

VcDemo

Contenido

Objetivos.....	2
Descripción.....	2
Formatos empleados	2
Técnicas de Compresión.....	3
Métricas de calidad de una imagen/vídeo.....	3
Imágenes y videos de test	5
Técnicas de compresión de imagen en VcDemo.....	6
Storing Module y Plot Module.	6
Técnica de Submuestreo (SS).....	7
Codificación PCM	8
Codificación Diferencial PCM: DPCM.....	9
Compresión 2D-DCT	10
Compresión JPEG	13
JPEG 2000	15
Técnicas de compresión de vídeo en VcDemo.....	17
ME (Estimación de Movimiento).....	17
MEnc (Mpeg 1/2 Encoder)	19
MDec (Mpeg 1/2 Decoder).....	20
Bibliografía adicional	23
ANEXOS y Referencias	24
Book - Understanding Compression:	24
Enlaces.....	24

Objetivos

A lo largo de esta práctica se pretende conocer la herramienta de compresión de imagen y vídeo VcDemo. Para ello, primero se hará una breve descripción de la aplicación detallando funciones, características, propiedades y curiosidades de la misma. Además se introducirán algunos conceptos y términos necesarios para la correcta comprensión tanto de su funcionamiento como de las técnicas de compresión con las que trabaja.

Descripción

El programa VcDemo fue creado por el Grupo de Teoría de la Comunicación e Información perteneciente a la universidad holandesa de [Delft University of Technology](#) ([Download Page](#)) Esta herramienta fue diseñada con el fin de realizar una sencilla demostración de cómo funcionan los sistemas de compresión de imagen y vídeo. La aplicación VcDemo es muy utilizada en ámbitos académicos debido a la gran cantidad de métodos de compresión que permite emplear, así como, los diferentes parámetros que permite modificar, con el objeto de observar de forma empírica el resultado y poder realizar análisis y comparaciones entre los diferentes sistemas de compresión.

Formatos empleados

La herramienta VcDemo permite trabajar con distintos tipos de formatos origen (sin comprimir), entre los que se encuentran:

- Imagen:
 - **Bitmap Image** (*.bmp) → Un archivo BMP (*Bit Map Picture*) es un archivo de mapa de bits, que contienen los píxeles de la imagen almacenados como una tabla de valores de un espacio de color (típicamente RGB). Estos archivos comienzan a leerse desde abajo (última fila de la imagen) hacia arriba. Sólo se admiten imágenes con una profundidad de pixel de 8 o 24 bits.
 - **Tiff Image** (*.tif) → Los archivos del tipo TIFF (*Tag Image File Format*) hacen referencia a un formato fotográfico digital que permite la especificación flexible de cómo está codificada y almacenada la imagen (admite una gran variedad de formatos sin compresión y con compresión).

- Vídeo:
 - Raw YUV Image Sequence (*.yuv).
 - Raw Y Image Sequence (*.y).
 - Sequence of Bmp Images (*.seq).

- Stream:
 - Mpeg Stream (*.mpg).
 - Mpeg-1 Video Stream (*.m1v)
 - Mpeg-2 Video Stream (*.m2v)
 - H.264 Stream (*.264).

Técnicas de Compresión

El programa VcDemo permite emplear una gran variedad de mecanismos de compresión para los formatos ya mencionados. De éste modo, se puede utilizar las siguientes técnicas asociadas a la compresión:

- Imagen:
 - Submuestreo,SS
 - Codificación PCM
 - Codificación Diferencia, DPCM.
 - Codificación de Imagen Fractal, FRAC.
 - Transformada Discreta del Coseno
 - Compresión JPEG
 - Codificación en Sub-bandas.
 - Codificación EZW.
 - SPITH.
 - JPEG2000.

- Vídeo y Stream:
 - Técnica de Estimación del Movimiento, ME.
 - Codificación Mpeg.
 - Codificación H.264.
 - Decodificador Mpeg.
 - Decodificador H.264.

La herramienta presenta un módulo de acceso rápido para cada una de las anteriores técnicas. Adicionalmente podemos encontrar otros módulos que permiten emplear otras funciones como la creación de gráficas o el reproductor de video. Más adelante describiremos el funcionamiento de cada una de estas técnicas y módulos así como los parámetros importantes

Métricas de calidad de una imagen/vídeo

Cuando se aplica un proceso de compresión a una imagen y/o vídeo, se introduce un ruido de cuantización que produce pérdidas irreversibles de la información original. Al descomprimir la imagen se obtiene una versión diferente de la original. Para medir la cantidad de ruido introducido o bien la degradación de calidad de la imagen/video reconstruida se utilizan métricas de calidad objetivas. Las métricas objetivas más conocidas son las siguientes:

- Error cuadrático medio, MSE.
- La relación Señal-Ruido, SNR.
- Pico de la relación Señal-Ruido, PSNR.

Existen otras métricas que son empleadas con el mismo fin, pero con las que no trabaja directamente el programa VcDemo. Por ejemplo, el valor denominado RECM: Raíz del Error Medio Cuadrático o la razón señal-ruido máxima (RSRM).

$$RSRM = 20 \log_{10} \frac{255}{RECM} = 20 \log_{10} \frac{255}{\sqrt{\frac{1}{\#pixels} \sum_{i,j} (\hat{p}_{i,j} - p_{i,j})^2}}$$

Siendo $p_{i,j}$ las intensidades de los pixeles en la imagen original y $\hat{p}_{i,j}$ el valor de las intensidades de los pixeles en la imagen obtenida.

Otro parámetro empleado a la hora de hablar de calidad de una imagen podría ser el término de *Correlación Cruzada* (CC). La *Correlación Cruzada* hace referencia al grado de dependencia estadística entre dos imágenes:

$$CC(I, I') = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} |I_{i,j} \cdot I'_{i,j}|^2}{\sqrt{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} |I_{i,j}|^2} \sqrt{\sum_{i=0}^{N-1} \sum_{j=0}^{M-1} |I'_{i,j}|^2}}$$

A. Error Cuadrático Medio, MSE.

El error cuadrático medio (MSE) hace referencia al error cuadrático acumulado entre el comprimido y la imagen original. Se obtiene entre una imagen original de $M \times N$ pixeles, I , y su reconstrucción, K . Su valor obtiene del siguiente modo:

$$MSE = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} ||I(i, j) - K(i, j)||^2$$

B. Relación Señal-Ruido (SNR).

La relación Señal/Ruido relaciona la potencia de la señal que es transmitida y la potencia del ruido que la corrompe. Esta magnitud se mide de decibelios (dB).

C. Relación Señal a Ruido de Pico (PSNR).

El PSNR (*Peak Signal-to-Noise Ratio*) permite definir la relación entre el ruido que afecta a una señal y la máxima energía posible de la señal. Al igual que en el caso de la SNR, la unidad de esta magnitud es el decibelio (dB).

El principal uso del valor PSNR es como medida cuantitativa de la calidad de la reconstrucción tras un proceso de compresión/descompresión sobre una imagen (o cuadro de vídeo). El valor del PSNR viene determinado por la siguiente función:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) = 20 \cdot \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right),$$

Donde MAX_I hace referencia al máximo valor que puede tomar un píxel de la imagen. Suponiendo que cada píxel de la imagen ocupa B bits, su valor será:

$$MAX_I = 2^B - 1$$

Por lo tanto, a la hora de tomar medidas cuanto más bajo sea el MSE y más altos sean los valores de SNR y PSNR obtenidos mayor será la calidad de la imagen.

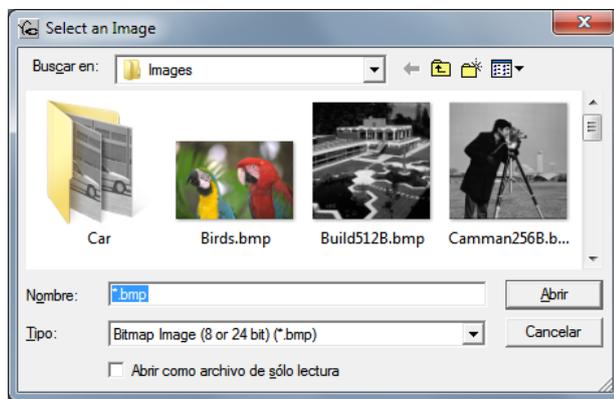
Para realizar comparaciones de las técnicas empleadas y de los distintos parámetros configurados, además de realizar tablas con los resultados obtenidos, se puede hacer uso de dos módulos que integra el programa: “*Storing Module*” y “*Plot Module*”, que nos permitirán construir gráficas de rendimiento R/D.

Imágenes y videos de test

Para trabajar con los sistemas de compresión incluidos en VcDemo, necesitamos imágenes y secuencias de vídeo originales (sin comprimir). VcDemo incluye un conjunto reducido de imágenes y vídeos que ya están disponibles tras la instalación y que son los que utilizaremos en esta práctica. No obstante, VcDemo puede trabajar con otras imágenes y vídeos siempre y cuando tengan formatos compatibles con los que VcDemo es capaz de leer. Como ejemplo, en compresión de imagen se suele utilizar mucho la imagen Lena256B (Figura 1(a)), en este caso una versión en escala de grises de 256x256 píxeles y una profundidad de pixel de 8 bits (sólo luminancia - plano Y). Esta imagen presenta buenas mezclas de detalles, de zonas homogéneas, y unas texturas que son interesantes para observar el comportamiento de los diferentes algoritmos de compresión de imagen.



(a)



(b)

Fig1. (a) Imagen de muestra Lena256B. (b) Ventana para seleccionar imagen

Para cargar una imagen en VcDemo, desde el menú “*File*” seleccionamos la entrada “*Open Image*” mostrando una ventana (Figura 1(b)) donde podremos seleccionar la imagen con la que queremos trabajar¹. Una vez seleccionada la imagen se carga en una ventana de la aplicación. Pulsando con el botón derecho del ratón sobre la imagen aparecerá un menú contextual desde donde podremos realizar diferentes operaciones sobre ella.

¹ VcDemo almacena todas las imágenes que lleva en la carpeta “*Images*” ubicada en el directorio donde se instale.

Técnicas de compresión de imagen en VcDemo.

En esta sección revisaremos algunas técnicas incluidas en VcDemo relacionadas con la compresión de imagen. En primer lugar, se indica cómo se pueden capturar medidas de rendimiento (*Storing Module*) y representar sus valores en gráficas (*Plot Module*). Estos módulos serán de utilidad cuando queramos comparar diferentes técnicas de compresión u observar el efecto de una técnica de compresión modificando alguno de sus parámetros de configuración.

A continuación iremos viendo algunas de las técnicas de compresión incluidas en el programa mostrando brevemente sus características. Para más información sobre cada una de ellas y descripciones detalladas sobre cada uno de los parámetros de configuración y/o el interfaz de usuario del programa, se dispone de una ayuda en línea accesible en la barra de menú “*Help*”, entrada “*VcDemo Help (F1)*”. La notación que seguiremos para indicar entradas de menú será “*Help → VcDemo Help (F1)*”

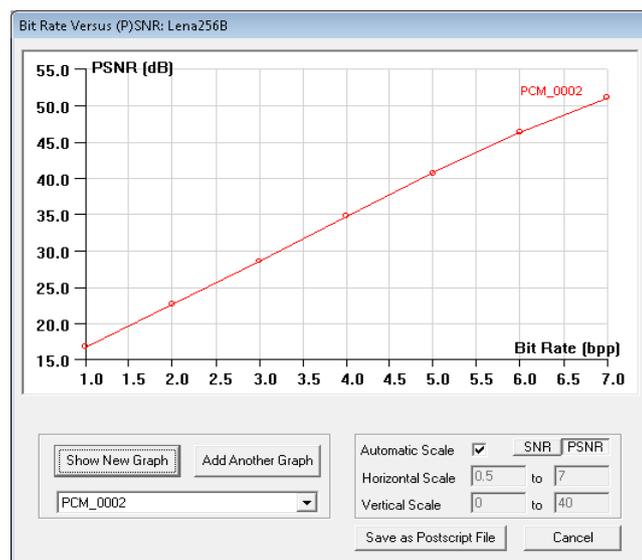
Storing Module y Plot Module.

Storing Module tiene como finalidad la captura de los valores de PSNR y SNR que se obtienen cuando se hace una operación de compresión de imagen (siempre y cuando ofrezcan este valor como resultado). Para utilizar es módulo, primero tenemos que seleccionar una operación de compresión sobre una imagen original (p.e. PCM con los valores de configuración que vienen por defecto) pulsando el botón derecho del ratón en la imagen original sobre la que se va a realizar la compresión, obtenemos un menú contextual. Seleccionamos la entrada “*Basic Compression → PCM coding*”. A continuación, a través del menú contextual, seleccionamos la entrada “*Create R/D Graph → Store R/D Points*”. Tras esto se abre una ventana en donde podemos iniciar el almacenamiento de resultados (Figura 2(a)). Para detener la captura de resultados, pulsaremos el botón “*Stop*” justo después de la última compresión de la que se desea tener un valor representado en la gráfica. Una vez tomadas las medidas deseadas, quedarán guardadas con el nombre que indique el usuario (*Profile Name*) o con un nombre por defecto.

Figura 2. Ventanas (a) “*Store R/D Points*” y (b) “*Plot R/D Points*”



(a)



(b)

Una vez terminado el proceso de captura de medidas R/D (*Rate/Distorsion*) podremos utilizar el módulo de gráficas (*Plot Module*). Éste módulo permite al usuario cargar los resultados correspondientes de las distintas medidas almacenadas (tasa de bits y SNR o PSNR) para representarlas en una gráfica de puntos. Para ello, se accede a través de la entrada del menú contextual “*Create R/D Graph → Plot R/D Points*” (Figura 2(b)). Si tenemos varias capturas ya realizadas, cada una con su propio nombre (*Profile Name*), podemos seleccionarlas con un desplegable que se encuentra debajo de este botón. Cada vez que queramos refrescar el gráfico o mostrar uno nuevo, tenemos que pulsar el botón “*Show New Graph*”. También podemos añadir más curvas, seleccionando los valores almacenados en el desplegable y pulsando el botón “*Add Another Graph*”. Si queremos almacenar la gráfica en un archivo, este módulo permite guardar la gráfica en formato *Postscript* (.eps ó .ps files).

Éste módulo es de gran utilidad para poder realizar comparaciones entre distintos tipos de técnicas de compresión o comparaciones entre una configuración u otra de una misma técnica de compresión (sensibilidad de los parámetros de configuración).

Técnica de Submuestreo (SS).

La técnica de submuestreo (en inglés *downsampling*) se basa en disminuir la frecuencia de muestreo con la que se capturó originalmente la imagen original, obteniendo una versión de la imagen reducida (con menos información). Este módulo permite fijar el factor de submuestreo entre 2 y 64. Así, según sea el valor de este parámetro, la imagen submuestreada tendrá “n x n” veces menos resolución que la original (se realiza el submuestreo en las dos dimensiones). En la siguiente tabla se puede comprobar el tamaño obtenido tras aplicar el submuestreo a la imagen original (tamaño 256x256) con diferentes factores:

Original	Factor 2	Factor 4	Factor 8
256x256	128x128	64x64	32x32
	Factor 16	Factor 32	Factor 64
	16x16	8x8	4x4

Técnica SS: Relación tamaño imagen – factor de submuestreo.

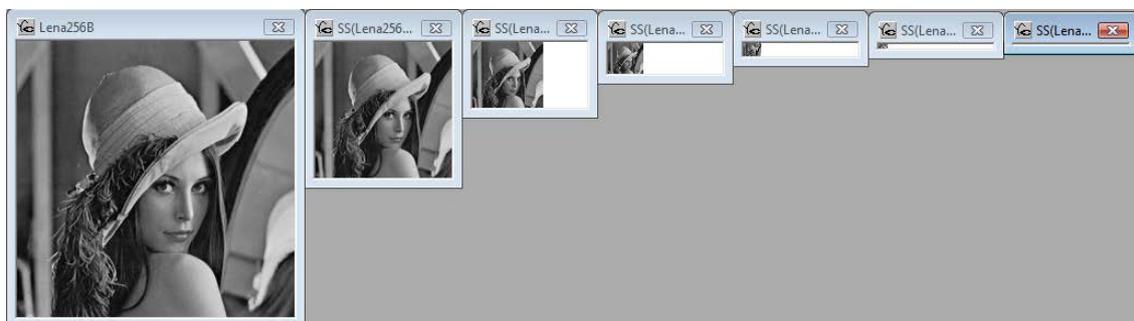


Figura 3. Imagen original y versiones submuestreadas con factores 2, 4, 8, 16, 32 y 64

Para realizar una operación de submuestreo sobre una imagen, pulsando con el botón derecho sobre la imagen y seleccionando las entradas “*Basic Compression* → *Subsampling*”² del menú contextual, se abre la ventana de la herramienta subsampling asociada a la imagen. En esta ventana aparecen los distintos parámetros de configuración de esta herramienta (en todas las técnicas de compresión VcDemo se utiliza la misma filosofía y forma de proceder).

Como se puede apreciar, este módulo no tiene activada la operación de submuestreo por defecto, por ello lo primero que haremos será activar la casilla “*Apply Subsampling*”. Esta técnica permite reconstruir la imagen submuestreada al tamaño de la imagen original (opción “*Blow-up Subsampled Image*” activada), lo que nos permitirá comparar la calidad de la imagen reconstruida con respecto a la original. Para hacer la reconstrucción al tamaño original se emplea un filtro de interpolación cuyos parámetros están definidos en la pestaña “*Filter*”. En la figura 3 se pueden comprobar los resultados obtenidos para cada uno de los factores de submuestreo con la opción “*Blow-up Subsampled Image*” desactivada.

Ejercicio: Con la misma imagen original realizar el submuestreo para los mismos factores que los aplicados en la figura 3, con la opción “*Blow-up-Subsampled Image*” para poder observar la degradación de calidad de la versión reconstruida. No aplicar ningún filtro en la reconstrucción de la imagen.

Ejercicio: Seleccionar un par de valores de submuestreo con los que vayamos a trabajar y en este caso aplicar filtros de distintos tamaños en la reconstrucción para apreciar cómo el filtro mejora la calidad de reconstrucción de la imagen.

Además del parámetro referente al factor de submuestreo y del filtro de interpolación hay que tener en cuenta otros factores como la aplicación de un filtro anti-alias y la visualización del espectro resultante del submuestreo.

Codificación PCM

Éste mecanismo de codificación no es más que un proceso de cuantización uniforme escalar como el utilizado en la digitalización de señales análogicas (convertidores A/D). A cada muestra de la señal se le asigna un número de B bits para su codificación, definiendo 2^B niveles de cuantificación. Por ello, cuanto mayor sea el rango de bits por muestra empleados en una imagen, se podrá representar una mayor gama de tonos de grises, ofreciendo una mayor calidad de imagen.

Para trabajar con la codificación PCM, seleccionamos la entrada “*Basic Compression* → *PCM Coding*” en el menú contextual de la imagen para obtener la ventana del PCM. En ella se muestran tres pestañas: *Bitrate*, *Dithering* y *Errors*. En la primera se puede especificar el número de bits por muestra que queremos aplicar a cada pixel de la imagen, definiendo pues el bitrate resultante en bits por pixel (bpp en *PCM bitrate*). En la segunda se puede utilizar la técnica de *dithering*, que introduce un ruido en la cuantización (con cuatro niveles de intensidad) para tratar de mejorar la calidad de las imágenes

² Muchas opciones del menú se pueden invocar directamente con la barra de botones de acción que se encuentra justo debajo de éste. Así en el caso del submuestreo, el botón asociado es el que contiene las letras “SS”

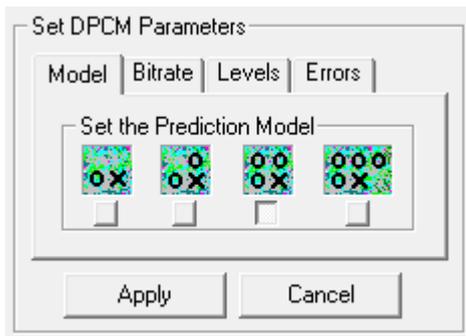
reconstruidas a altas tasas de compresión. Y en la tercera, se emula la transmisión de la imagen por un canal de comunicación con errores, pudiendo seleccionar diferentes BERs (*Bit Error Rate*).

Ejercicio: Sobre la imagen Lena256B, aplicar la codificación PCM a 1, 2, 3, 4 y 5 bits por pixel, sin la opción de *dithering* y un canal de comunicación sin errores.

- (a) Muestra en una tabla los resultados de calidad (*Mean Square Error* y PSNR) obtenidos para cada bitrate.
- (b) Para la tasa de compresión más alta (1 bit por pixel)
 - i. Calcular el tamaño de la imagen comprimida resultante y obtener el factor de compresión.
 - ii. Aplicar la técnica de *dithering* utilizando los cuatro niveles de intensidad (con la opción “*subtract dither*” desactivada) y muestra una tabla con los resultados de calidad (*Mean Square Error* y PSNR), comparando las diferencias con los obtenidos en (a).
 - iii. Fijando el nivel de *dithering* a 0.75 (*dithering stepsize*) analiza el impacto que tiene en la calidad de la imagen reconstruida cuando activamos la opción “*Substract Dither*”.

Codificación Diferencial PCM: DPCM.

La técnica de codificación DPCM es un sistema de compresión basado en la predicción lineal de los valores de los píxeles vecinos, codificando únicamente la diferencia o error cometidos en la predicción.



Tras invocar la codificación DPCM en el menú contextual de la imagen original, nos aparece la ventana de configuración con cuatro pestañas:

- ✓ “**Model**”, indica el tipo de predictor lineal que vamos a utilizar. El pixel indicado con una cruz es el que vamos a predecir, mientras que los píxeles representados con un círculo serán los píxeles vecinos que se van a utilizar para predecir a éste.
- ✓ “**Bitrate**”, especifica el número de bits que vamos a asignar al error cometido en la predicción de un pixel (de 1 a 6 bits). A menor número de bits, mayor tasa de compresión y dependiendo de la imagen (similitud entre píxeles vecinos) menor calidad.
- ✓ “**Levels**”, establece el número de niveles de cuantización que se van a utilizar
- ✓ “**Errors**”, se utilizará para emular la transmisión de la imagen por un canal con errores.

Cada vez que realizamos una codificación DPCM se muestra una ventana con la representación del error de predicción en toda la imagen. La imagen del error de predicción se muestra en una escala que varía del gris (no hay error) a blanco (error máximo).

Ejercicio: Fijando el valor del *DPCM bitrate* a 2 bits por pixel, codifica la imagen Lena256B con cada uno de los modelos de predicción que vienen con el VcDemo.

- (a) Muestra en una tabla los valores de calidad de las imágenes reconstruidas (*Mean Square Error* y *PSNR*).
- (b) Calcula el tamaño de la imagen comprimida.

Ejercicio: Compara las técnicas PCM y DPCM utilizando la imagen Lena256B usando tasas de bits entre 1 y 5 bits por pixel.

- Utiliza la mejor configuración encontrada para cada técnica de compresión y muestra un gráfico con las curvas de R/D (bitrate/PSNR) de ambos compresores.
- Crea gráfico de “Dispersión con líneas suavizadas y marcadores” en excel para ver las curvas R/D.
- Crea un gráfico de R/D con vcdemo y plotealo.
- Compara ambos gráficos.

Compresión 2D-DCT

La transformada discreta del coseno es capaz de compactar la energía de la señal en unos pocos coeficientes.

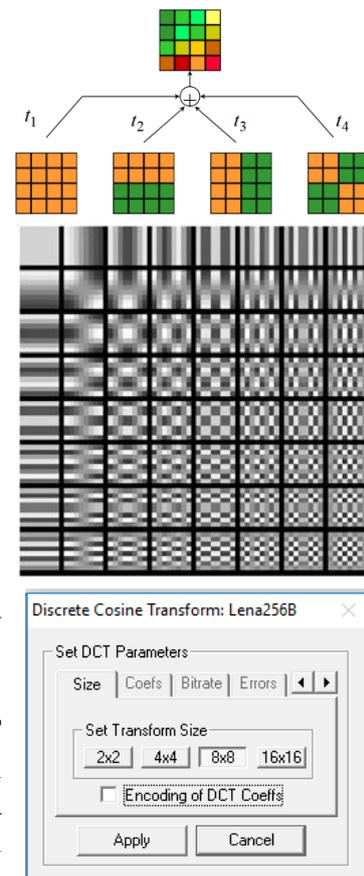
La idea subyacente de la aplicación de la DCT a imágenes es que una imagen puede ser construida como combinación lineal de unas determinadas imágenes base.

Estas imágenes base (bases) se obtienen al aplicar distintas frecuencias la función coseno para generar patrones de frecuencia espacial.

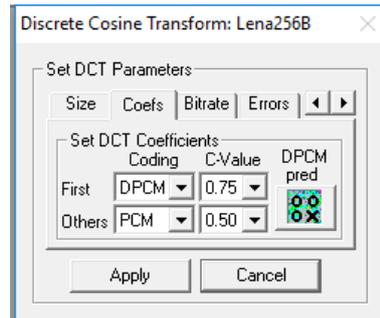
Normalmente en vez de trabajar con imágenes completas se trabaja con bloques de NxN bits, lo más habitual es 8x8.

La imagen muestra estas bases, cuando aplicamos la función coseno para crear patrones frecuenciales por filas y por columnas.

Cuando a un bloque de una imagen, se le aplica la 2D-DCT obtenemos una matriz de coeficientes DCT que representan el peso (la cantidad de la base) que hay que aplicar a cada base para ser sumada con el resto y componer la imagen original.



En VcDemo disponemos de un módulo que nos permite comprimir una imagen utilizando la 2D-DCT (DCT a partir de ahora), permitiendo seleccionar diferentes tamaños de bloque, 2x2, 4x4, 8x8 y 16x16. Se descompone la imagen en bloques del tamaño seleccionado y se aplicará la DCT a cada uno de ellos, obteniendo una matriz de coeficientes para cada bloque.



Estos coeficientes pueden ser codificados y comprimidos. El primer coeficiente de la matriz (esquina superior izquierda) representa la continua de la señal, una representación escalada de la señal, le llamaremos “continua” y al resto “coeficientes DCT”.

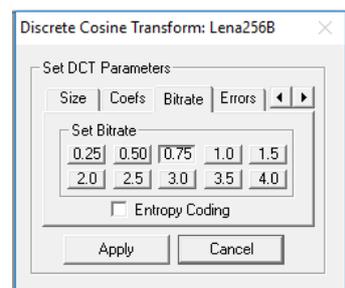
El software permite decidir si queremos utilizar el método PCM o el DPCM para comprimir los bloques. Permite distinguir cómo codificamos la continua de los coeficientes DCT, pudiendo elegir para cada uno (continua o coeficientes) entre PCM o DPCM. Tener en cuenta que la varianza en el bloque de continua es muy superior a la de los bloques de coeficientes pues es una versión escalada de la imagen. Una vez codificados se puede pedir que se use un codificador entropico VLC para comprimir los coeficientes o sus diferencias.

VcDemo nos ofrece una interfaz para comprimir con la DCT compuest por una serie de pestañas:

Size: The size of the DCT transform blocks can be selected (2x2, 4x4, 8x8, 16x16). The quantization of the DCT coefficients can be switched off to measure the quality of forward and reverse DCT transform.

Coefs: The quantization of the DCT coefficients can be selected. The first DCT coefficient (mean value in the DCT block) can be compressed independently of the higher frequency DCT coefficients. For the DCT coefficients a choice can be made between PCM and DPCM. The quantizers considered for each DCT coefficient depend on the selected PDF. The c-value specifies the shape parameter of a generalized Gaussian PDF. If DPCM is chosen, the DPCM prediction model can be selected. The DCT coefficients with the same index (originating from different DCT blocks) are all quantized with the same quantizer. In this way the bit allocation can optimize the allocation of the bits to all DCT coefficients. Thus the DCT compression implemented here has a global quantization behavior. In contrast to this, JPEG is a DCT compression scheme with local quantization behavior.

Bit rate: Different bit rates can be selected. The limited number of choices here has been hard coded, but is not essential to DCT compression as the bit allocation procedure (implemented here as the greedy or convex hull algorithm) can achieve any desired (overall) bit rate. Normally entropy coding (Huffman) is applied to the quantizer outputs, but in order to accommodate

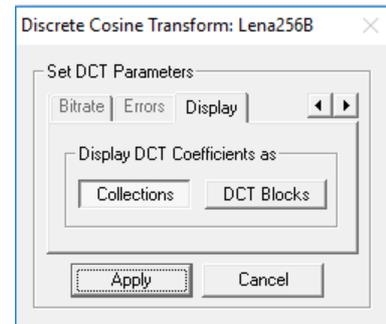


experiments with channel bit errors, the entropy coding should be switched off.

Errors: This tab allows the injection of random bit errors into the compressed bit stream, prior to decompression. In this way an impression can be obtained of the effects of (simple) channel errors. Different bit error rates can be selected, namely 0.005 (0.5% of the bits are erroneous), 0.001, 0.0005, and 0.0001. Bit errors can only be injected if the entropy coding of the quantizer output levels has been switched off.

Display: The DCT coefficients obtained after transformation and after quantization can be displayed in two ways, namely:

- group by index (collections) of DCT coefficients. In this case the DCT coefficients with the same index from all DCT blocks are collected in a single set. These sets are much like subband coefficients, and give an easy to understand impression of the bit allocation result,
- group by DCT Blocks in the correct spatial location. In this case the DCT coefficients are shown as NxN blocks that sit in the spatial position corresponding to the NxN pixels the DCT coefficients are calculated from. For visibility purposes the DCT coefficients are scaled. This display mode gives an easy to understand impression of the local frequency content of an image, and – after the bit allocation – an impression of which part of an image is easy or hard to compress.



Como resultado de la aplicación de los parámetros de codificación VcDemo nos devuelve la imagen reconstruida y la representación de los distintos coeficientes DCT obtenidos para la imagen en dos formas: en bloques DCT y en Collection, como se ha comentado en la pestaña Display anteriormente.

The DCT coefficients of the image to be compressed are shown to the right of the start image. The DCT coefficients are organized according to the Display mode selected. In order to visualize small DCT coefficients, the values are scaled to maximum intensity range. “Gray” means zero value, and negative and positive values are darker and brighter, respectively, than the zero value. The actual variance of the subbands can be found in the text window. Because of the scaling, no direct interpretation of the importance (variance) of DCT coefficients can be derived from the displayed data.

After quantization and VLC encoding, the DCT coefficients are transmitted and decoded by the receiver. The quantized DCT coefficients are shown immediately below the original DCT coefficients. DCT coefficients that have received zero bits in the bit allocation are entirely gray. The actual bit allocation results can be found in the text window. Again, DCT coefficients are scaled for maximum visibility. The effect of channel errors is also visible in the individual DCT coefficients. The quantized DCT coefficients are organized depending on the Display mode selected. The left image shows an example of quantized DCT coefficients if display mode Collection is selected, while the right image shows the DCT coefficients if the option DCT Blocks is selected.

Ejercicio: Codifica la imagen Lena256B con bloques de 2x2, 4x4, 8x8 y 16x16.

- Observa los bloques DCT y las colecciones (subbandas) para cada tamaño de bloque.
- Para bloques de 8x8, habilita la codificación de los coeficientes DCT sin codificación entrópica y observa cómo para distintas tasas de compresión el conjunto de coeficientes que sobreviven a la cuantización varía. ¿Que conclusiones puedes sacar de esto?
- Crea un gráfico R/D comparativo en excel de la compresión con DCT cuando se usa codificación entrópica y cuando no. Utiliza al menos 5 valores de bitrate. Utiliza la mejor configuración que puedas para la codificación de la continua y los coeficientes DCT y mantenla fija para los distintos bitrates. Analiza los resultados.

Compresión JPEG

JPEG es un standard de compresión de imágenes en tono-continuo propuesto por el grupo Joint Photograph Experts Group (<http://www.jpeg.org/>) y aprobado como estándar ISO en 1992.

El algoritmo de compresión JPEG utiliza fenómenos visuales del ojo humano como (a) el hecho de que es mucho más sensible al cambio en la luminancia que en la crominancia; es decir, capta más claramente los cambios de brillo que de color (por eso se aplica un submuestreo 4:1:1 a la imagen original). (b) Se aprecian con más facilidad pequeños cambios de brillo en zonas homogéneas que en zonas donde la variación es grande; por ejemplo en los bordes de los cuerpos de los objetos de la imagen (baja sensibilidad a las altas frecuencias espaciales → uso de la DCT).

Una de las características del JPEG es la flexibilidad a la hora de ajustar el grado de compresión. Un grado de compresión muy alto generará un archivo de pequeño tamaño, a costa de una pérdida significativa de calidad. Con una tasa de compresión baja se obtiene una calidad de imagen muy parecida a la del original, pero con un tamaño de archivo mayor.

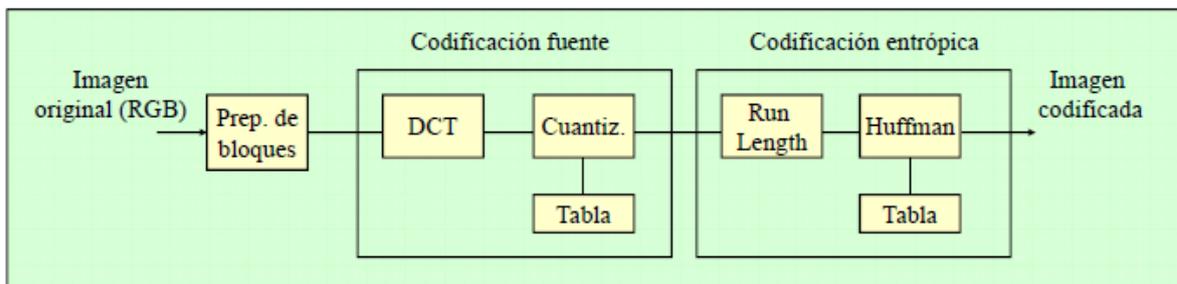


Figura 4. Diagrama de bloques del compresor JPEG

Tal y como se muestra en la figura 4, el compresor JPEG prepara la imagen para realizar un procesado por bloques de 8x8 píxeles usando un espacio de color YCbCr con submuestreo 4:1:1. Cada bloque sufre una codificación independiente basada en dos etapas:

- i. Codificación basada en la fuente, mediante una transformada DCT y una cuantización escalar uniforme basada en tabla.
- ii. Codificación basada en la entropía por medio de (a) una codificación DPCM de para la componente en continua (elemento (0,0) del bloque), y (b) una codificación *run-length* (RLE) más una codificación VLC (*Variable Length Codes*) de tipo *Huffman* para el resto de coeficientes cuantizados.

Cada bloque de 8x8 procedente de la preparación de la imagen es procesado de forma independiente, almacenando su resultado (tira de bits) en el bitstream de salida.

En VcDemo, accedemos a la herramienta de compresión JPEG seleccionando la entrada “*Linear Transforms* → *JPEG (DCT) Compression*” del menú contextual de la imagen.

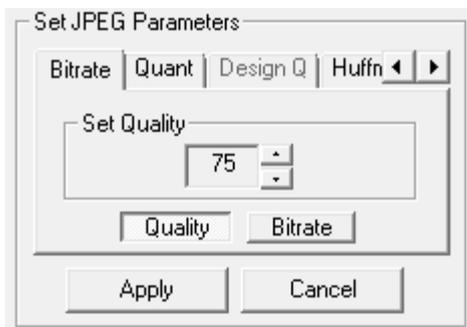


Figura 5. Parámetros de JPEG

Los parámetros de configuración de JPEG se muestran en la Figura 5, en donde se encuentran las siguientes pestañas:

- ✓ “**Bitrate**” permite definir el nivel de cuantización que se va a aplicar, bien indicando un índice de calidad, *Quality*, cuyo valor varía de 1 a 100 (siendo 100 la máxima calidad), o bien mediante un valor de *Bitrate* en que varía entre 0.1 y 8 bits por pixel (bpp) (siendo 8 bpp el equivalente a no comprimir).
- ✓ “**Quant**” especifica las tablas de cuantización que se van a utilizar, entre las que se pueden definir las propuestas por el estándar tanto para la luminancia (*StandardY*) como para los planos de color (*StdUV*), así como una tabla de cuantización plana (*Flat*) y otra para preservar las altas frecuencias (*HighPass*).
- ✓ “**Huffman**” incluye tres aproximaciones a seguir en la codificación entrópica de los coeficientes cuantizados: (a) *FLC (Fixed Length Codes)* no utiliza *Huffman*, (b) *Standard VLC* utiliza las tablas *Huffman* propuestas en el estándar, y (c) *Optimal VLC* calcula las tablas *Huffman* óptimas para la imagen a comprimir.
- ✓ “**Smooth**” aplica un suavizado sobre la imagen descomprimida para reducir las distorsiones que aparecen a altas tasas de compresión. No afecta al proceso de compresión, sólo a la calidad visual.
- ✓ “**Markers**” está relacionado con la compresión entrópica (*Huffman*) y la robustez ante errores. Son marcas que se ponen en el bitstream comprimido que actúan como puntos de resincronización en la decodificación *Huffman*, que permiten recuperar la decodificación cuando se producen errores.
- ✓ “**Errors**” simula la generación de bits erróneos en el bitstream justo antes de decodificar. Para generar errores tendremos que activar la casilla “*Simulate Channel Errors*” y definir la probabilidad de error que queremos simular. El decodificador tratará de reconstruir la imagen con errores en el bitstream que serán más o menos severos en función de la información a la que afecten, pudiendo incluso llegar a interrumpir la decodificación.

Ejercicio: Con los parámetros que vienen por defecto (*Quality:75, Qmatrix:Standard, Huffman:Standard VLC, Smooth Output: NO, Restart Markers: NO, Simulated Channel Errors: NO*), realiza varias compresiones JPEG de la imagen Lena256B con los siguientes valores de bitrate: 1.5, 1.2, 1.0, 0.8, 0.6, 0.4 y 0,2 bpp. Observa la degradación de la calidad de la imagen descomprimida cuando aumentamos la tasa de compresión. En una tabla, indica los valores de PSNR y tamaños de las imágenes comprimidas para cada uno de los bitrates analizados.

Ejercicio: Usando la matriz de cuantización Flat, realiza varias compresiones JPEG de la imagen Lena256B con los siguientes valores de bitrate: 1.5, 1.2, 1.0, 0.8, 0.6, 0.4 y 0,2 bpp. Añade a la tabla anterior los nuevos valores, indica los valores de PSNR y tamaños de las imágenes comprimidas para cada uno de los bitrates analizados. ¿Dibuja un gráfico con las curvas R/D. Son iguales? ¿Observas diferencias en las imágenes comprimidas con distintas matrices de cuantización? Utiliza la imagen Birds y guarda en disco dos bmp obtenidos para 0,4 bpp, uno con la matriz estándar y otro con la Flat. Carga ambas imágenes en el visor de imágenes de windows, alterna entre una y otra y determina cuál se ve mejor. ¿El PSNR marca mejor la de Flat, no? ¿Hay un error en el VcDemo? Explicalo.

Ejercicio: Fijando la tasa de compresión a un factor de calidad de 75, vamos a activar la simulación de errores de transmisión.

- Para ello realizaremos tres compresiones para cada probabilidad de error y una compresión sin errores. Construye una tabla con los resultados estadísticos (valor medio, mínimo, máximo, desviación, etc.)
- Para una misma probabilidad de error, ¿por qué el resultado es diferente?. Justifica la respuesta.
- Para una tasa de error de 0.0005, realiza varias compresiones con los diferentes opciones de marcadores, realizando un estudio similar al apartado (a), incluyendo también la tasa de compresión resultante (*Encoded Bitrate*). Analiza la efectividad del uso de marcadores.

JPEG 2000

JPEG2000 es un estándar de compresión basado en el uso de la transformada Wavelet, aprobado por ISO en el año 2000, que ha sido propuesto para reemplazar al estándar anterior ya que mejora sensiblemente el rendimiento del anterior e incluye muchas funcionalidades adicionales (escalabilidad, región de interés, etc.).

JPEG2000 permite realizar compresión con pérdidas o sin pérdidas (*Lossless*). Esto es útil

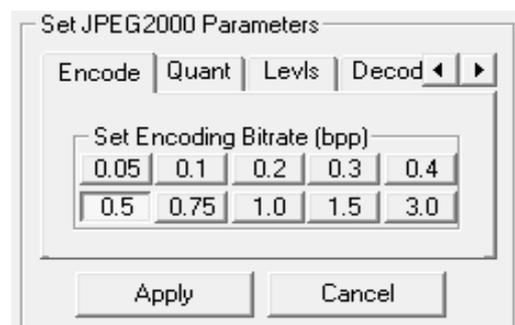


Figura 6. Parámetros de JPEG2000

para algunas aplicaciones como podrían ser las imágenes médicas, en las cuales las pérdidas de información pueden concluir con diagnósticos equivocados.

En VcDemo, accedemos a la herramienta de compresión JPEG2000 seleccionando la entrada “*Linear Transforms* → *Wavelet Coding: JPEG2000*” a través del menú contextual de la imagen.

Los parámetros de configuración de JPEG2000 se muestran en la Figura 6, en donde se encuentran las siguientes pestañas:

- ✓ “**Encode**” permite definir la tasa de compresión que queremos alcanzar.
- ✓ “**Quant**” especifica la forma en que se va a comprimir: (a) *Lossless quantization* comprime la imagen sin pérdidas (no se aplica control de bitrate), (b) *SNR Layers* comprime la imagen almacenando en primer lugar la información más relevante para obtener una buena calidad. De esta manera, si se trunca el bitstream podríamos reconstruir la imagen con la mejor calidad posible (codificación escalable en calidad), (c) *Code Block* indica el tamaño de bloque que se utilizará para codificar la imagen (similar a JPEG pero en el dominio transformado Wavelet)
- ✓ “**Levls**” define el número de niveles de descomposición Wavelet que se aplican en el proceso de compresión. La casilla “*perfect reconstruction filterbank*” permite que el proceso de transformada (directa e inversa) no introduzca errores de redondeo.
- ✓ “**Decode**” permite seleccionar la tasa de bits deseada que utilizará el decodificador (truncado de bitstream), que no ha de ser mayor que la tasa de bits que se utilizó al comprimir la imagen.
- ✓ “**VisMask**” es una función que modifica el proceso de cuantización de los coeficientes Wavelet aplicándoles una cuantización particular en función de su impacto en la calidad perceptual de la imagen reconstruida.
- ✓ “**Tiling**” permite dividir la imagen en zonas rectangulares que se codifican como si fueran imágenes independientes (sub-imágenes). De especial interés para contenidos de muy alta resolución (satélite, GIS, etc.)

Ejercicio: Con los parámetros que vienen por defecto, realiza una comparación entre los compresores de imagen JPEG y JPEG2000 utilizando como imágenes de test: Lena256B, Birds y Cameraman256B. Para cada imagen construye una gráfica con las curvas R/D (bitrate/PSNR) de ambos compresores usando los bitrates 0.1, 0.2, 0.3, 0.4, 0.5, 1.0, 1.5. Analiza los resultados.

Técnicas de compresión de vídeo en VcDemo.

La herramienta VcDemo además de trabajar con imágenes permite trabajar también con secuencias de vídeo. En éste apartado, se mencionarán y describirán algunos de los métodos implementados en VcDemo.

ME (Estimación de Movimiento).

El primer módulo versa sobre la estimación del movimiento. La estimación de movimiento es una técnica que busca para cada bloque (macrobloque) de un cuadro de vídeo el movimiento que ha sufrido respecto a otro cuadro de vídeo que utilizamos como referencia. El objetivo es sólo determinar el movimiento (vectores de movimiento) con los que posteriormente se pueden codificar las diferencias en lugar del bloque entero.

Para comprender mejor como funciona esta técnica utilizaremos el modulo ME (*Motion Estimation*) con la secuencia de vídeo “*Vectra21frames.yuv*” compuesta por 21 imágenes (fotogramas o cuadros) en formato CIF (352x288). Para cargar la secuencia de vídeo usamos la entrada de menú “*File → Open Sequence*” y seleccionamos la secuencia anteriormente citada. Tras esto, en el menú contextual de la secuencia de vídeo seleccionamos la entrada “*Video Compression → Motion Estimation*”, abriéndose la ventana con los parámetros de configuración del módulo ME.

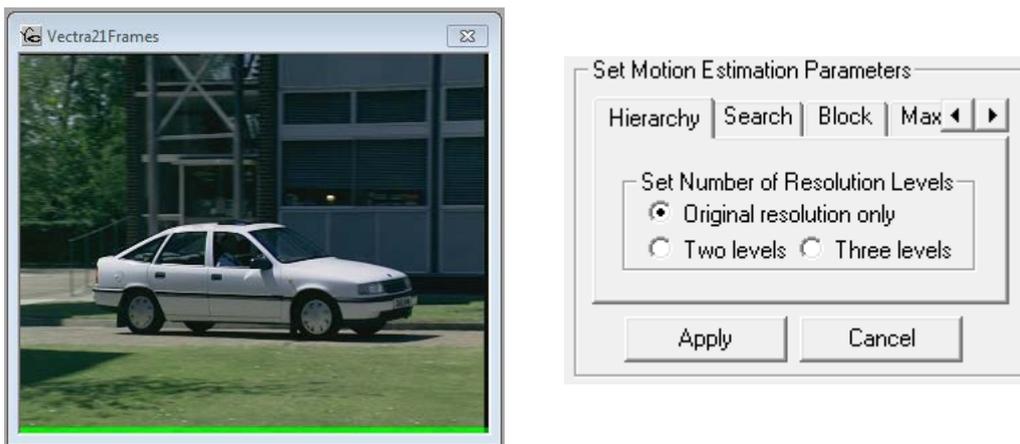


Figura 7. Primer cuadro de vídeo de Vectra21frames.yuv (izquierda) y parámetros de configuración de ME (derecha)

Las diferentes pestañas de la ventana de configuración son:

- ✓ “**Hierarchy**”, en donde se indica si se hace un búsqueda jerárquica (de dos o tres niveles) o no. Ver [Hierarchical Estimation of the motion vector field](#)
- ✓ “**Search**” permite seleccionar diferentes algoritmos de búsqueda de macrobloques: (a) “*Full Search*” realiza la búsqueda del macrobloque a codificar en todas las posiciones del área de búsqueda definida en el cuadro de referencia (alto coste computacional), (b) “*One/Time*” y (c) “*N-Step*” son algoritmos de búsqueda más rápidos que permiten estimar el desplazamiento de los macrobloques aunque no sean capaces de realizar la búsqueda de forma óptima como “*Full Search*”, perdiendo cierta precisión.

- ✓ “**Block**” especifica el tamaño del macrobloque. Por defecto es 8x8, aunque en la mayoría de estándares de compresión de vídeo es de 16x16.
- ✓ “**Max.Displ.**” determina la extensión del área de búsqueda alrededor de la posición del macrobloque a codificar (se mide en píxeles). Por ejemplo, si la posición del macrobloque (8x8) en el cuadro (esquina superior izquierda) es (n,m) y el desplazamiento es 7, el área de búsqueda sería la representada en la figura. La esquina superior izquierda del área de búsqueda estaría ubicada en la posición (n-7, m-7) del cuadro de referencia.

Área de búsqueda



Macrobloque
- ✓ “**N-Step**” se activa cuando seleccionamos el algoritmo de búsqueda “*N-Step*” y determina el número de pasos de dicho algoritmo.
- ✓ “**Video**” se utiliza para especificar cómo queremos ver la evolución de la estimación de movimiento de la secuencia de vídeo. O bien lo queremos ver cuadro a cuadro (*Frame-by-Frame*), en cuyo caso procesa un cuadro de vídeo y espera indicación del usuario para continuar, o bien realiza la estimación de toda la secuencia y cuando termina lo muestra un número de veces más (*Loops*)

Para ver el proceso, con los valores por defecto de los parámetros de configuración lanzamos la estimación de movimiento (botón *Apply*). Para cada cuadro de vídeo se observan cuatro ventanas:

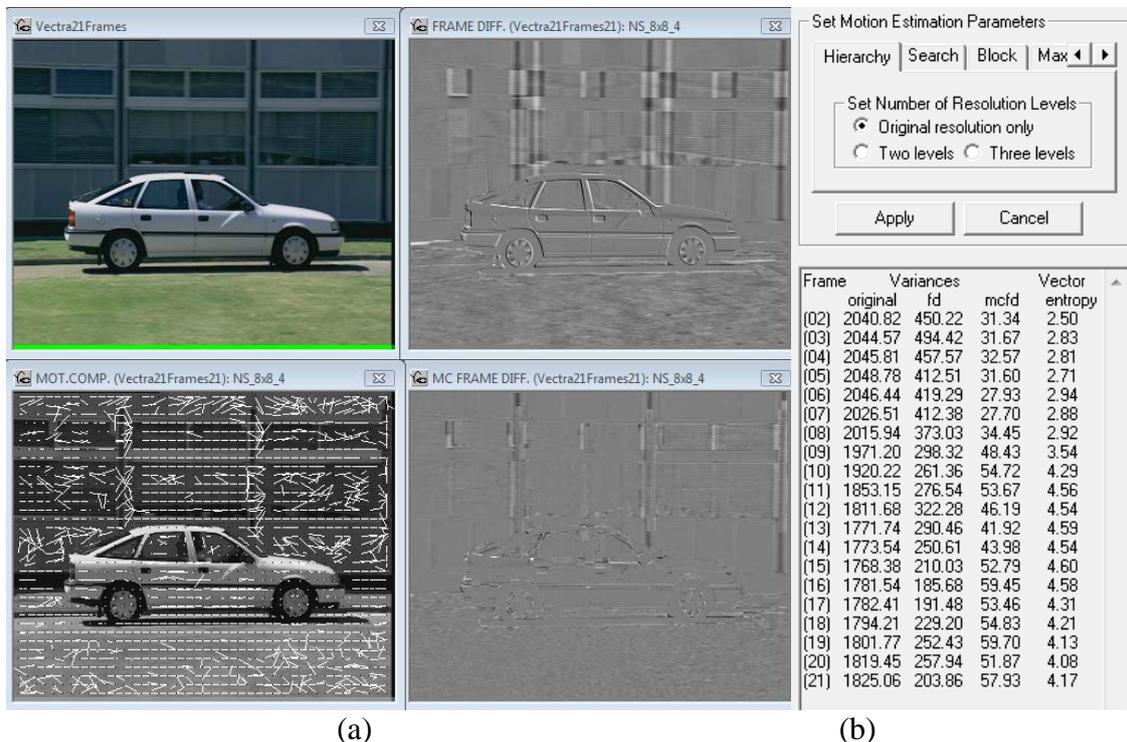


Figura 8. (a) Ventanas que muestran la evolución de la estimación de movimiento. (b) Ventana de configuración con las estadísticas de la estimación de movimiento.

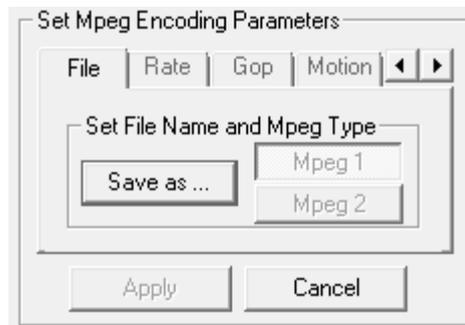
- (a) El cuadro de vídeo a procesar (arriba izquierda).
- (b) La diferencia entre este cuadro y el de referencia (arriba derecha), que en realidad muestra los cambios producidos entre ambos cuadros de vídeo.
- (c) El cuadro resultante de la compensación de movimiento (cuadro de referencia + movimiento) con los vectores de movimiento sobreimpresionados.
- (d) La diferencia entre el cuadro (c) y el cuadro actual, representa el error que cometo tras la estimación y compensación de movimiento.

Una vez ha terminado de realizar la estimación de movimiento de toda la secuencia, en la ventana de resultados (Figura 8(b)) se suministran unas estadísticas para cada cuadro de vídeo. La primera columna identifica el número de cuadro, las tres siguientes columnas muestran los valores de varianza de (1) el cuadro original, (2) el cuadro “*fd*” (*frame difference*), diferencia entre el cuadro actual y el de referencia y (3) el cuadro “*mcf*” (*motion compensated frame difference*), diferencia entre el cuadro actual y la versión resultante del proceso de estimación y compensación de movimiento. Estos valores de varianza sirven como indicadores de la cantidad de información que contienen los cuadros. La última columna, “*Vector Entropy*” muestra una estimación de la entropía de los vectores de movimiento calculados en bpv (bits per vector).

Ejercicio: Con los parámetros del módulo ME que vienen por defecto y la secuencia de vídeo *Vectra21frames.yuv*, recoge los datos de la ventana de resultados de la estimación de movimiento para cada uno de los tres algoritmos de búsqueda (*Full*, *One/Time* y *N-Step*). Coloca los resultados en una tabla y analiza el rendimiento de los tres algoritmos de búsqueda.

MEnc (Mpeg 1/2 Encoder)

Este módulo contiene los compresores de vídeo MPEG 1 y MPEG 2. Son versiones no optimizadas del estándar que generan un bitstream compatible MPEG que se guarda en un fichero. Por eso, para trabajar con este módulo, lo primero que tenemos que especificar es el nombre y la ubicación de un archivo en el que se almacenará el resultado de la compresión.



Para activar este módulo, desde el menú contextual de la secuencia de vídeo cargada (p.e.

Vectra21frames.yuv), seleccionamos la entrada “*Video Compression → Mpeg Video Encoder*” para abrir la ventana del módulo. Se disponen de las siguientes pestañas para su configuración:

- ✓ “**File**”. Esta pestaña es la primera que debemos activar, seleccionando el nombre y ubicación del archivo en donde se va a almacenar el video comprimido. También tenemos que indicar si vamos a utilizar el compresor *Mpeg1* o el *Mpeg2*.
- ✓ “**Rate**” indica la tasa de bits (*bitrate*) objetivo que el compresor tiene que suministrar.

- ✓ **“Gop”** indica cual es la estructura del GOP (Group Of Pictures) que va a utilizarse para la compresión, mostrando el patrón de tipos de cuadros a codificar. Se dispone de seis estructuras de GOP:

(a) **“I Only”**, cada cuadro se codifica como un cuadro de tipo I (INTRA), por lo que no se utiliza la estimación/compensación de movimiento,

(b) **“IBB”**, la secuencia de video se organiza en GOPs de tres cuadros, donde el primero es de tipo I (INTRA), y los dos siguientes de tipo B (*Bidirectional*),

(c) **“I11(P)”** define GOPs con un primer cuadro de tipo I seguido de 11 cuadros de tipo P (*Predictive*).

El resto de patrones siguen la misma sintaxis en su definición.

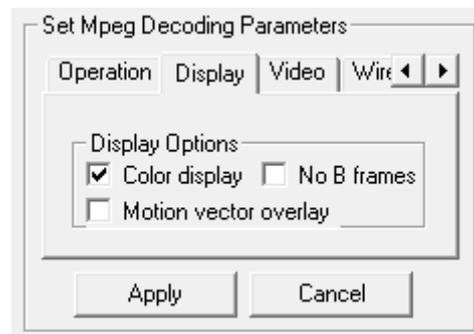
- ✓ **“Motion”** especifica el máximo desplazamiento entre dos cuadros consecutivos. Es un indicador de la cantidad de movimiento que contiene la secuencia de video, y se utiliza para delimitar las áreas de búsqueda en la estimación de movimiento.
- ✓ **“Format”** (sólo para Mpeg2) especifica cómo se trata la fuente de vídeo. Si es progresivo (*Progressive*) o entrelazada (*Interlaced*).
- ✓ **“Field/Frame”** (sólo para Mpeg2 y fuente de vídeo entrelazada) especifica cómo se va a realizar la compresión:

(a) codificando cada campo por separado (*Field Coding*), o bien

(b) codificando el cuadro completo (*Frame Coding*).

MDec (Mpeg 1/2 Decoder).

El módulo MDec permite realizar la decodificación de los archivos comprimidos de tipo MPEG (generados por MEnc). Como con el resto de módulos, a continuación describimos sus parámetros de configuración:



- ✓ **“Operation”**. El decodificador produce tres salidas diferentes:
 - (a) **“Decoded Frames”** el video descomprimido como lo haría cualquier otro decodificador,
 - (b) **“Frame Prediction”** muestra los cuadros de vídeo resultado de la predicción (sin añadir el error residual de los macrobloques). Este modo permite observar la calidad de la predicción en sí,
 - (c) **“Coded Difference”**, muestra únicamente el error residual encontrado para cada macrobloque, y los macrobloques que han sido codificados en modo INTRA.
- ✓ **“Display”** indica diferentes formas de mostrar el video decodificado:
 - (a) **“Color display”** en color o escala de grises (depende del video original),
 - (b) **“Motion vector overlay”** muestra los vectores de movimiento sobreimpresionados,

(c) “*No B frames*” evita decodificar los cuadros de tipo B, para acelerar la decodificación en aquellas máquinas con poca capacidad de cómputo.

- ✓ “**Video**” indica a qué ritmo se va a decodificar el vídeo, o bien se decodifica cuadro a cuadro, o se decodifican todos los cuadros sin interrupción. También se puede indicar que se repita el proceso varias veces (“*Loops*”).
- ✓ “**Wireless Channel**”. Este módulo incluye un simulador de canal inalámbrico que permite determinar en qué partes del *bitstream* se producen los errores de bit (o paquete) tras la transmisión del vídeo comprimido por un canal inalámbrico. El simulador inyecta los errores en el *bistream* antes de la decodificación. Dependiendo del patrón de error producido, y dada la escasa robustez del decodificador MPEG, es posible que éste no pueda terminar. Para más información acerca del simulador se recomienda acceder a la ayuda (entrada de menú “*Ayuda → VcDemo HElp (F1)*”).
- ✓ “**Save**” permite definir el nombre de archivo en donde se guardará el resultado de la decodificación (el video reconstruido).

Ejercicio: Con los parámetros que vienen por defecto del módulo **MEnc**, realizamos la compresión de la secuencia de vídeo *Vectra2Iframes.yuv* con las dos versiones de MPEG (Mpeg1 y Mpeg2). Guarda las compresiones en ficheros con nombres diferentes.

- (a) Anota la siguiente información del último cuadro de vídeo comprimido en ambas compresiones: “*Picture Type*”, “*Bit rate*”, “*PSNR Y*”. Analiza los resultados.

Ejercicio: Cierra la ventana del módulo **MEnc** (botón Cancel) y la ventana de la secuencia de video. A continuación, carga la secuencia de vídeo que has comprimido con *Mpeg1* (entrada de menú “*File → Open Mpeg stream*”). El compresor ha guardado las secuencias comprimidas en la carpeta “*UserOutput*” del directorio en donde se instaló VcDemo. Una vez cargada la secuencia comprimida, activa el módulo **MDec** para realizar las siguientes operaciones:

- (a) Decodifica la secuencia usando los tres modos de operación. Observa el resultado y analiza la calidad de la predicción de movimiento. Para analizar mejor la decodificación, configura el decodificador para que se realice cuadro a cuadro (pestaña *Video*, casilla *frame-by-frame*)
- (b) Establece el modo de operación en “*Decoded Frames*” y activa el simulador de errores de transmisión sobre un enlace inalámbrico para esta secuencia comprimida. Utilizamos los parámetros por defecto del simulador: “*HiperLAN 20 MBps*”, “*Random Bit Errors*”, “*P(error) = 1e-005*” y “*Number of HiperLAN users = 1*”. Pulsamos el botón “*Apply*” para realizar una simulación de envío en el canal inalámbrico previamente configurado. A continuación decodificamos el bitstream recibido con el botón “*Decode Stream*”. Para visualizar el bitstream decodificado tenemos que pulsar el botón “*Apply*” de la ventana del decodificador.
 - i. Anota la información que genera el simulador e interprétala.
 - ii. Aumenta el número de usuarios en la red para crear tráfico de fondo con los siguientes valores: 5, 10, 15, 20, 25. Anota la información del paso anterior para cada simulación e interpreta los resultados.

Bibliografía adicional

- [1] *Information and Communication Theory Group* de la Delft University of Technology Available: <http://siplab.tudelft.nl/content/history>
- [2] Sayood, Khalid. “*Introduction to data compression*” Ed. Elsevier.
- [3] V. G. Ruiz, “*Compresión Reversible y Transmisión de Imágenes*”. Almería, España: Universidad de Almería, 2000.
- [4] A.N. Skodras, C.A. Christopoulos and T. Ebrahimi, “*JPEG2000: The Upcoming Still Image Compression Standard*”
- [5] Ignacio Bañó “Wavelets: Un método de compresión de imágenes”
- [6] Rafael Molina “Tema 6: Codificación basada en transformaciones. Aplicaciones e imágenes”, Documentación Escuela Técnica Superior de Ingeniería Informática Granada.

Informe

El informe o memoria de esta sesión de laboratorio deberá incluir los siguientes aspectos:

- Una página de presentación con el nombre de la asignatura, curso, titulación, tu nombre y apellidos, número y título de la práctica y las fechas de realización y envío la memoria.
- Un resumen de los objetivos perseguidos en la práctica.
- Metodología: Una breve descripción del proceso que has seguido a la hora de definir los distintos escenarios, modelos, etc. contemplados en la práctica.
- Exposición y análisis de los resultados obtenidos a lo largo de la práctica y, si procede, una comparación de los mismos con lo que esperabas obtener.
- La respuesta a las cuestiones o ejercicios planteados en la memoria de la práctica. Si la respuesta incorpora nuevas gráficas, debe realizarse un breve análisis de las mismas.
- Una conclusión general en la que se describa lo que se ha aprendido con esta práctica, las dificultades que se han encontrado en la realización de los diferentes pasos y cualquier sugerencia, propuesta de ampliación o comentario que permita mejorar la práctica propuesta.

Book - Understanding Compression:

Understanding Compression - Data compression for modern developers - ColtMcAnlis and Aleks Haecky - Oreilly

Enlaces

Dithering

See <https://en.wikipedia.org/wiki/Dither>

YUV

See <https://es.wikipedia.org/wiki/YUV>

YcbCr

See <https://es.wikipedia.org/wiki/YCbCr>

DPCM

See <https://es.wikipedia.org/wiki/DPCM>

Arithmetic Coding

See [Basic arithmetic coding by Arturo Campos](#)

Variable Length Codes

See https://en.wikipedia.org/wiki/Variable-length_code

Codificación Huffman

See https://es.wikipedia.org/wiki/Codificaci%C3%B3n_Huffman

See <https://www.quora.com/What-is-an-intuitive-explanation-of-Huffman-coding>

Codificación Entrópica

See

[https://es.wikipedia.org/wiki/Codificaci%C3%B3n_entr%C3%B3pica#RLE:_Codificaci%C3%B3n_por_longitud_de_series_\(Run_Length_Encoding\)](https://es.wikipedia.org/wiki/Codificaci%C3%B3n_entr%C3%B3pica#RLE:_Codificaci%C3%B3n_por_longitud_de_series_(Run_Length_Encoding))

Run Length Encoding (RLE)

See https://es.wikipedia.org/wiki/Run-length_encoding

Codificación Huffman

https://es.wikipedia.org/wiki/Codificaci%C3%B3n_Huffman

Motion Estimation

See https://en.wikipedia.org/wiki/Motion_estimation

Block Matching Algorithm

See https://en.wikipedia.org/wiki/Block-matching_algorithm