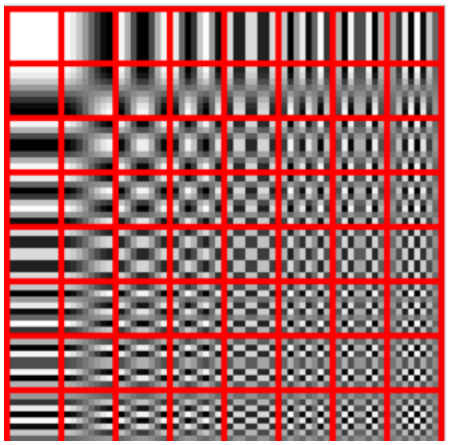
Servicios Multimedia
Grado de Ingeniería en Tecnología de Telecomunicación – 4º curso

Trabajo Laboratorio: Sistemas Multimedia- Imagen III - (codec JPEG)

En esta práctica vamos a implementar el códec estándar empleado por la mayoría de softwares de tratamiento de imágenes, el formato JFIF, cuando creamos archivos JPEG.

Como hemos visto en clase, este proceso consiste en la realización de un conjunto de pasos, del que destacamos el tratamiento de la transformada discreta del coseno.

En primer lugar, realizaremos el cálculo de la matriz que representa el espectro en dos dimensiones que utilizaremos para la conversión de la imagen original al dominio de la frecuencia (equivalente al cálculo de cosenos que vimos en el laboratorio de la semana pasada, y que en este caso denominamos “banderas”, y que se corresponde con la imagen:



A continuación, partiendo de una imagen concreta (fotografía), realizaremos todo el proceso de codificación y decodificación, siguiendo los pasos realizados en clase.

Para ello, realiza las siguientes tareas:

1. En java, implementa el código donde calcules y visualices una muestra del conjunto de 64 “banderas” que utilizaremos para calcular la 2D-DCT.

2. En java, introduce los valores de la imagen (original) que aparece en el documento: https://en.wikipedia.org/wiki/JPEG, y visualiza los valores de la matriz resultante para comprobar que coinciden con los valores originales.

3. En java, elimina de la imagen original la componente dc, y visualiza la matriz resultante.

4. En java, calcula la transformada 2D-DCT de la matriz anterior y visualízala.

5. En java, introduce y visualiza la Tabla de Cuantización para el caso propuesto, de una calidad del 50%.

6. En java, realiza la etapa de Cuantización, mostrando los valores de la matriz resultante, que será

la matriz que se envía (imagen comprimida).

7. En java, realiza la etapa de Recuperación, que comenzará con la fase de Descuantización. Visualiza la matriz resultado de esta etapa.

8. En java, calcula la inversa 2D-DCT y visualiza sus valores con redondeo.

9. En java, calcula y visualiza la matriz error.

10. En java, calcula y visualiza el error promedio.

11. Crea una imagen en java donde dada su simetría puedas predecir las componentes espectrales que la caractericen. Realiza todo el proceso para otra imagen original creada por tu cuenta, y comenta el resultado obtenido.

Tarea 1: En java, implementa el código donde calcules y visualices una muestra del conjunto de 64 "banderas" que utilizaremos para calcular la 2D-DCT.

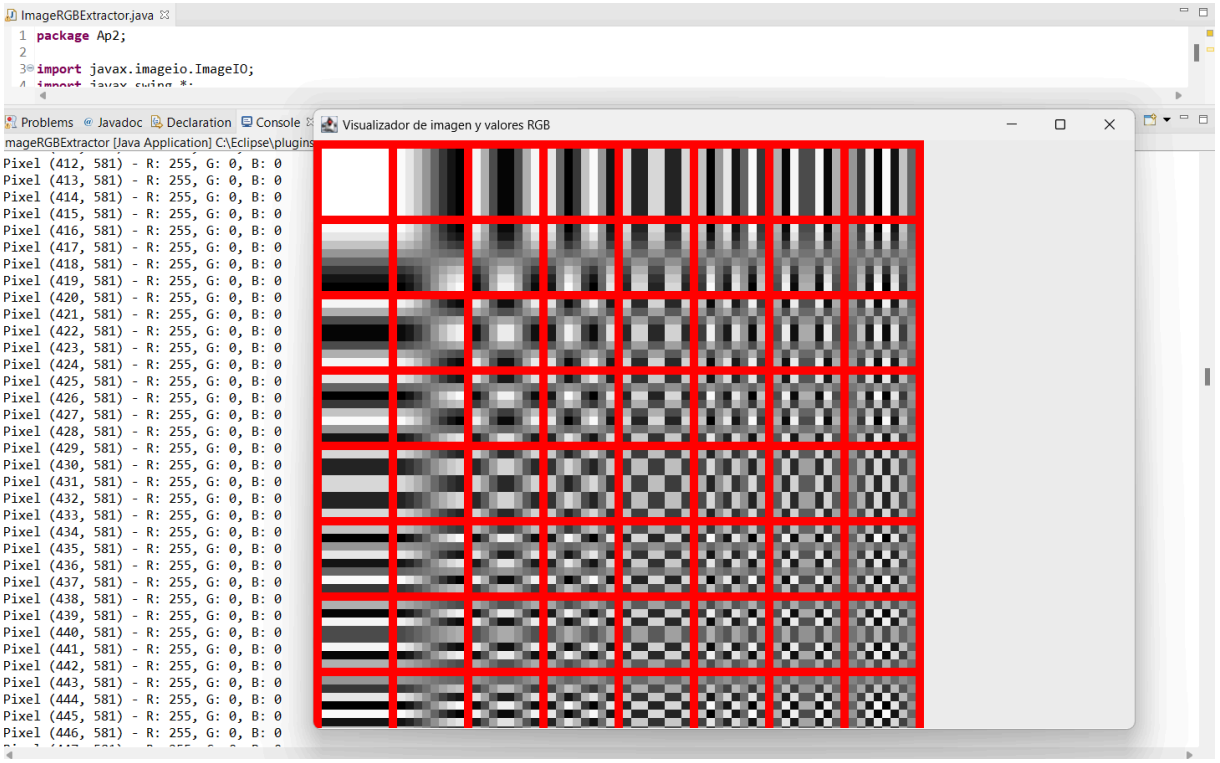
Screenshot 1:

Breve descripción:

En este apartado implementamos la Transformada Discreta del Coseno en 2D (2D-DCT), que es clave para la compresión de imágenes JPEG. Generamos y visualizamos las 64 bases DCT ("banderas") que descomponen un bloque de imagen de 8x8 en diferentes frecuencias. Mostramos estas bases gráficamente y también imprimimos sus valores numéricos en la consola. Este proceso permite comprimir imágenes eliminando información redundante, manteniendo la calidad visual.

Tarea 2: En java, introduce los valores de la imagen (original) que aparece en el documento: https://en.wikipedia.org/wiki/JPEG, y visualiza los valores de la matriz resultante para comprobar que coinciden con los valores originales.

Screenshot 2:

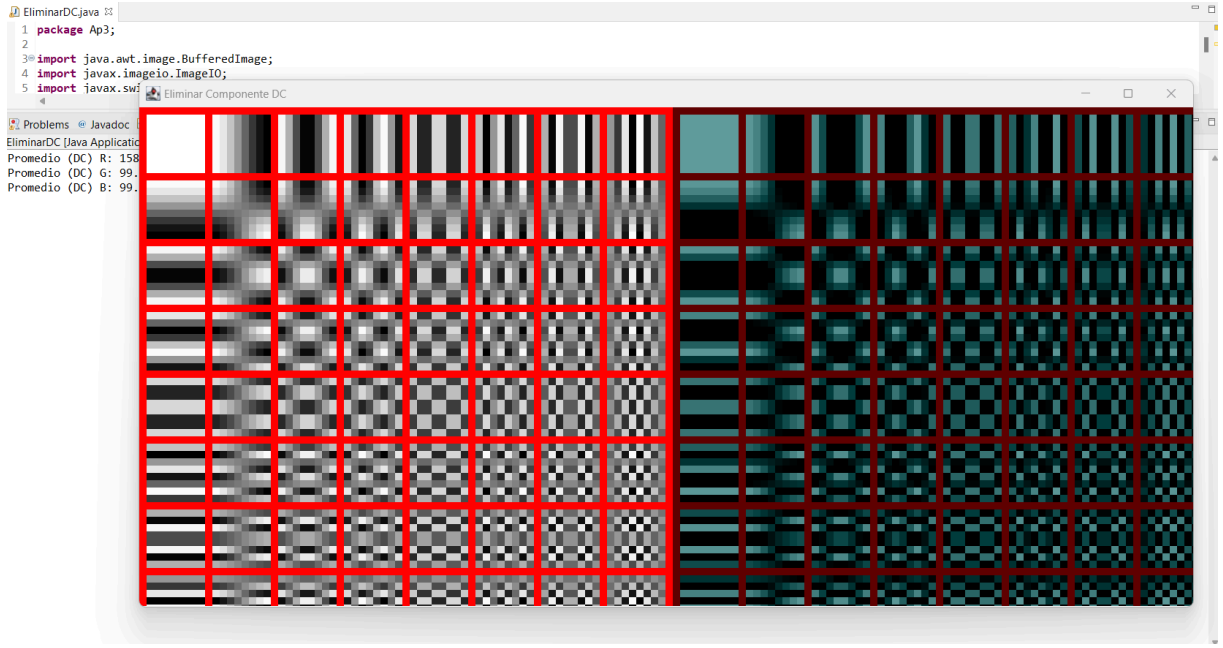


Breve descripción:

Este apartado describe un proceso en el que se extraen los valores RGB de una imagen JPEG y se visualizan mediante una simulación gráfica. El objetivo es verificar que los valores obtenidos coinciden con los originales, mostrando tanto los datos extraídos como una representación visual de la imagen. Al final, se obtiene una recreación de la imagen basada en los valores de color de sus píxeles, confirmando la precisión de los datos extraídos.

Tarea 3: En java, elimina de la imagen original la componente dc, y visualiza la matriz resultante.

Screenshot 3:

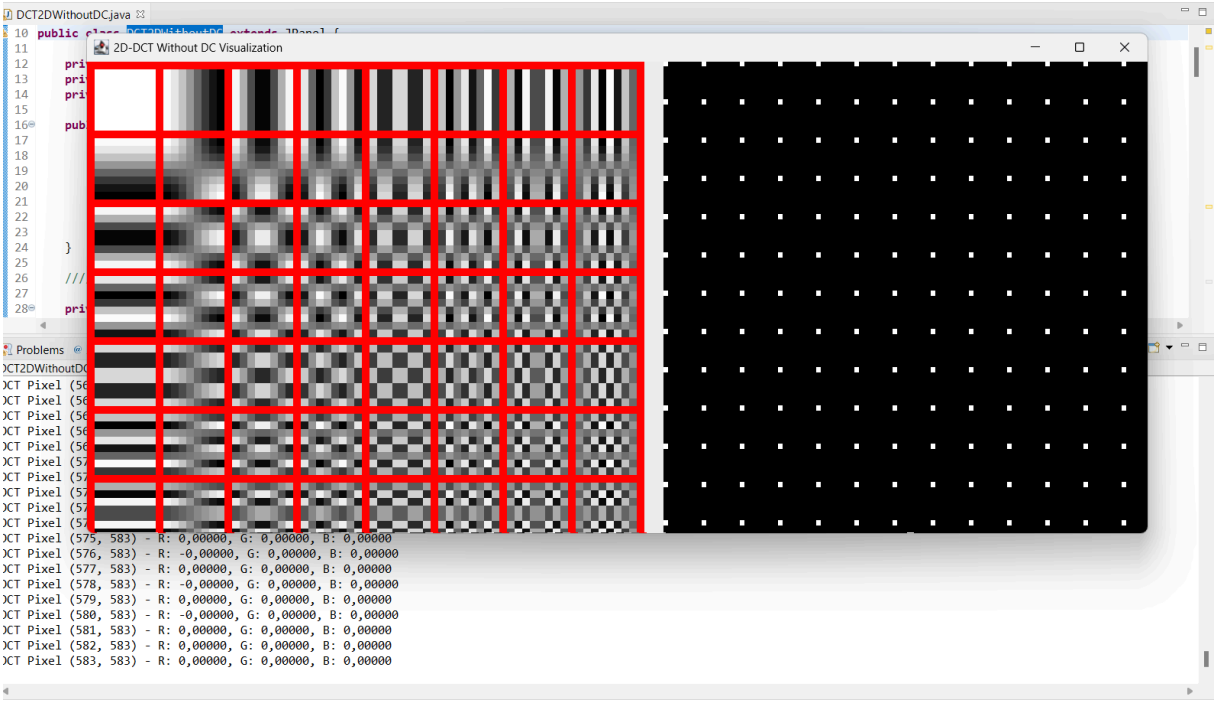


Breve descripción:

En este apartado se busca eliminar la componente DC de una imagen, que representa la información de brillo promedio. Al eliminar esta componente, se pretende modificar la imagen, removiendo el brillo general de los colores. Como resultado, la imagen debería perder su intensidad lumínica, mostrando tonos más apagados o neutros en comparación con la imagen original, lo que afecta la percepción visual de los colores sin alterar su estructura general.

Tarea 4: En java, calcula la transformada 2D-DCT de la matriz anterior y visualízala.

Screenshot 4:

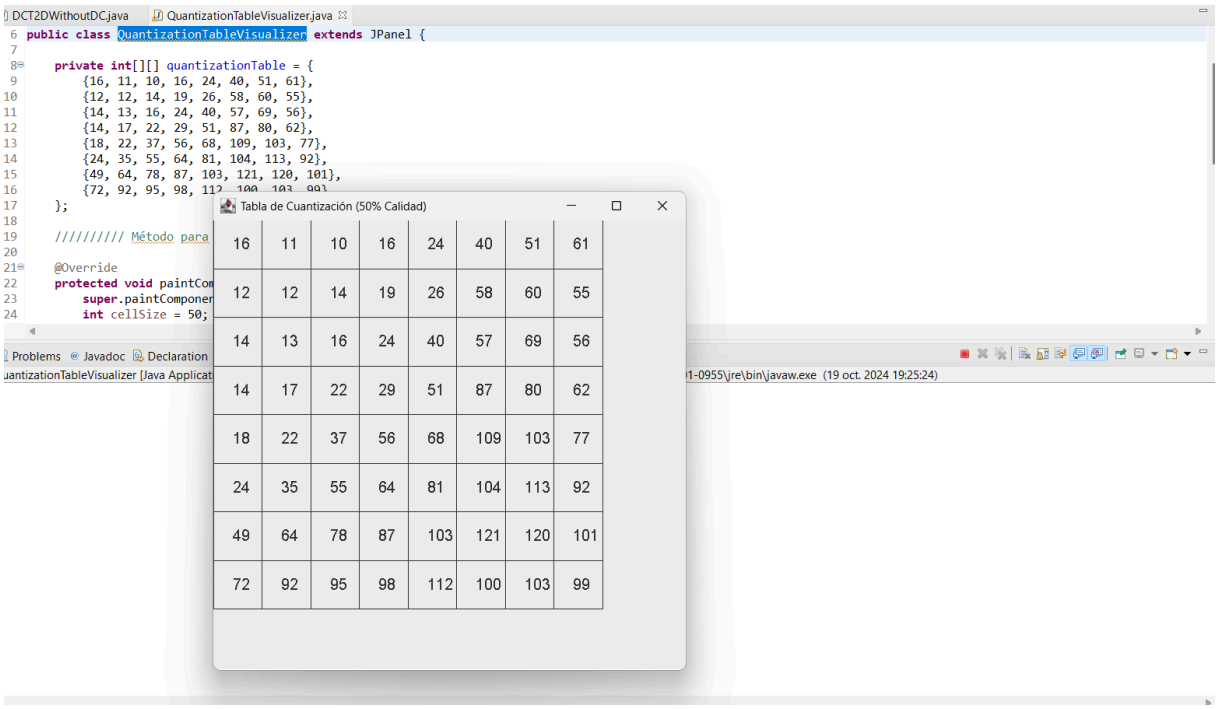


Breve descripción:

En este apartado se aplica la Transformada Discreta del Coseno (DCT) a una imagen para descomponerla en sus componentes de frecuencia. La imagen original, que muestra las bases DCT, resulta en coeficientes mayormente pequeños, lo cual es esperado, ya que la imagen está compuesta por patrones simples de frecuencia. El resultado confirma que el proceso de la DCT funciona adecuadamente, reflejando la simplicidad de la imagen.

Tarea 5: En java, introduce y visualiza la Tabla de Cuantización para el caso propuesto, de una calidad del 50%.

Screenshot 5:

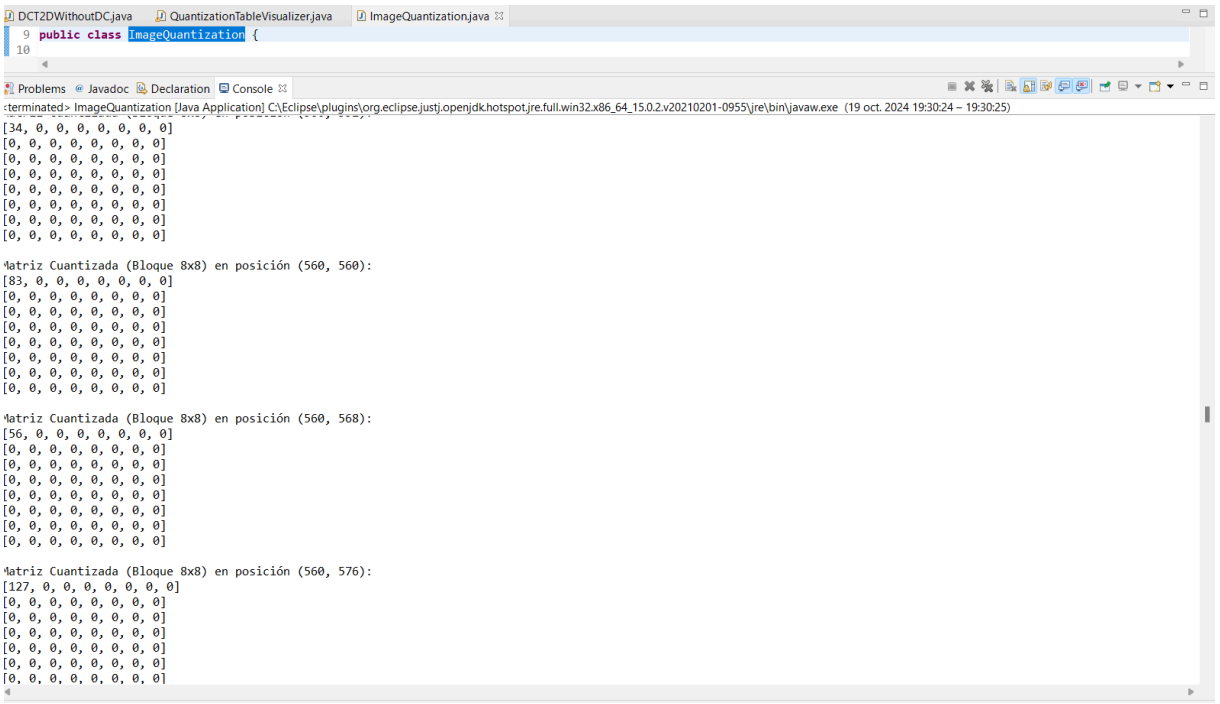


Breve descripción:

En este apartado se introduce y visualiza la Tabla de Cuantización utilizada en el proceso de compresión JPEG para una calidad del 50%. Esta tabla sirve para reducir los coeficientes obtenidos en la Transformada Discreta del Coseno (DCT), priorizando las frecuencias bajas y eliminando detalles finos menos perceptibles por el ojo humano. El resultado es una representación visual de una tabla 8x8 que muestra los valores de cuantización que se aplicarán a los coeficientes DCT, equilibrando la calidad y la compresión de la imagen.

Tarea 6: En java, realiza la etapa de Cuantización, mostrando los valores de la matriz resultante, que será la matriz que se envía (imagen comprimida).

Screenshot 6:

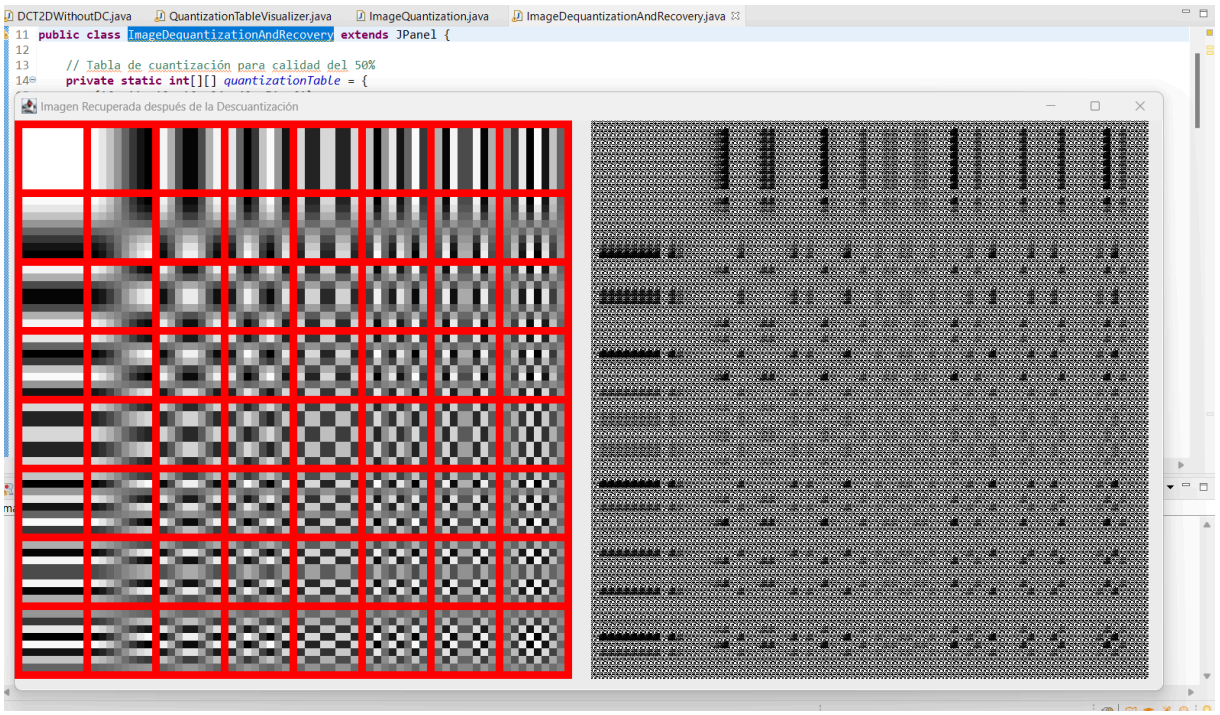


Breve descripción:

En este apartado se realiza la cuantización de la imagen como parte del proceso de compresión JPEG. La imagen se divide en bloques de 8x8, a los que se les aplica la Transformada Discreta del Coseno (DCT) y luego se cuantizan utilizando una tabla estándar con una calidad del 50%. El resultado es una matriz comprimida que reduce los detalles menos importantes, mostrando los valores finales cuantizados de la imagen.

Tarea 7: En java, realiza la etapa de Recuperación, que comenzará con la fase de Descuantización. Visualiza la matriz resultado de esta etapa.

Screenshot 7:

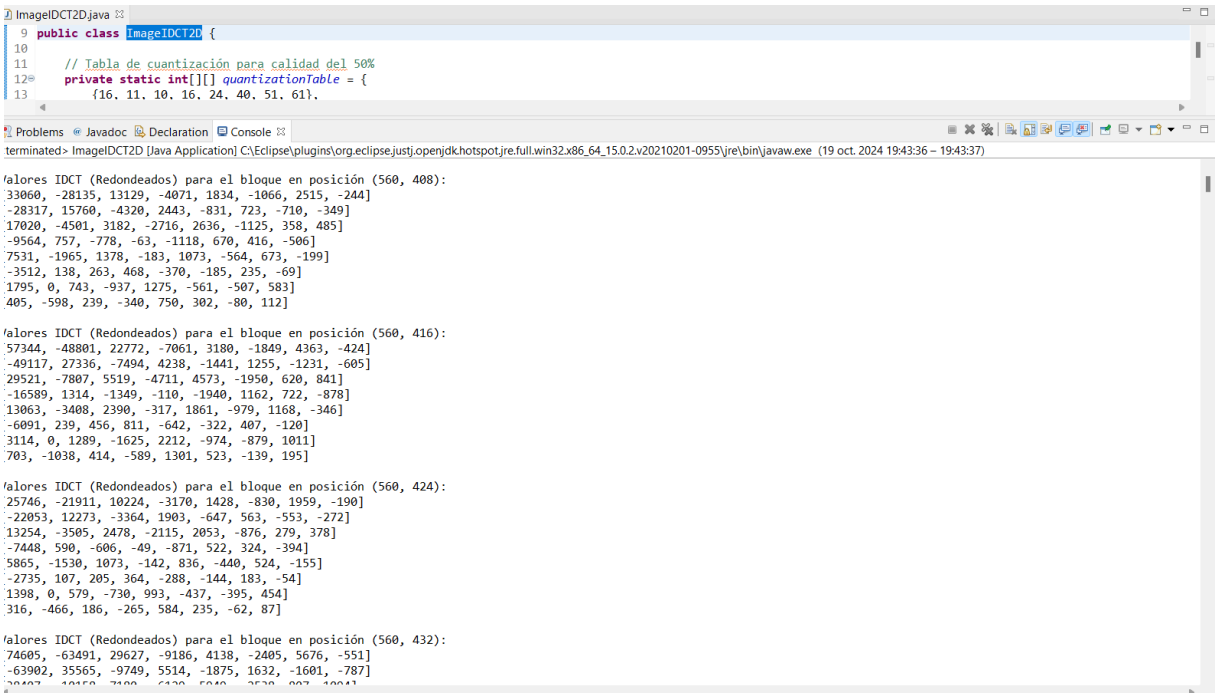


Breve descripción:

En este apartado se lleva a cabo la recuperación de la imagen tras el proceso de cuantización, comenzando con la fase de descuantización. Se multiplican los coeficientes cuantizados por los valores de la tabla de cuantización, y luego se aplica la Transformada Inversa del Coseno Discreto (IDCT) para reconstruir la imagen en el dominio espacial. Como resultado, se visualiza la imagen recuperada junto a la imagen original. Aunque ambas imágenes se parecen, la imagen recuperada presenta una pérdida de detalles y se ve ligeramente más suave debido a la compresión y la reducción de información introducida por la cuantización.

Tarea 8: En java, calcula la inversa 2D-DCT y visualiza sus valores con redondeo.

Screenshot 8:

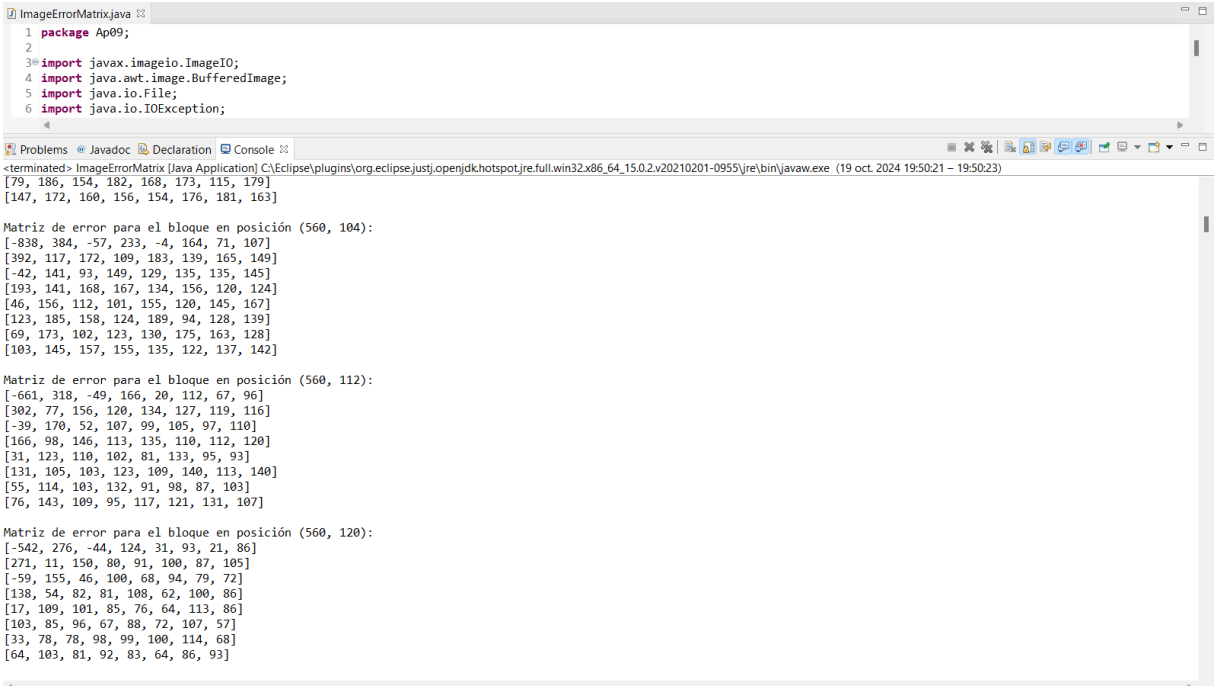


Breve descripción:

En este apartado se realiza la Transformada Inversa del Coseno Discreto en 2D (IDCT), partiendo de los valores cuantizados de la imagen para devolverlos al dominio espacial. Primero se aplica la descuantización multiplicando los coeficientes cuantizados por la tabla de cuantización, y luego se calcula la IDCT para cada bloque de 8x8 píxeles. Finalmente, los valores obtenidos son redondeados y mostrados, representando una aproximación de los valores originales de la imagen antes de la cuantización. El resultado es la matriz de píxeles recuperada tras la compresión y su posterior reconstrucción.

Tarea 9: En java, calcula y visualiza la matriz error.

Screenshot 9:



Breve descripción:

En este apartado se calcula la matriz de error al comparar los valores originales de la imagen con los valores recuperados tras la IDCT. La matriz de error refleja las diferencias entre ambos, mostrando cómo la compresión afecta la calidad de la imagen. El resultado son matrices que indican las diferencias entre la imagen original y la imagen reconstruida tras el proceso de compresión y descompresión.

Tarea 10: En java, calcula y visualiza el error promedio.

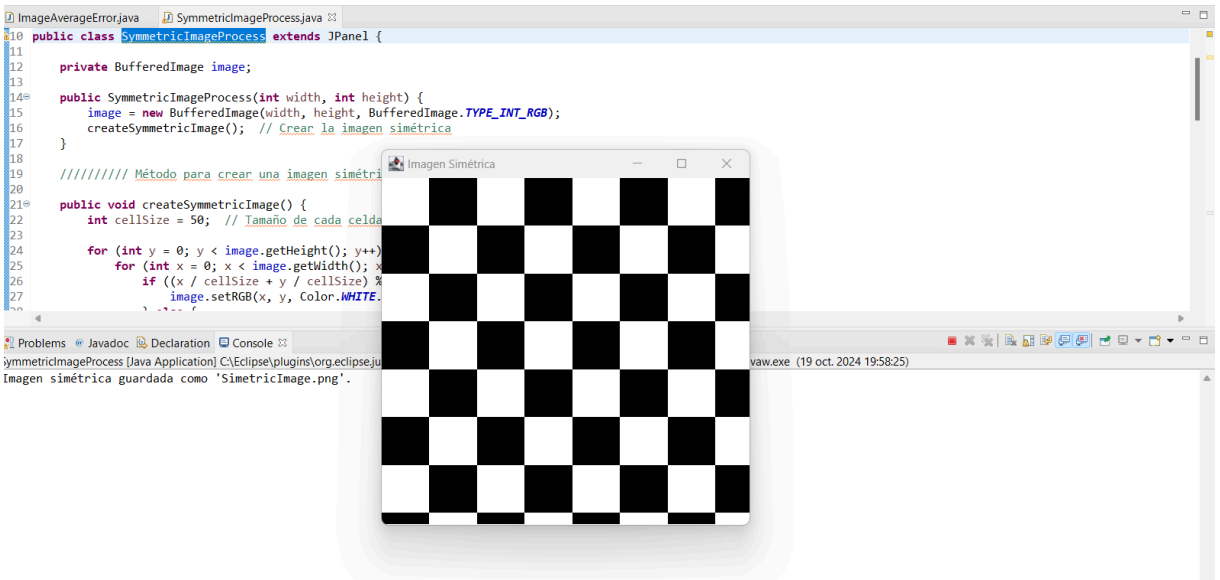
Screenshot 10:

Breve descripción:

En este apartado se calcula el error promedio de la imagen tras el proceso de compresión y recuperación. Para cada bloque de 8x8 píxeles, se cuantiza, descuantiza y aplica la Transformada Inversa del Coseno Discreto (IDCT). Posteriormente, se calcula la matriz de error comparando los valores originales con los recuperados, y se obtiene el error promedio al promediar los valores absolutos de la matriz de error de todos los bloques de la imagen. El resultado es una medida del impacto global de la compresión sobre la calidad de la imagen.

Tarea 11: Crea una imagen en java donde dada su simetría puedas predecir las componentes espectrales que la caractericen. Realiza todo el proceso para otra imagen original creada por tu cuenta, y comenta el resultado obtenido.

Screenshot 11:



Breve descripción:

En este apartado se crea una imagen simétrica con un patrón de ajedrez en Java para predecir sus componentes espectrales al aplicar la DCT. Debido a la simplicidad del patrón, las frecuencias bajas dominaron en la DCT. Tras aplicar todo el proceso de compresión y recuperación, el error promedio es bajo, confirmando que la compresión afecta mínimamente la calidad de la imagen debido a su estructura sencilla.