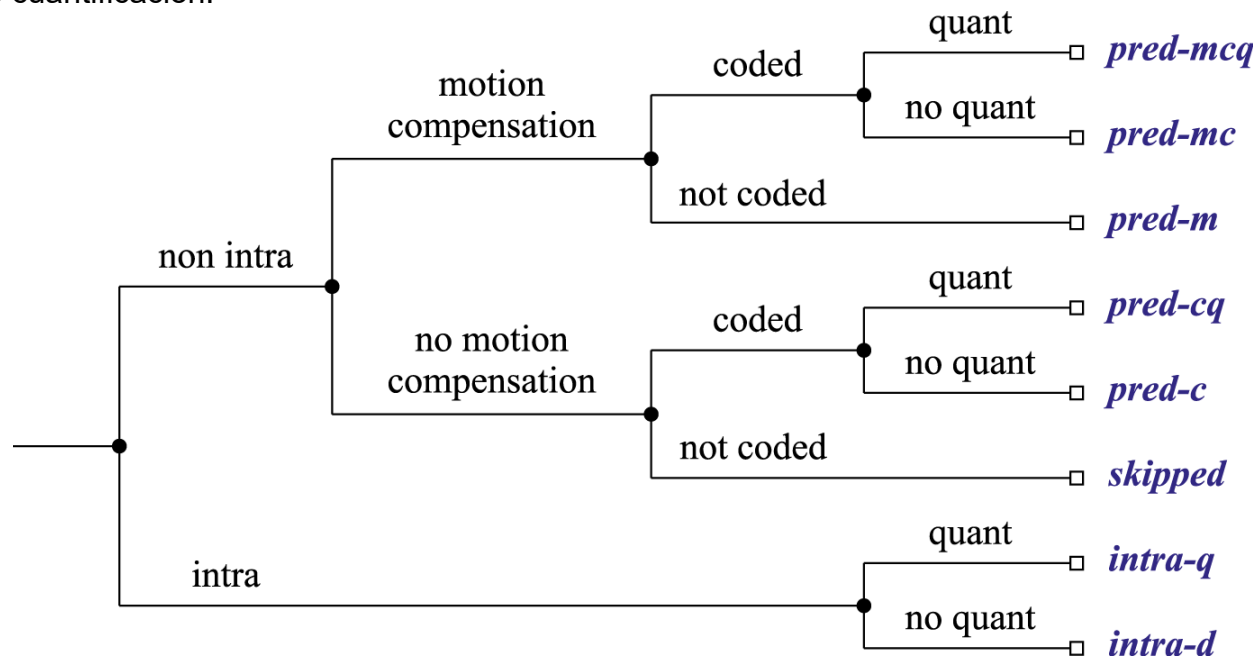
- A nivel de MB, el codificador decide si utilizar predicción intra-cuadro (no MC) o inter-cuadro (MC).
- En el caso de predicción tipo ínter → Se toma como referencia la primera imagen previa que sea de tipo I o de tipo P.
- Las decisiones de utilizar predicción ínter vs. intra y MC vs. no MC se realizan del mismo modo que en H.261.
- La codificación de este tipo de imágenes es idéntica a la de H.261, salvo por diferencias menores en el proceso de cuantificación.
- Para codificar una imagen de este tipo hace falta tener almacenada en la memoria del codificador una imagen previa de tipo I o tipo P.



2. Tipos de imágenes

• Imágenes tipo P (tipos de MB):

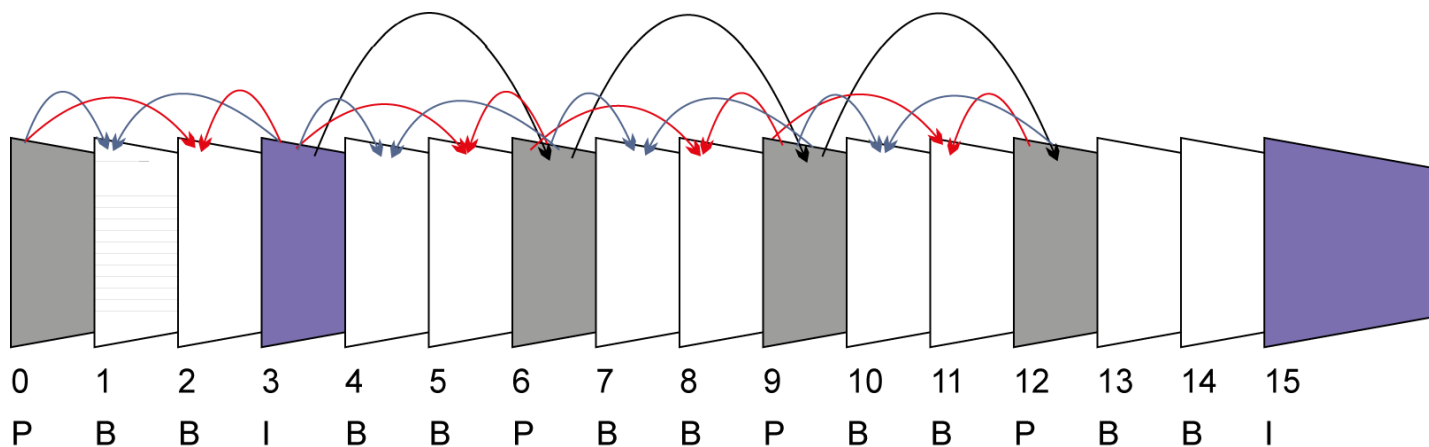
- En el caso de utilizarse una codificación tipo intra, el MB es tratado igual que los de las imágenes tipo I → Se decide si modificar o no el nivel de cuantificación.
- En el case de codificación ínter (independientemente de si se usa o no MC), se toman las siguientes decisiones:
 - 1) Si todos los coeficientes de un bloque son despreciables → No se codifican.
 - 2) Si hay coeficientes significativos → Se codificarán y será necesario decidir si modificar o no el nivel de cuantificación.



2. Tipos de imágenes

• Imágenes tipo B:

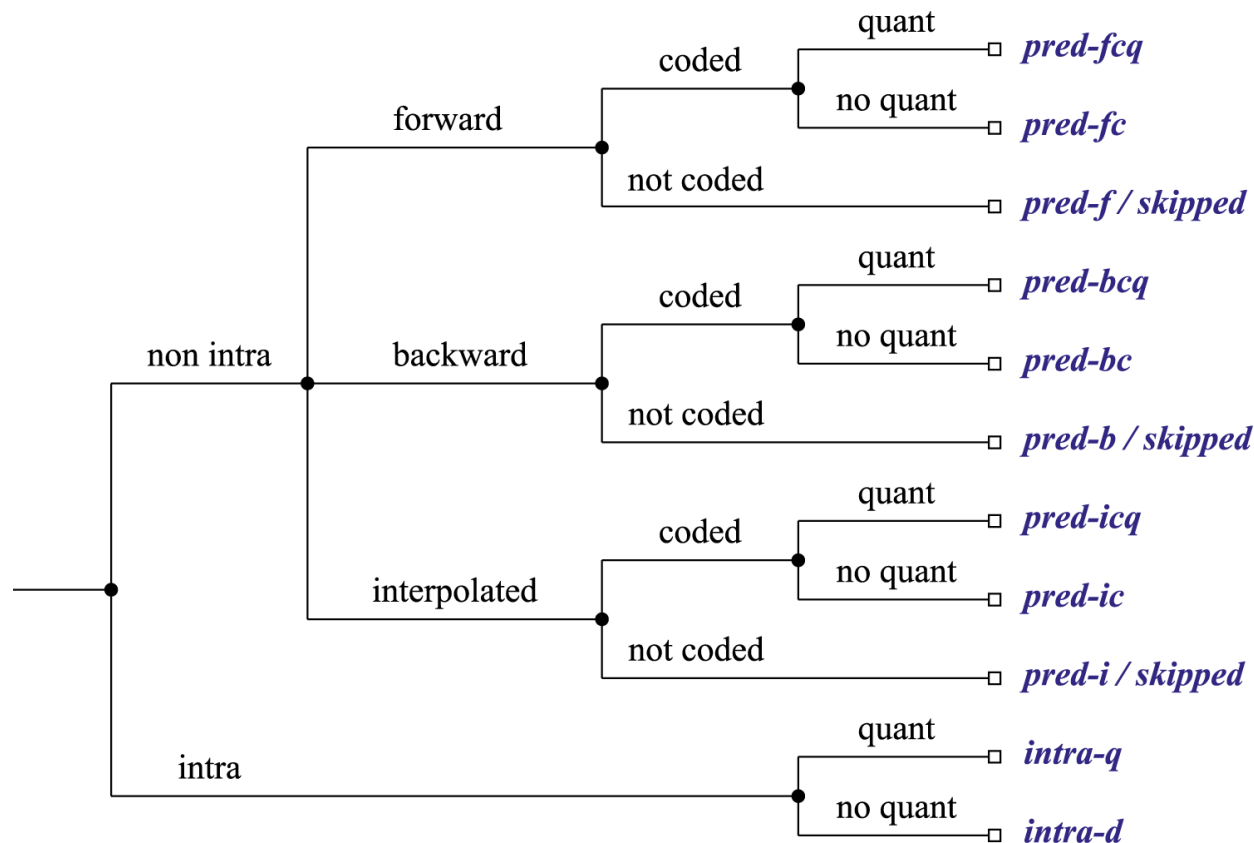
- Utilizan como predicción una imagen previa, una futura o una combinación de ambas.
- Las imágenes de referencia pueden ser de tipo I o tipo P.
- De esta forma la codificación es mucho más eficiente.
- No pueden ser utilizadas como predicción para otras imágenes → Esto permite reducir su calidad, ya que no propagan errores.
- Tanto para codificarlas como para decodificarlas es necesario disponer de 2 imágenes de referencia previamente codificadas y decodificadas → **El orden de codificación es distinto al de reproducción.**



2. Tipos de imágenes

• Imágenes tipo B (tipos de MB):

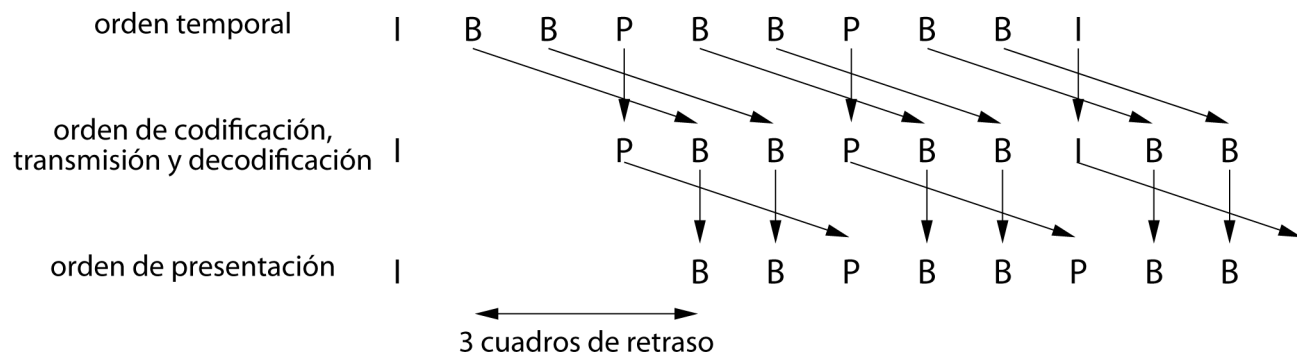
- Para cada MB se analizan los errores utilizando los 3 tipos de predicción y se selecciona el menor (también se considera la posibilidad de codificar en modo intra).
- En este tipo de imágenes, siempre se usa MC.



2. Tipos de imágenes

• Imágenes tipo B (ventajas e inconvenientes):

- ✓ La predicción con interpolación da muy buenos resultados → Reduce notablemente el error de predicción.
- ✓ Facilitan la reproducción a velocidad rápida, ya que este tipo de reproducción se basa únicamente en los cuadros I y P.
- ✗ Elevado coste computacional → Requieren de numerosas comparaciones.
- ✗ Requieren mayor memoria en el codificador → Utilizan dos imágenes de referencia.
- ✗ Reducen la eficiencia de la predicción en las imágenes tipo P → Cuantas más imágenes B seguidas, más distancia temporal entre una imagen P y su imagen de referencia. Además, los vectores de movimiento serán más grandes.
- ✗ Introducen un retardo adicional en la presentación de la secuencia decodificada (este retardo depende del número de imágenes B consecutivas).



2. Tipos de imágenes

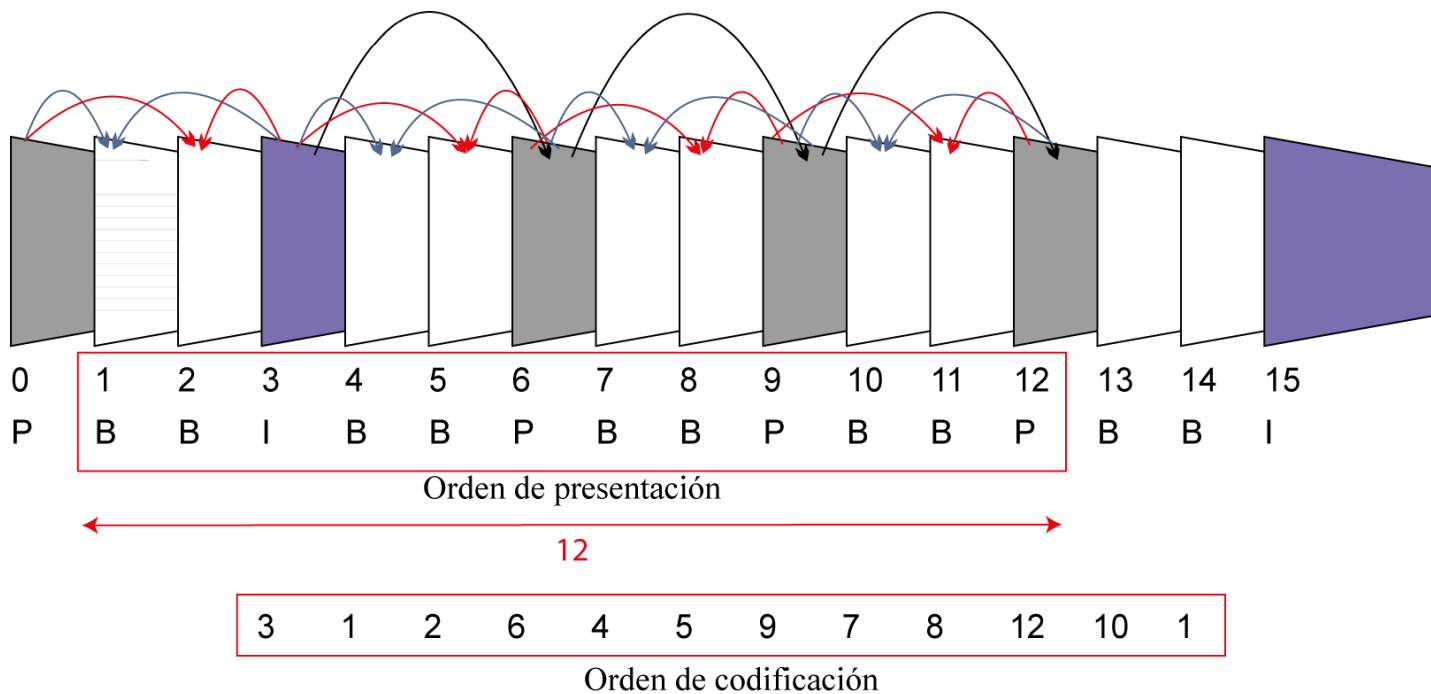
- **Imágenes tipo D:**

- Se codifican en el cuadro → Tipo intra.
- Sólo se codifica el coeficiente de continua (DC).
 - La calidad es muy pobre.
 - Pero su codificación y decodificación es muy rápida.
- Se utilizan para aplicaciones específicas como el avance rápido.
- No se mezclan con imágenes tipo I, P o B.
- Se usan en raras ocasiones.

3. El GOP

Definición:

- Un **GOP (Group Of Pictures)** es un conjunto de imágenes consecutivas (orden de visualización), que soportan tanto el acceso aleatorio como la edición de vídeo.



3. El GOP

- **Propiedades:**

- Debe contener, al menos, una imagen tipo I.
 - Esta imagen podrá ir seguida de otras imágenes tipo I o P.
 - También podrá haber imágenes tipo B:
 - Entre pares de imágenes I/I, I/P o P/P.
 - Precediendo a la primera imagen tipo I.
 - Longitud mínima → Una imagen.
 - Longitud máxima → No hay límite.
- Su codificación siempre comienza por una imagen tipo I, e irá seguida del resto de imágenes del GOP.

3. El GOP

- **Orden de presentación:**

- Comienza por una imagen I o B.
- Termina con una imagen I o P.

- **Ejemplos:**

I

I P P

I B P B P

B B I B P B P

B B I B B P B B P B B P

B I B B B B P B I B B I I

3. El GOP

- **Características habituales (las que vamos a considerar de ahora en adelante):**

- Una sola imagen tipo I.
- Una única definición de GOP para toda la secuencia.
- El número de imágenes tipo B consecutivas siempre será el mismo.
- El primer GOP no incluirá imágenes tipo B antes de la imagen tipo I.

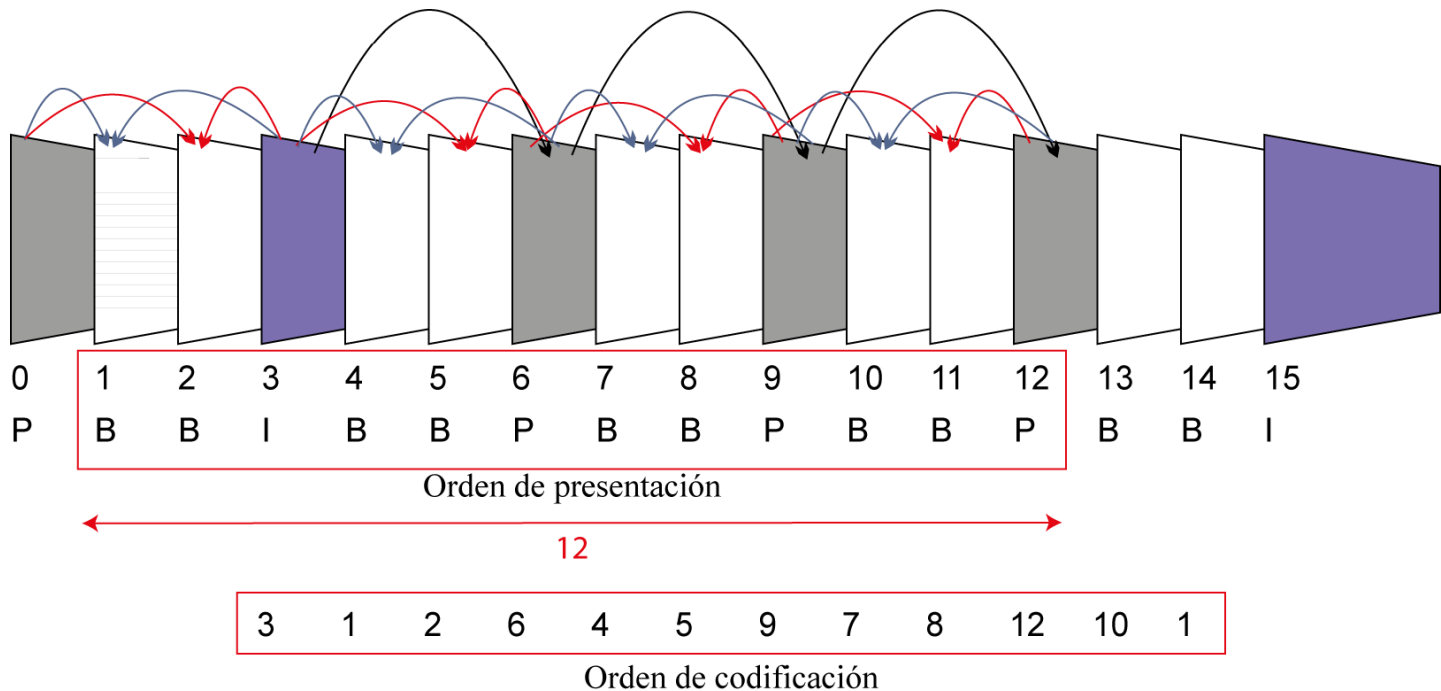
❖ El GOP se definirá mediante dos parámetros:

- N – longitud del GOP
- M – distancia entre imágenes I/P (número de imágenes B consecutivas + 1)



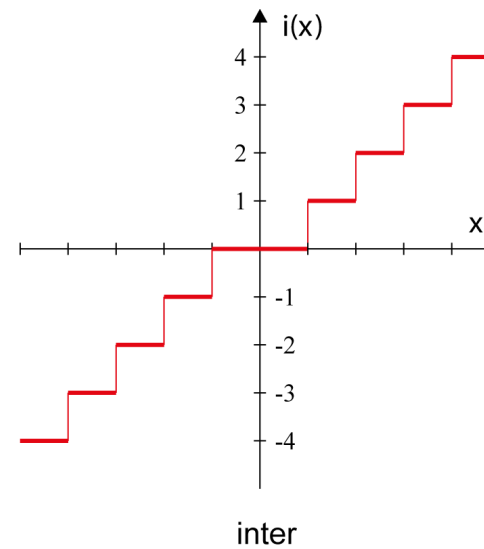
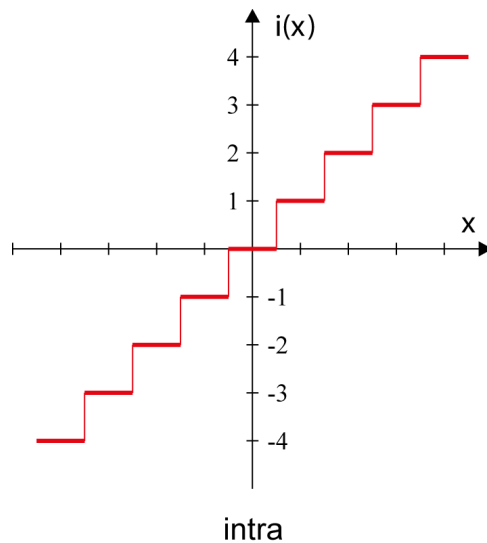
3. El GOP

- **GOP cerrado:** El que puede decodificarse sin utilizar imágenes del GOP anterior.
 - Tiene que comenzar por una imagen tipo I o por imágenes tipo B que solo utilicen codificación hacia atrás (*backward*).
- **GOP abierto:** Su decodificación requiere imágenes del GOP previo.
 - Por ejemplo, el representado en la figura.



4. Cuantificación

- Se utilizan dos cuantificadores distintos, uno para el modo intra y otro para el modo ínter.
- Ambos cuantificadores son uniformes.
- El del modo ínter incluye una zona muerta (*dead-zone quantizer*).



4. Cuantificación

- El paso de cuantificación se obtiene a partir del producto de:
 - La escala de cuantificación, S_q :
 - Puede tomar cualquier valor del conjunto $\{2, 4, \dots, 62\}$ y es la misma para todo el MB.
 - Su valor se decide en función de la actividad espacial (varianza) del MB y de la ocupación del *buffer*.
 - Un peso $w(u, v)$:
 - Se asigna un valor distinto a cada coeficiente dependiendo de la frecuencia espacial y horizontal que representa. Mayor peso para los coeficientes de mayor frecuencia.
 - El paso del DC en el modo intra siempre es 8 → Mayor precisión debido a que el ojo es muy sensible a las bajas frecuencias.
 - Los coeficientes cuantificados son valores enteros limitados al intervalo $[-255, 255]$.

4. Cuantificación

• Pesos de los coeficientes:

- MB tipo intra → Se asignan distintos pesos a los distintos coeficientes.
 - Los pesos se han determinado en función de la sensibilidad del ojo a las distintas frecuencias espaciales, considerándose una distancia de visión de entre 4 y 6 veces el alto de la imagen.
- MB tipo ínter → Se usa el mismo peso para todos los coeficientes:
 - En este caso, las altas frecuencias no tienen por qué tener su origen en las altas frecuencias espaciales.

Intra							
8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Inter							
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16

4. Cuantificación

- **Cuantificación - Modo intra:**

1. Se calcula, para cada coeficiente transformado, el valor:

Coeficiente DC

$$i_0(1,1) = \frac{C(1,1)}{8}$$

Coeficientes AC

$$i_0(u,v) = 8 \frac{C(u,v)}{w(u,v)S_q}$$

2. El índice del intervalo de cuantificación, $i(u,v)$, se obtiene redondeando estos valores al entero más próximo.
 - Si el valor a redondear es de la forma $n + 0,5$ se redondea al entero más próximo en la dirección a 0 (para favorecer la aparición de índices más pequeños).

4. Cuantificación

- **Reconstrucción - Modo intra:**

1. Se calcula, para cada valor cuantificado, el valor:

Coeficiente DC

$$r_0(1,1) = i(1,1) \cdot 8$$

Coeficientes AC

$$r_0(u,v) = \left\lfloor \frac{i(u,v)w(u,v)S_q}{8} \right\rfloor$$

2. El resultado es un entero que, si es diferente a 0, se redondea al impar más próximo en dirección a 0 → Obteniéndose así el valor reconstruido $r(u,v)$.

4. Cuantificación

• Cuantificación - Modo inter:

1. Se calcula, para cada coeficiente transformado, el valor:

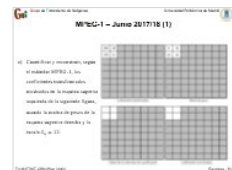
$$i_0(u, v) = \frac{8 \cdot C(u, v)}{w(u, v) S_q}$$

2. El índice del intervalo de cuantificación, $i(u, v)$, se obtiene **truncando hacia cero la parte decimal**.

• Reconstrucción - Modo inter:

1. Se calcula, para cada valor cuantificado, el valor:

$$r_0(u, v) = \frac{(2 \cdot i(u, v) + \text{sign}(i(u, v))) \cdot w(u, v) \cdot S_q}{16}$$



2. El valor reconstruido, $r(u, v)$, se obtiene redondeando $r_0(u, v)$ al entero más próximo y, si el número obtenido es distinto de 0, redondeándolo al impar más próximo en dirección a 0.

5. Compensación de movimiento (1)

- En H.261 era opcional, ya que se pensaba que las muestras del error de predicción estarían muy poco correlacionadas y que, por lo tanto, apenas aportaría ganancia.
- Sin embargo, más tarde se demostró que sí que existe esa correlación espacial → Los codificadores posteriores (como MPEG-1) utilizan MC.
- Principales diferencias de la estimación y compensación de MPEG-1 frente a H.261:
 - Es una característica integral del códec.
 - El área de búsqueda es mucho mayor.
 - Se incrementa la precisión de los vectores de movimiento.
 - En las imágenes tipo B se utiliza compensación bidireccional.

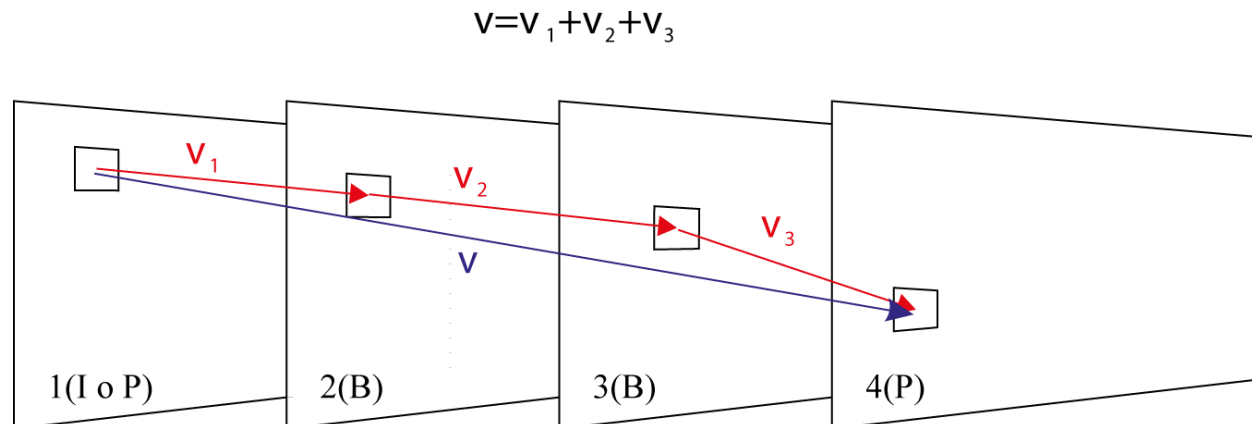
5. Compensación de movimiento (2)

- **Área de búsqueda:**

- MPEG-1 está pensado para películas, donde el movimiento de los objetos es mucho más violento que en secuencias de vídeo-`{confere,vigila}`ncia (H.261).
- Además, en MPEG-1 las imágenes de referencia pueden estar a más de una imagen de distancia.

¡Es necesario incrementar el área de búsqueda!

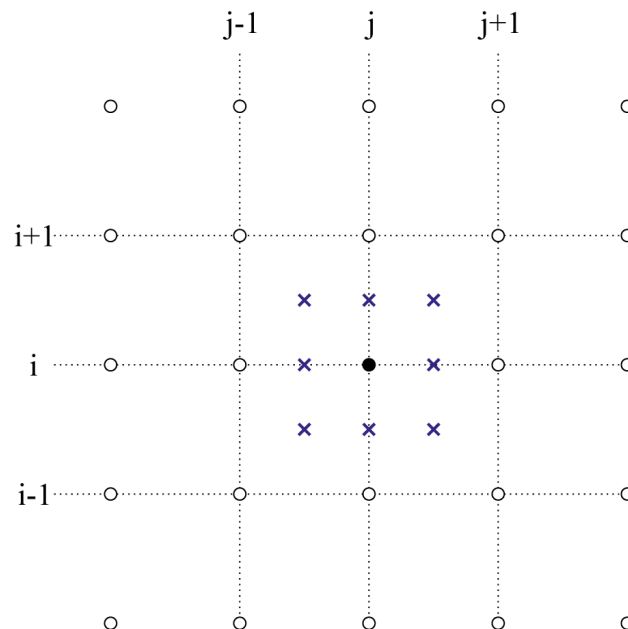
- Problemas: La búsqueda exhaustiva es muy costosa y los métodos de búsqueda rápida (p.ej., logarítmica) no funcionan bien en áreas de búsqueda grandes.
- Solución: Búsqueda telescópica.
 - En vez de usar sólo la imagen de referencia, los vectores de movimiento se obtienen haciendo búsquedas en cadena a lo largo de las imágenes tipo B.
 - De esta forma se pueden utilizar áreas de búsqueda más pequeñas en las que los algoritmos rápidos funcionan mejor.



5. Compensación de movimiento (3)

- **Precisión de los vectores de movimiento:**

- La precisión de los vectores puede incrementarse a medio píxel.
- El tipo de precisión se indica con unos *flags* en la cabecera de cada imagen.
- Si se utiliza precisión de medio píxel, los valores de la imagen de referencia se interpolan (interpolación bilineal) y se redondean al entero más próximo.
- Para reducir el coste computacional primero se hace una búsqueda a nivel de píxel y, en un segundo paso, únicamente se evalúan los 8 puntos más cercanos con precisión de subpíxel.

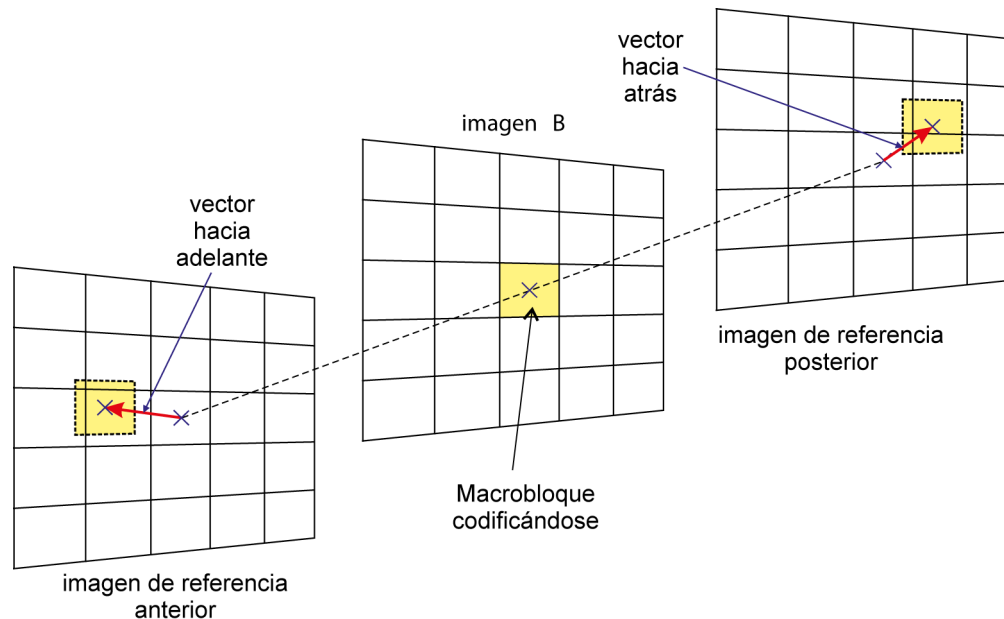


- Se necesitan 2 bits más por vector (uno por coordenada) → Mayor calidad, pero menor compresión.
- Al igual que en H.261, los vectores se codifican de forma diferencial.

5. Compensación de movimiento (4)

• Estimación bidireccional:

- Las imágenes de tipo B utilizan imágenes anteriores y posteriores → La estimación de movimiento puede referirse a:
 - Una imagen de referencia anterior (hacia delante, *forward*).
 - Una imagen de referencia futura (hacia atrás, *backward*).



○ Esta posibilidad incrementa notablemente la calidad de la compensación de movimiento, especialmente cuando aparecen en la escena objetos que son ocluidos por otros.

○ La sobrecarga asociada a los vectores de movimiento de una imagen tipo B es muy superior a la de una imagen tipo P:

- Pueden dar lugar al envío de más vectores (MB en los que se use tanto una imagen previa como una futura).
- Tienen asociados más tipos de MB.

5. Compensación de movimiento (5)

- Codificación de los vectores de movimiento:**

- Su rango se especifica mediante el parámetro **f_code**. Este parámetro toma valores entre 1 y 7 y, además, el rango asociado a cada valor depende de si la estimación se hace a nivel de píxel o de medio píxel.

f_code	f	Rango de los vectores		Modulo
		full_pel =0	full_pel =1	
1	1	− 8 a 7.5	− 16 a 15	32
2	2	− 16 a 15.5	− 32 a 31	64
3	4	− 32 a 31.5	− 64 a 63	128
4	8	− 64 a 63.5	− 128 a 127	256
5	16	− 128 a 127.5	− 256 a 255	512
6	32	− 256 a 255.5	− 512 a 511	1024
7	64	− 512 a 511.5	− 1024 a 1023	2048

- Cuanto mayor es f_code, más bits se necesitan para codificar los vectores. Para evitar tener que usar una tabla demasiado grande cada diferencia a codificar (MD) se representa con 2 enteros (mc, mr).

$$f = 2^{f_{code}-1}$$

- Parte principal (motion code, mc) → Resultado de redondear la siguiente expresión (valores entre -16 y 16 codificados con VLC):

$$mcf = \frac{MD + \text{sgn}(MD) \cdot (f - 1)}{f}$$

- Residuo (mr) → Es la diferencia entre |MD| y |mc| expresada como complemento a 1 (valores entre 0 y f-1 codificados con f_code − 1 bits).

$$mr = (f - 1) - (|mc \cdot f| - |MD|)$$

6. *Video buffering verifier* (1)

- En MPEG-1 la cantidad de datos codificados es muy variable:
 - Las imágenes tipo I pueden llegar a ocupar 2 veces más que las imágenes tipo P y 4 veces más que las imágenes tipo B.
 - La mayor parte de las aplicaciones requieren que la tasa binaria que llegue al receptor sea constante (R bit/s).
- Además, el decodificador necesita extraer los datos de las imágenes de forma constante (P imágenes/s).

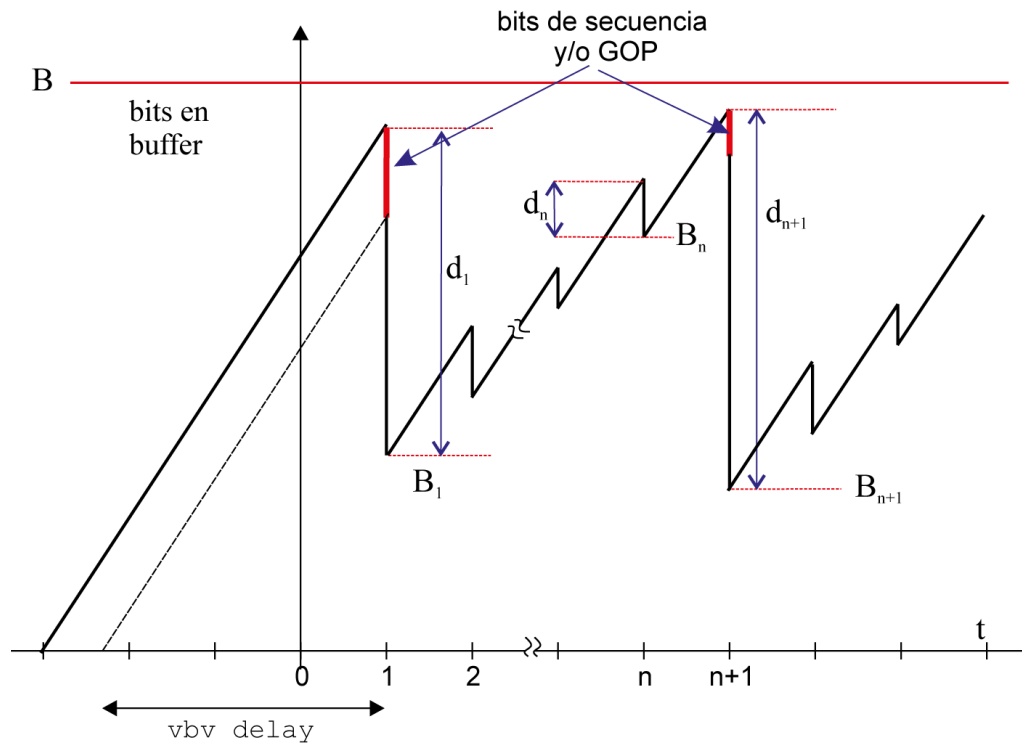
¡Es necesario utilizar un *buffer* que gestione la velocidad de transmisión de los datos!

- VBV (*Video Buffering Verifier*):
 - Especifica la forma en la que debe funcionar el codificador para no provocar sobrecargas en el *buffer*.
 - Funciona con la misma frecuencia de reloj y el mismo número de imágenes por segundo que el codificador.
 - Especifica el tamaño del *buffer*, B , al comienzo de la secuencia.
 - Inicialmente está vacío y se llena durante un tiempo especificado antes de empezar la decodificación.
 - Cuando el decodificador requiere una nueva imagen, se sacan del *buffer* a la vez todos los datos asociados a dicha nueva imagen.

6. Video buffering verifier (2)

- Ocupación del buffer:

- El VBV se examina antes y después de cualquier extracción de datos y, en cualquier caso, su ocupación debe estar comprendida entre 0 y B bits.
- El número de bits que entran en el *buffer* durante el tiempo correspondiente a una imagen es $R(\text{bps})/P(\text{fps})$.



7. Control de la tasa binaria (1)

- El codificador debe asegurarse de que el VBV nunca esté vacío ni se desborde:
 - Dado que se retiran de golpe todos los datos asociados a cada imagen, es necesario controlar el número total de bits por imagen.
- En H.261 el grado de ocupación del *buffer* se controla ampliando o reduciendo el tamaño de los intervalos de cuantificación.
- En MPEG-1, dado que existen 3 tipos de imágenes que generan distinta información, el control de la ocupación del *buffer* es más complicado.
 - Se realiza a nivel de secuencia y de imagen.

7. Control de la tasa binaria (2)

- **A nivel de secuencia:**

- La cantidad de bits generados depende del contenido de la secuencia a codificar:
 - Las secuencias con poco movimiento dan lugar a menos bits.
- Si hay poco movimiento las imágenes tipo I deberán recibir una gran proporción de bits, mientras que si hay mucho movimiento dicha proporción será menor.
- La norma sugiere que se comience con una proporción de bits para cada tipo de imagen igual a 6:3:2.
 - Según vaya siendo necesario y dependiendo de la cantidad de movimiento en la secuencia se irá modificando dicha proporción.
 - Se adapta el ancho de los intervalos de cuantificación para lograr el objetivo.

- Ejemplo:

- Secuencia SIF-625 a 1,4 Mb/s; framerate = 25 fps; GOP con N=12 y M=3; relación 6:3:2
 - N° medio de bits por GOP $\rightarrow 672$ kbits/GOP
 - Imágenes tipo I, P y B $\rightarrow 1$ tipo I, 3 tipo P, 8 tipo B.
 - Bits a asignar a las tipo I $\rightarrow 130,06$ kbits
 - Bits a asignar a las tipo P $\rightarrow 65,03$ kbits
 - Bits a asignar a las tipo B $\rightarrow 43,35$ kbits

7. Control de la tasa binaria (3)

- **A nivel de secuencia:**

- En la práctica, la tasa de las imágenes tipo B se reduce 1,4 veces debido a:

- Los coeficientes de la DCT suelen tener una magnitud menor que en el resto de las imágenes → aumentando el ancho de los intervalos de cuantificación la calidad apenas varía.
 - Las imágenes tipo B no se usan para decodificar otras imágenes → Si el error de codificación resulta demasiado grande dicho error no se propagará.

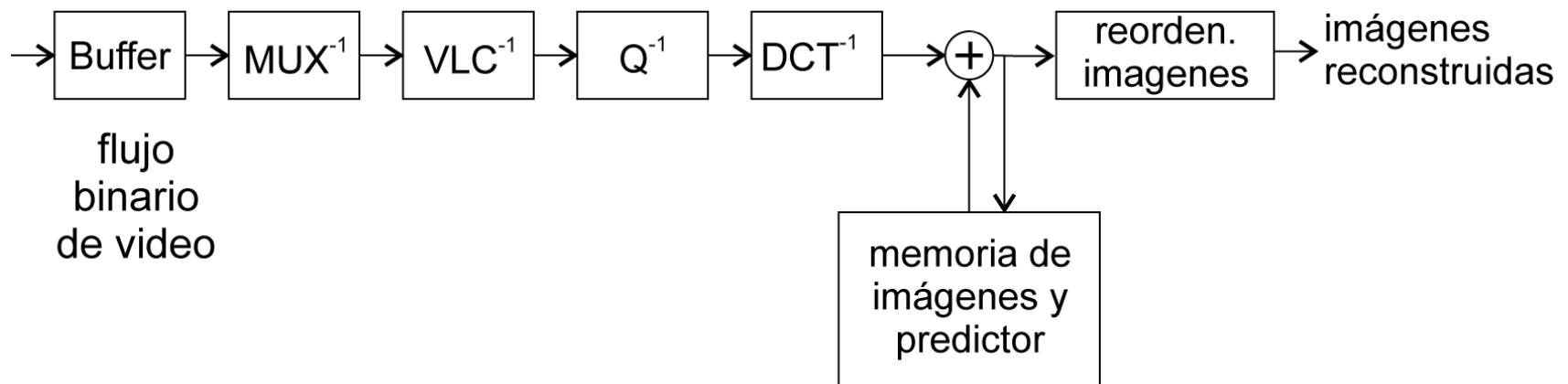
7. Control de la tasa binaria (4)

- A nivel de imagen:

- Para conseguir que la tasa binaria a nivel de imagen se acerque al objetivo calculado se puede modificar el ancho de los intervalos de cuantificación del mismo modo que se hace en H.261.
- Los mejores algoritmos de asignación, además de tener en cuenta la desviación del nº de bits generados frente a la tasa objetivo, tienen en cuenta las características de la secuencia (e.g., cantidad de movimiento).
- Si el *buffer* se llena demasiado, el codificador también puede optar por descartar coeficientes de la DCT (los de mayor frecuencia).
- Si el *buffer* se vacía peligrosamente, el codificador puede optar por insertar bits de relleno.

8. Decodificación (1)

1. El flujo binario se demultiplexa en: coeficientes DCT, vectores de movimiento, parámetros, etc.
2. Los datos demultiplexados se decodifican usando las tablas VLC correspondientes.
3. Los coeficientes DCT se reconstruyen y se les aplica la DCT inversa.
4. Se suma la misma imagen de predicción utilizada en codificación para construir la imagen de error.
5. Las imágenes se reordenan según el orden de presentación adecuado.



- De la cabecera de la secuencia se extraen parámetros necesarios para la decodificación:
 - Resolución, relación de aspecto, matrices de pesos, tasa binaria requerida,...
- A continuación, se decodifica la cabecera del GOP para determinar su estructura y el nivel de llenado del *buffer* requerido (la decodificación se retrasa hasta que el buffer alcanza el nivel mínimo establecido).

8. Decodificación (2)

- **Decodificación para avance rápido:**

- Puede implementarse haciendo uso exclusivo de imágenes tipo D o tipo I.
 - El decodificador selecciona sólo estas imágenes, descartando las tipo B y P.
 - La secuencia se reproduce a una velocidad N veces superior a la normal.
- Requisitos:
 - El medio de reproducción debe ser capaz de leer y entregar imágenes a N veces la velocidad normal.
 - El decodificador debe ser capaz de aceptar dicha velocidad y decodificar las imágenes tipo I (la tasa media es muy superior a la correspondiente al modo de reproducción normal).
- Otra forma de conseguir este tipo de reproducción es que el medio de almacenamiento seleccione únicamente las imágenes tipo I y se las entregue al decodificador en forma de un flujo MPEG válido.

8. Decodificación (3)

- Decodificación para pausa y reproducción imagen a imagen:**

- Se requiere que el decodificador controle el flujo binario entrante y sea capaz de presentar una imagen decodificada sin decodificar más imágenes.
- Si el control sobre el flujo binario no es total (por ejemplo, en los CD-ROM), se producirá un retardo antes de reiniciar la reproducción.

- Decodificación para reproducción hacia atrás:**

- El decodificador debe decodificar cada GOP en dirección hacia delante, almacenar las imágenes y presentarlas en orden inverso.
 - Requiere mucha memoria de almacenamiento.
 - Si el GOP es pequeño se facilita este tipo de reproducción.
- Se puede reducir la cantidad de memoria necesaria si las imágenes se decodifican en un orden distinto en función de si se usa reproducción hacia delante o hacia atrás.

B	B	I	B	B	P	B	B	P	B	B	P	orden de presentación
0	1	2	3	4	5	6	7	8	9	10	11	referencia temporal
I	B	B	P	B	B	P	B	B	P	B	B	orden de decodificación
2	0	1	5	3	4	8	6	7	11	9	10	referencia temporal
I	P	P	P	B	B	B	B	B	B	B	B	orden de rep. hacia atrás
2	5	8	11	10	9	7	6	4	3	1	0	referencia temporal

9. Edición

- Si se decodifica el vídeo, se edita y se vuelve a codificar → Se pierde calidad.
- Es mejor editar directamente el flujo binario.
- Para conseguirlo, se insertan puntos de corte en los que el flujo está preparado para las operaciones de edición:
 - Deben ir seguidos de un GOP cerrado.
 - El nivel de llenado del buffer tras él debe ser igual al de la primera imagen de la secuencia.

10. Estructura del flujo binario (1)

- **Nivel de secuencia:**

- Una secuencia está formada por un conjunto de grupos de imágenes (GOPs), precedidos de una cabecera de secuencia.



- Los valores más importantes codificados en la cabecera son:

- **sequence_header_code** → 32 bits “0x000001B3” señalando el comienzo de la secuencia.
- **horizontal_size** → 12 bits para anchura de la parte visible de la componente Y.
- **vertical_size** → 12 bits para altura de la parte visible de la componente Y.
- **pel_aspect_ratio** → 4 bits para relación de aspecto de los píxeles. Los 4 valores no “reservados” sino realmente usados son: 1:1; 4:3; 16:9; 2,21:1.
- **picture_rate** → 4 bits para velocidad de secuencia [fps]. Los 8 valores usados son: 24/1.001 (~23,98); 24; 25; 30/1.001 (~29,97); 30; 50; 60/1.001 (~59,94); 60.
- **bit_rate** → 18 bits para velocidad binaria. El valor “0x3FFFF” indica velocidad variable.

10. Estructura del flujo binario (2)

- Nivel de secuencia:

- **vbv_buffer_size** → 10 bits para el tamaño mínimo del buffer. Dicho tamaño se calcula como:

$$B = 16 \times 1024 \times vbv_buffer_size$$

- **constrained_parameter_flag** → 1 bit que es igual a 1 si se cumplen las siguientes condiciones:

- $horizontal_size \leq 768$
 - $vertical_size \leq 576$
 - NMB (Número de MB/imagen) ≤ 396 . Este número se calcula como:

$$NMB = \frac{horizontal_size + 15}{16} \times \frac{vertical_size + 15}{16}$$

- Número de MB/s $\leq 396 \times 25$
 - $picture_rate \leq 30$ fps.
 - $forward_f_code \leq 4$.
 - $backward_f_code \leq 4$.

10. Estructura del flujo binario (3)

- **Nivel de secuencia:**

- **load_intra_quantizer_matrix** → 1 bit.

- Si está a 1 indica que a continuación se incluye una matriz de cuantificación intra.
 - Si está a 0 indica que se usará la matriz por defecto.

- **intra_quantizer_matrix** → Lista opcional de 64 enteros de 8 bits que remplazan a la matriz intra por defecto.

- El 0 no está permitido y el coeficiente de continua siempre es 8.

- **load_non_intra_quantizer_matrix** → 1 bit.

- Si está a 1 indica que a continuación se incluye una matriz de cuantificación ínter.
 - Si está a 0 indica que se usará la matriz por defecto.

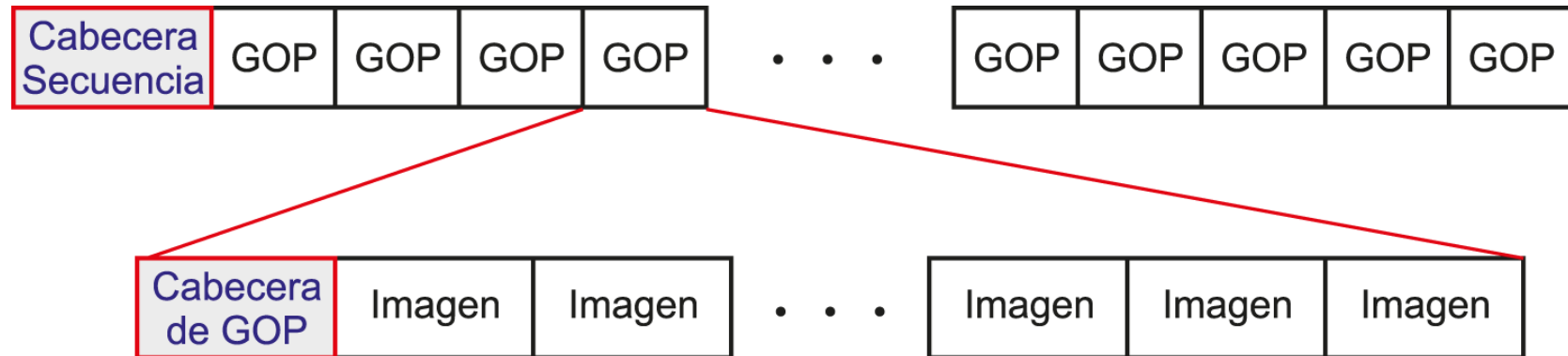
- **non_intra_quantizer_matrix** → Lista opcional de 64 enteros de 8 bits que remplazan a la matriz ínter por defecto.

- El 0 no está permitido.

10. Estructura del flujo binario (4)

- Nivel de GOP:

- Un GOP consiste en un conjunto de imágenes precedido de una cabecera de grupo.



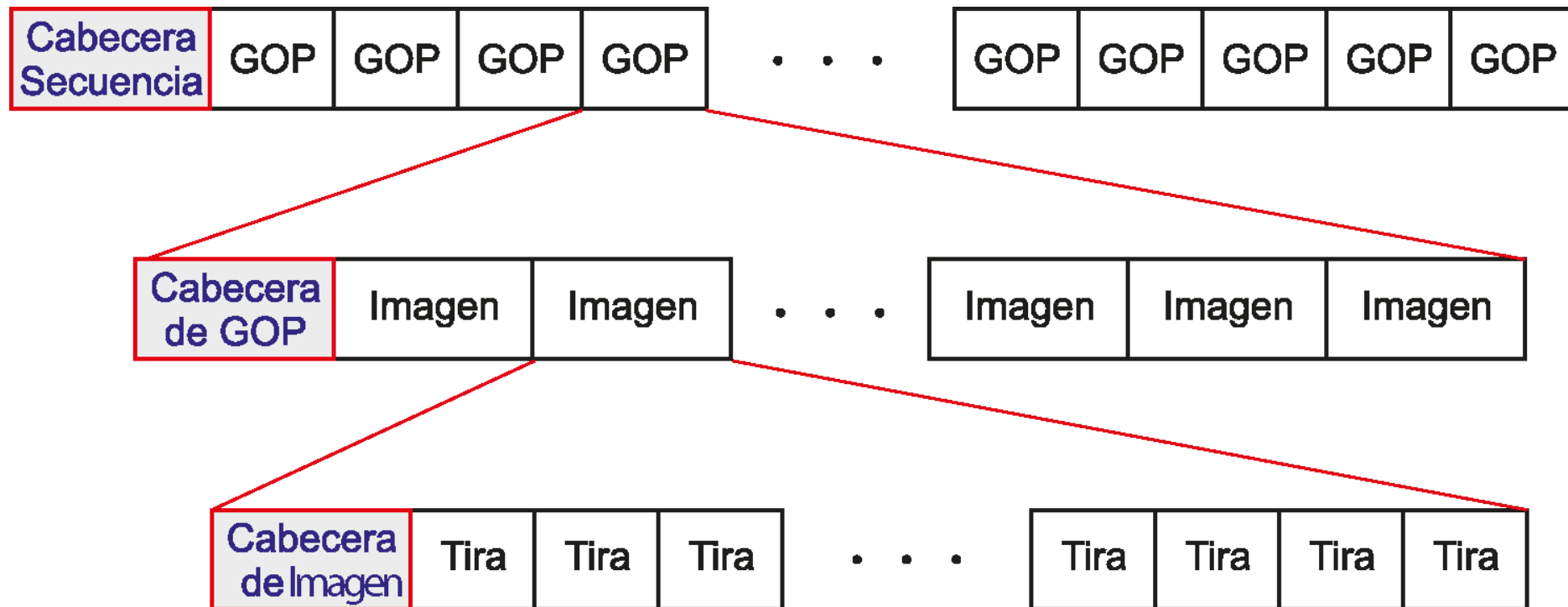
- Los valores más importantes codificados en la cabecera son:

- **group_start_code** → 32 bits “0x000001B8” señalando el comienzo del GOP.
- **closed_gop** → 1 bit que indica si el GOP incluye o no vectores de movimiento que referencien a GOP anteriores.
- **broken_link** → 1 bit que indica que las imágenes B que siguen a la primera imagen I no pueden descodificarse correctamente.

10. Estructura del flujo binario (5)

- **Nivel de imagen:**

- Una imagen se codifica como una cabecera de imagen seguida de una o más tiras.



10. Estructura del flujo binario (6)

- **Nivel de imagen:**
 - Los valores más importantes codificados en la cabecera son:
 - **picture_start_code** → 32 bits “0x00000100” señalando el comienzo de la imagen.
 - **temporal_reference** → 10 bits para el orden de presentación de la imagen.
 - Si vale 0 → Indica que es la imagen que debe presentarse en primer lugar.
 - Para sucesivas imágenes se incrementa en una unidad (módulo 1024).
 - **picture_coding_type** → 3 bits para el tipo de imagen. Valores usados: I(001), P(010), B(011), D(100).
 - **vbv_delay** → 16 bits usados sólo para velocidades binarias constantes. Indica el valor de llenado inicial del *buffer* al comienzo de la decodificación de cada imagen.
 - Si la velocidad binaria es variable ($\text{bit_rate} = \text{“0x3FFFF”}$), su valor es “0xFFFF”.

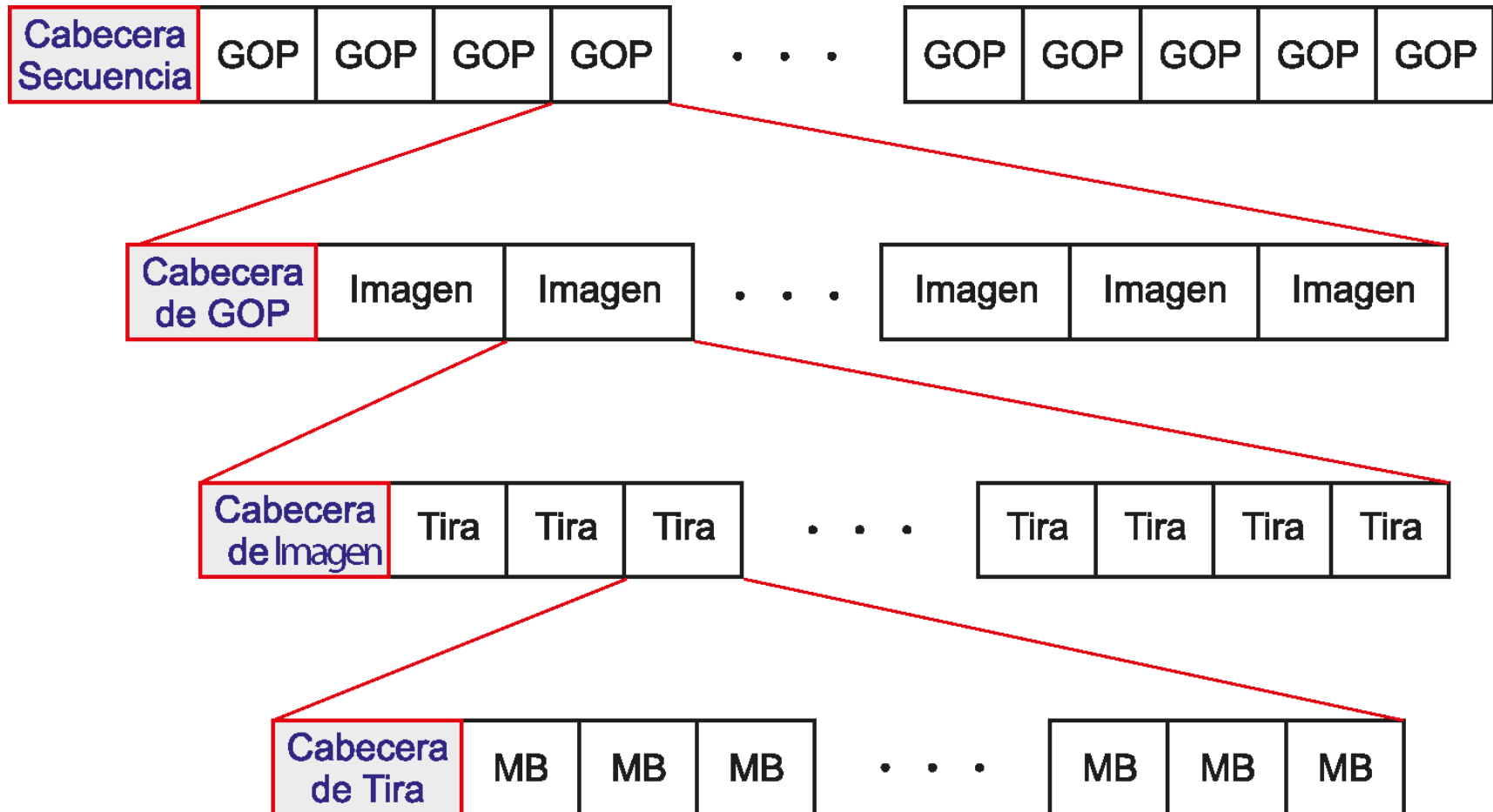
10. Estructura del flujo binario (7)

- **Nivel de imagen:**

- **full_pel_forward_vector** → 1 bit. Está presente sólo si la imagen es de tipo P o B.
 - 1 → Indica que los vectores de movimiento (hacia adelante) están basados en muestras enteras.
 - 0 → Indica que los vectores de movimiento (hacia adelante) están basados en medias muestras.
- **forward_f_code** → 3 bits. Indican el rango de las componentes de los vectores de movimiento hacia adelante. Sólo existe en las imágenes tipo P o B.
- **full_pel_backward_vector** → 1 bit. Está presente sólo si la imagen es de tipo B.
 - 1 → Indica que los vectores de movimiento (hacia atrás) están basados en muestras enteras.
 - 0 → Indica que los vectores de movimiento (hacia atrás) están basados en medias muestras.
- **backward_f_code** → 3 bits. Indican el rango de las componentes de los vectores de movimiento hacia atrás. Sólo existe en las imágenes tipo B.
- **extra_bit_picture** → Es opcional. Se usa si se desea incluir información adicional a nivel de imagen.

10. Estructura del flujo binario (8)

- **Nivel de tira:**
 - Cada tira se compone de una cabecera y un conjunto de macrobloques.

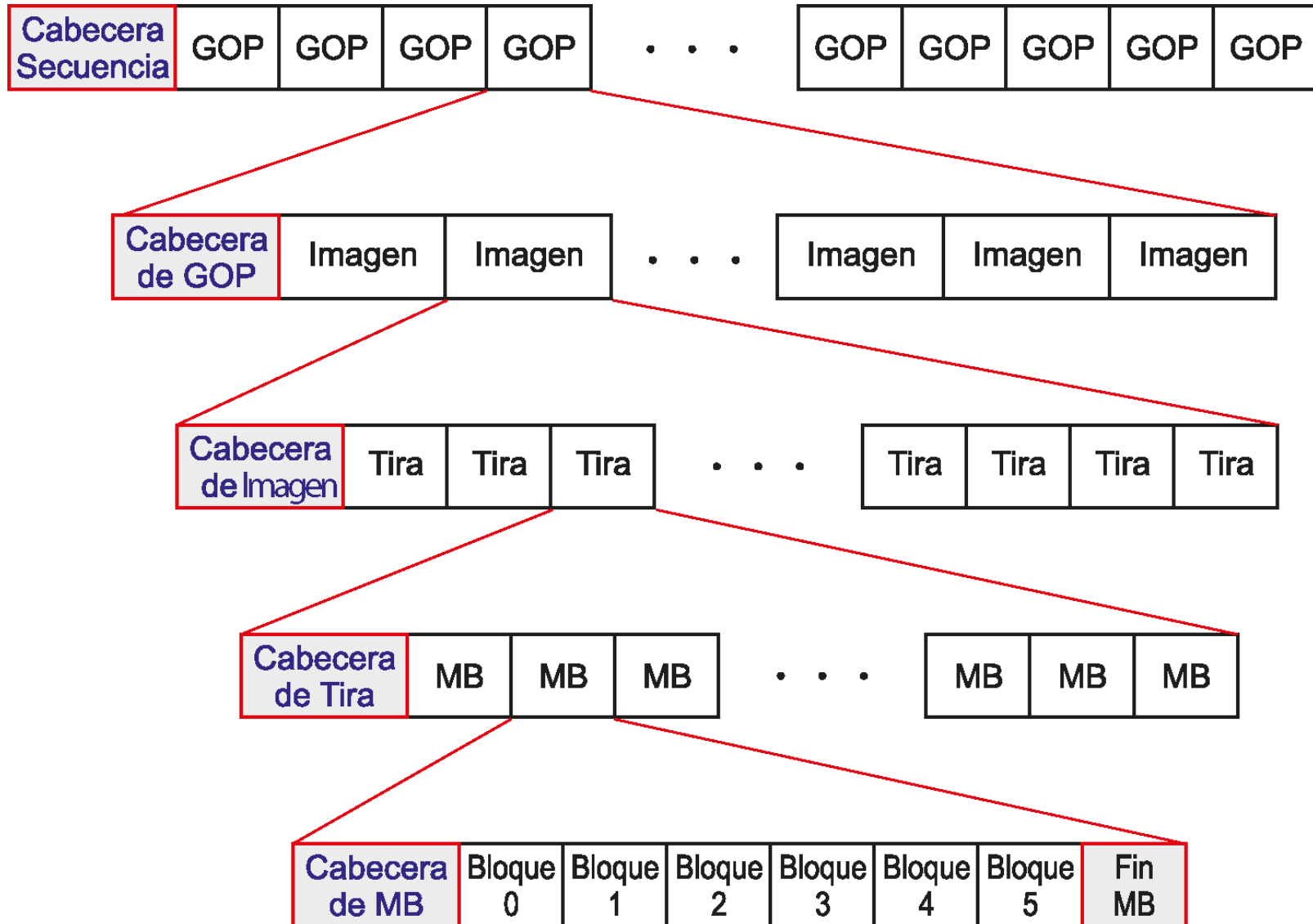


10. Estructura del flujo binario (9)

- **Nivel de tira:**
- Los valores más importantes codificados en la cabecera son:
 - **slice_start_code** → 32 bits.
 - Los 24 primeros son “0x000001” → Comienzo de una tira.
 - Los 8 últimos indican la posición vertical de primer MB de la tira:
 - Primera fila → 0x01.
 - Última fila posible → 0xAF.
 - **quantizer_scale** → 5 bits (con valores entre 1 y 31) que indican la escala de cuantificación utilizada para cuantificar los coeficientes de la DCT.
 - **extra_bit_slice** → Es opcional. Se usa si se desea incluir información adicional a nivel de tira.

10. Estructura del flujo binario (10)

- Nivel de macrobloque:



10. Estructura del flujo binario (11)

- **Nivel de macrobloque:**

- **macroblock_stuffing** → 11 bits opcionales “000 0000 1111”. Se utilizan como relleno para incrementar la tasa binaria si es necesario. Pueden usarse varios campos de este tipo seguidos.
- **macroblock_address_increment** → Diferencia entre la posición del MB y el último MB codificado.
 - Si vale 1 indica que el anterior MB fue codificado.
 - Si su valor es superior indica que ha habido MB anteriores sin codificar (*skipped*).
 - Su valor máximo es 33.
 - Si ha habido más de 32 MB consecutivos sin codificar, deberá usarse “macroblock_escape”.
- **macroblock_escape** → 11 bits opcionales “000 0000 1000”. Indican que hay que sumar el valor 33 al valor indicado por “macroblock_address_increment”.
- **macroblock_type** → Campo de longitud variable. Indica el tipo de codificación utilizada:
 - Si es intra, si hay que cambiar la escala del cuantificador, si se utiliza compensación de movimiento hacia delante o hacia atrás, si hay bloques sin codificar.

10. Estructura del flujo binario (12)

- **Nivel de macrobloque:**

- **quantizer_scale** → Factor de escala del cuantificador (entre 1 y 31). El decodificador usará este factor hasta que reciba uno nuevo (bien en la cabecera de la tira, bien en otro MB).
- **motion_horizontal_forward_code** → Indica la parte principal de la componente horizontal del vector de movimiento hacia delante.
- **motion_horizontal_forward_r** → Residuo de la componente horizontal del vector de movimiento hacia delante.
- **motion_vertical_forward_code** → Indica la parte principal de la componente vertical del vector de movimiento hacia delante.
- **motion_vertical_forward_r** → Residuo de la componente vertical del vector de movimiento hacia delante.

10. Estructura del flujo binario (13)

- **Nivel de macrobloque:**

- **motion_horizontal_backward_code** → Indica la parte principal de la componente horizontal del vector de movimiento hacia atrás. Sólo en imágenes tipo B.
- **motion_horizontal_backward_r** → Residuo de la componente horizontal del vector de movimiento hacia atrás. Sólo en imágenes tipo B.
- **motion_vertical_backward_code** → Indica la parte principal de la componente vertical del vector de movimiento hacia atrás. Sólo en imágenes tipo B.
- **motion_vertical_backward_r** → Residuo de la componente vertical del vector de movimiento hacia atrás. Sólo en imágenes tipo B.
- **codec_block_pattern** → Indica qué bloques están presentes. Es una secuencia binaria de la forma $Y_0Y_1Y_2Y_3C_B C_R$, donde un 1 indica que el bloque correspondiente ha sido codificado (alguno de sus coeficientes es distinto de 0).
- **end_of_macroblock** → Campo opcional de 1 bit. Sólo existe en imágenes tipo D.

10. Estructura del flujo binario (14)

- **Nivel de bloque:**

- Cada bloque de luminancia empieza con un campo **dct_dc_size_luminance** (indica el número de bits empleados para codificar el coeficiente de continua), seguido del valor de dicho coeficiente.
- El coeficiente de continua se codifica de forma diferencial:
 - Si su valor es 0 → Se envía el código 000.
 - Si su valor es $\{-1, 1\}$ → Se envía el valor 01 (indica que se necesita 1 bit para la codificación), seguido de un 0 (si el valor es -1) o un 1 (si el valor es 1).
 - Si su valor es $\{-3, -2, 2, 3\}$ → Se envía el valor 10 (indica que se necesitan 2 bits), seguidos de los correspondientes 2 bits.
 - Etc.
- A continuación se codifican los coeficientes de alterna (recorrido, nivel).
- Por último, se envía el código **end_of_block** (10) para indicar que no hay más coeficientes distintos de 0.
- Para los bloques de crominancia se hace lo mismo, excepto porque el número de bits empleados para codificar el coeficiente de continua es menor.

11. Sintaxis (1)

- A nivel de sintaxis, el estándar MPEG-1 define dos niveles: el de sistema y el de compresión.
- Nivel de compresión:
 - Está constituido por los ES (*Elementary Streams*: flujos elementales) de bits a los que dan lugar la codificación del vídeo y el audio.
 - Cada uno de estos ES está compuesto por:
 - AU (*Access Units*: unidades de acceso) → Representaciones codificadas de las imágenes y el audio.
 - *Private Data* → Datos privados adicionales.
- Nivel de sistema:
 - Define la sintaxis necesaria para combinar de forma ordenada los flujos elementales de audio y vídeo.
 - Añade información de distinto tipo para el decodificador pueda separar estos flujos, sincronizarlos y decodificarlos.

11. Sintaxis (2)

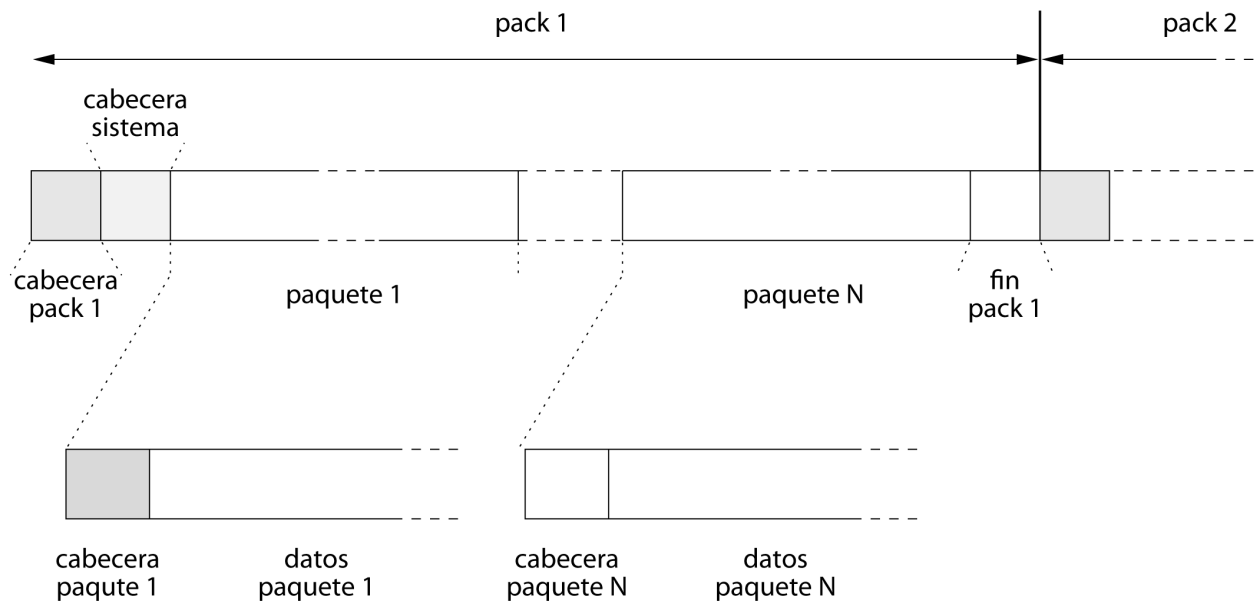
- **Funciones básicas del nivel de sistema:**

1. Sincronización de los flujos elementales de datos para su reproducción.
2. Combinación de los flujos elementales en un único flujo de datos.
3. Inicialización de los buffers al comienzo de la reproducción.
4. Gestión de los buffers → Control de la ocupación (ritmo de llenado) y de la extracción de datos durante la reproducción.
5. Identificación del tiempo de presentación → Inserción de marcas de tiempo (*time stamps*).

11. Sintaxis (3)

• Estructura del nivel de sistema:

- Cada ES se divide en paquetes (*packets*).
 - Cada paquete contiene una cabecera (*packet header*), seguida de datos.
 - Los datos de un paquete sólo pueden pertenecer a un canal (audio o vídeo).
- Los paquetes se agrupan en unidades de mayor tamaño denominadas *packs*.
 - También constan de una cabecera inicial con información de sincronización.
 - Un *pack* sí puede contener información de distintos canales.
- También existe una cabecera de sistema (*system header*) que aparece en el primer *pack*.
 - Contiene información general: identificación de los ES, tamaño mínimo del *buffer*...



11. Sintaxis (4)

- **Contenido de la cabecera de un paquete:**

- Código de comienzo de paquete → “0x000001”.
- Tipo de datos que contiene el paquete (4 bits) → Vídeo, audio o datos privados.
- Identificación del flujo de datos elemental al que pertenece en paquete (4 bits).
- Longitud del paquete (16 bits) :
 - La longitud de los paquetes no está fijada.
 - La longitud depende de distintos aspectos como, p.ej., el tipo de aplicación o el medio de almacenamiento.
- Bytes de relleno (0xFF):
 - Puede haber hasta un máximo de 16 bits de relleno (*stuffing bytes*).
 - Se utilizan para adaptar el tamaño del paquete a la dimensión de las unidades básicas del dispositivo de almacenamiento (dimensión de los sectores físicos).
- Tamaño del *buffer* (14 bits).
- Marcas de tiempos para la presentación (40 bits, opcional):
 - Se denominan PTS (*Presentation Time Stamps*).
 - Indican el instante en el que cada PU (*Presentation Unit*: unidad de presentación) ha de ser presentada.
- Marcas de tiempos para la decodificación (40 bits, opcional):
 - Se denominan DTS (*Decoding Time Stamps*).
 - Indican el instante de la decodificación de la primera AU del paquete.

11. Sintaxis (5)

- **Información de sincronización de los *packs*:**

- SCR (*System Clock Reference*: referencia del reloj del sistema):
 - Permite sincronizar en el receptor un reloj de referencia común (90 Hz) como el utilizado en el codificador.
 - Este reloj sirve de base de tiempos y para como unidad de medida para los campos DTS y PTS de los distintos paquetes contenidos en el *pack*.
- Velocidad de llenado del *buffer*:
 - Especifica la velocidad a la que se debe llenar el *buffer* con los datos del *pack* (en múltiplos de 50 Byte/s).

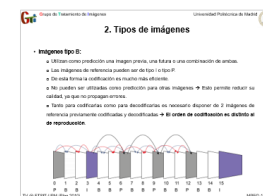


MPEG-1

Ejercicios

MPEG-1 – Enero 2017/18

- a) Dibujar el árbol de direccionamiento de macrobloques de una imagen P en MPEG-1 usando la siguiente nomenclatura para especificar las decisiones tomadas en las sucesivas bifurcaciones:
- ★ intra vs. non intra;
 - ★ motion compensation vs. no motion compensation;
 - ★ coded vs. not coded;
 - ★ quant vs. no quant.
- b) Justificar en cuál de las ramas de árbol anterior se genera menos información.



MPEG-1 – Julio 2015/16

Indicar las ventajas e inconvenientes del uso de las imágenes de tipo B en un codificador de vídeo conforme a la norma MPEG-2 Parte 2.



MPEG-1 – Enero 2015/16

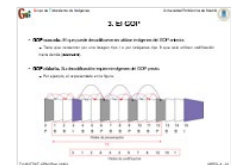
Se codifica una secuencia de 5 s y 125 imágenes de resolución CIF (288×352 píxeles) según el estándar MPEG-1, usando GOPs con $N=8$ y $M=2$, y buscando una velocidad binaria objetivo de 0,8 Mb/s. Indicar:

- a) La estructura del primer GOP codificado, y la del resto de GOPs.
- b) El orden de codificación de las imágenes 0 a 10.

MPEG-1 – Julio 2020

Considerar una secuencia de vídeo Full HD de 30 fps codificada según la norma MPEG-1 a $R = 30 \text{ Mb/s}$ y usando siempre un GOP de $N = 8$ imágenes (salvo quizá para el primer GOP).

- Indicar las posibles estructuras de los GOPs (distintos del primero), sabiendo que contienen imágenes de los tipos I, P y B.



MPEG-1 – Junio 2017/18 (1)

- a) Cuantificar y reconstruir, según el estándar MPEG-1, los coeficientes transformados mostrados en la esquina superior izquierda de la siguiente figura, usando la matriz de pesos de la esquina superior derecha y la escala $S_q = 12$.

298	-38						
370	13						

Coefficientes transformados

8	16						
16	16						

Matriz de pesos

Índices de los intervalos de cuantificación

Coefficientes reconstruidos

MPEG-1 – Junio 2017/18 (2)

- b) Calcular el error de cuantificación máximo posible para cada uno de los coeficientes analizados.

