TRATAMIENTO DIGITAL DE LA SEÑAL
Tratamiento digital de imágenes

Práctica 4

Fecha: 3 de mayo de 2010

1. Introducción

Una imagen digital es una señal de dos dimensiones que se representa por una matriz. Cada elemento de la matriz representa un determinado nivel de intensidad (de gris en esta práctica) o densidad fotográfica, y su posición en la matriz depende de las coordenadas sobre el plano.

El procesado digital de imagen es un campo particular del procesado de señal en el que la matriz que representa la imagen es manipulada para obtener otra. Todas las transformaciones y operadores vistos en la asignatura para señales unidimensionales se pueden extender a las imágenes, trabajando con estos mismos operadores aplicados a dos variables.

La presente práctica pretende consolidar los conocimientos adquiridos sobre tratamiento digital de señales y extenderlos para el procesado en dos dimensiones de imágenes. La teoría asociada a los operadores a usar, está incluida en este enunciado. Así por ejemplo: los conceptos de convolución y DFT extendidas a 2D están resumidos en el apéndice A.

2. Generación y Lectura de Imágenes

1. Además de imágenes generadas por el alumno, en esta práctica se usarán fotografías digitales. Acompañando a este enunciado puede encontrar diferentes fotografías en blanco y negro con 256 niveles de gris y con tamaños 512×512, 256×256, 128×128. Escoja el tamaño de la fotografía en función de la capacidad de la máquina donde vaya a trabajar.

   Para leer los ficheros .tif, puede usar la instrucción `imread` [1] con la sintaxis

   ```
   [F]=imread(´nombrefichero.tif´); F=double(F);
   ```

   donde `F` es la matriz que contiene los niveles de los puntos de la imagen. Tenga en cuenta que `imread` obtiene una matriz cuyos elementos son enteros sin signo (`uint`) y, por lo tanto, para que estos puedan ser modificados libremente, han de convertirse a un formato de doble precición (`double`), tal y como se hizo en las anteriores instrucciones. Para visualizar la imagen asociada a la matriz **F**, debe usar la instrucción

   ```
   image(F); colormap(paleta);
   ```

   donde `paleta` es la paleta de colores usada (gris en nuestro caso). `Paleta` es una matriz con tres columnas y un número de filas igual al de niveles de intensidad. Cada columna representa uno de los colores básicos: rojo, verde y azul, respectivamente. En el caso de una paleta con 256 grises, las tres columnas de `paleta` son idénticas y cada una tiene 256 valores equidistantes en el intervalo [0,1]. Recomendación: puede utilizar las instrucciones `linspace` o `gray` para generar esta paleta[1].

   Pruebe a visualizar alguna de las fotografías digitales de la página web antes indicada.

[1] Si no se especifica lo contrario, todas las imágenes de esta práctica se visualizarán con una paleta de 256 niveles de gris.

2. Haga un programa que genere, visualice imágenes de tamaño $N \times N$ con las siguientes figuras geométricas:

   - Una línea blanca horizontal de un punto de grosor (intensidad 255) sobre fondo negro (intensidad 0).
   - Una línea blanca vertical de un punto de grosor (intensidad 255) sobre fondo negro (intensidad 0).
   - Un rectángulo relleno de color blanco sobre fondo negro, con un alto M y ancho igual al tamaño de la imagen, N.
   - Un rectángulo relleno de color blanco sobre fondo negro, con un ancho M y alto igual al tamaño de la imagen, N.
   - Un rectángulo relleno de color blanco sobre fondo negro, centrado en la imagen, y de lados M y $2M$ puntos.
   - Un rectángulo relleno de color blanco sobre fondo negro, centrado en la imagen, y de lados $2M$ y M puntos.
   - Un círculo relleno de color blanco centrado en la imagen y de radio M puntos. *NOTA:* Recuerde que la ecuación de las coordenadas (x, y) de una circunferencia es: $x^2+y^2 = M^2$.

   Genere los 7 tipos de imágenes geométricas para un tamaño 64×64 y $M = 20$. Visualícelas y guárdelas en ficheros con extensión `.tif` para su posterior uso. Para guardar imágenes en formato .tif, use la instrucción `imwrite` de la siguiente forma:

   ```
   F=uint8(F); imwrite(F,´nombrefichero.tif´);
   ```

   donde se ha hecho una conversión a enteros sin signo (`uint`) de los elementos de la matriz de imagen.

3. Procesado Espacial de Imágenes

Este tipo de procesado esta basado en la manipulación directa de los pixels de una imagen. Este tipo de procesado puede expresarse como

$$\mathbf{G}(x, y) \quad = \quad T(\mathbf{F}(x, y)) \tag{1}$$

donde $\mathbf{F}(x, y)$ y $\mathbf{G}(x, y)$ son las matrices asociadas a la imagen original y la procesada, respectivamente, y $T()$ es un operador definido sobre algún vecindario de puntos alrededor de las coordenadas (x, y). Habitualmente, el vecindario de puntos sobre los que actúa el operador es una región cuadrada centrada en (x, y). En esta sección vamos a ver diferentes tipos de procesado espacial según el número de puntos del vecindario considerado.

3.1. Transformaciones de Intensidad

Este tipo de transformaciones son procesados que varían la intensidad de cada punto individual, usando para ello únicamente el punto en cuestión. En este caso el procesado espacial de (1) puede reducirse a

$$s \quad = \quad T(r) \tag{2}$$

donde s y r son las intensidades de un punto de la imagen después y antes del procesado, respectivamente.

Para analizar las transformaciones de intensidad efectuadas, se puede utilizar el histograma de una imagen. El histograma de una imagen digital con niveles de gris en el rango $[0, L-1]$ (en nuestro caso $L = 256$) es una función discreta $p(r_k) = n_k/N$, donde r_k es el $k-$ésimo nivel de gris, n_k es el número de puntos en la imagen con ese nivel de gris, N es el número total de puntos en la imagen y

$k = 0, \cdots, L-1$. Es decir, $p(r_k)$ representa la probabilidad de obtener un nivel de gris k en la imagen. `MATLAB` tiene una función llamada `hist()` que hace el histograma de un vector (*Recomendación*: para extender esta función para imágenes piense que $F(:)$ es un vector cuyos elementos son los de las columnas de la matriz F concatenadas).

3.1.1. Negativo de una Imagen

Los negativos de imágenes son útiles en numerosas aplicaciones como la medicina o la fotografía. El negativo de una imagen digital es obtenido mediante la transformación representada en la gráfica izquierda de la figura 1, donde L es el número de niveles de gris (256 en las imágenes que usaremos nosotros).

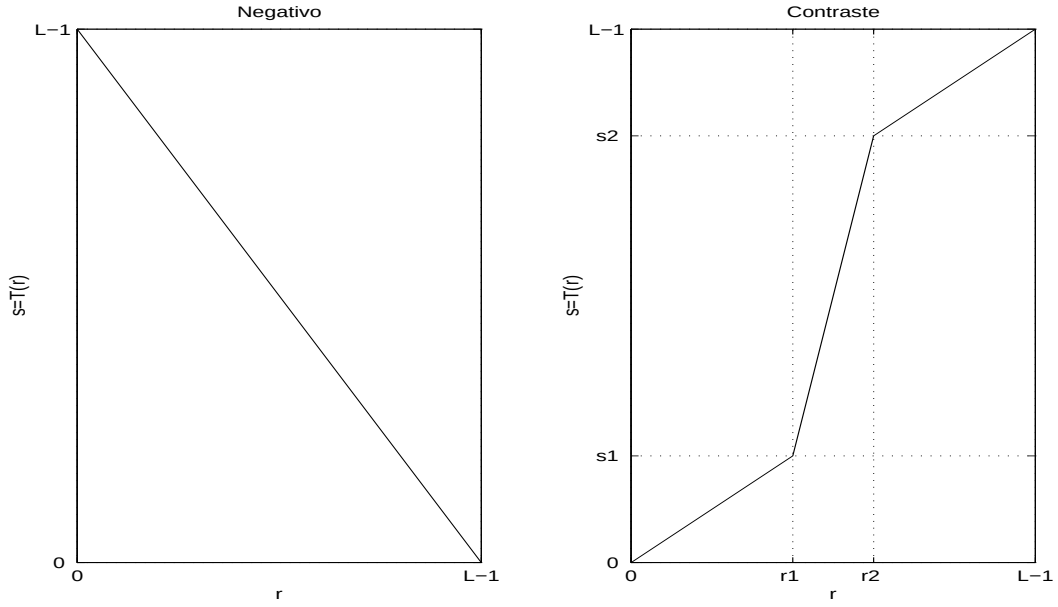

Figura 1: Izquierda: Transformación para obtener el negativo de una imagen. Derecha: Transformación para modificar el contraste de una imagen.

Escriba una función, `negativo.m`, que genere el negativo de una imagen. Visualice la curva de la transformación[2] y compruebe que es igual a la de la figura 1. Pruebe esta función visualizando su efecto sobre una de las imágenes de las fotos obtenidas en el apartado de lectura de imágenes. Presente el histograma de la imagen antes y después de ser procesada. ¿Qué observa?

3.1.2. Modificación del Contraste

El bajo contraste de ciertas imágenes puede ser debido a una pobre iluminación, falta de rango dinámico del sensor de imagen, o incluso de una apertura errónea de las lentes durante la adquisición de la imagen. La idea de la modificación del contraste se basa en la modificación del rango dinámico de las intensidades de la imagen a procesar. En la gráfica derecha de la figura 1 se puede ver una transformación típica usada para el cambio de contraste. Observe que los puntos (r_1, s_1) y (r_2, s_2) controlan la forma de la función de transformación. Por ejemplo, si $r_1 = s_1$ y $r_2 = s_2$, la transformación no produce cambios en los niveles de gris. Si $r_1 = r_2$, $s_1 = 0$ y $s_2 = L - 1$ (donde L es el número de grises), la transformación se convierte en una función de umbral que crea una imagen binaria. Valores intermedios de (r_1, s_1) y (r_2, s_2) producen varios grados de expansión de los niveles de gris, afectando así a su contraste. En general, se considera $r_1 \leq r_2$ y $s_1 \leq s_2$ para que la función sea monótona creciente y se conserve el orden de los niveles de gris.

[2]Para ello, aplique la transformación a un vector de valores entre 0 y 255 y haga un `plot` del vector transformado respecto al original.

Genere una función, `contrast.m`, que implemente una transformación de contraste como la de la figura 1 según los parámetros r_1, s_1, r_2, s_2. Pruebe esta función visualizando su efecto sobre una de las imágenes de las fotos obtenidas en un apartado anterior para:

- $r_1 = r_2 = 128$, $s_1 = 0$ y $s_2 = 255$
- $r_1 = 20$, $r_2 = 235$, $s_1 = 40$ y $s_2 = 215$
- $r_1 = 40$, $r_2 = 215$, $s_1 = 20$ y $s_2 = 235$

Visualice la curva de la transformación y compruebe que tiene una forma similar a la de la figura 1 y represente el histograma de la imagen antes y después de ser procesada. ¿Qué observa?

3.1.3. Compresión del Rango Dinámico

En ocasiones el rango dinámico de una imagen procesada excede la capacidad del dispositivo de visualización, en cuyo caso sólo las partes más brillantes de la imagen son visibles en la pantalla. Un método muy efectivo para comprimir el rango dinámico de los valores de los puntos es por medio de la siguiente transformación:

$$s = c \log_{10}(1 + |r|) \tag{3}$$

donde c es una constante de escalado y $log_{10}()$ representa el logaritmo decimal. Por ejemplo, si queremos visualizar en un display de 8 bits una imagen cuyos niveles de gris están en el rango $[0, R]$, será necesario escalar éste al de $[0, 255]$. Si utilizamos un escalado lineal, mediante una sencilla regla de tres, un valor de R muy grande (rango dinámico grande) provocará que los puntos más brillantes dominen la visualización, viéndose únicamente éstos. En este caso es conveniente el uso de un escalado no lineal como el de (3), donde $c = \frac{255}{log_{10}(1+|R|)}$.

Genere una función, `escalado.m`, que modifique el rango dinámico de los niveles de gris de la imagen `escalado.tif` (proporcionada con el enunciado) según (3). Visualice la imagen antes y después de realizar dicho escalado. ¿Cuál de las dos imágenes permite ver más detalles? Visualice también la curva de la transformación.

Este tipo de compresión se utiliza habitualmente al visualizar la transformada de Fourier de una imagen (que como veremos es otra matriz que se puede interpretar como una imagen en el dominio de la frecuencia).

3.2. Filtrado Espacial

Al igual que en una dimensión, un filtro puede caracterizarse por un sistema que transforma una señal de entrada, en nuestro caso una imagen. Estos filtros pueden clasificarse según las frecuencias que dejen pasar: paso bajo, paso banda y paso alto. En imagen, las componentes de frecuencia bajas, son las asociadas a cambios suaves de la intensidad en las imágenes y las altas, las relacionadas con los cambios bruscos como puede ser el ruido y los bordes de una figura. El filtrado paso banda se utiliza fundamentalmente para reconstrucción de imágenes. Además, otra clasificación de los filtros puede basarse en su linealidad. En los filtros (sistemas) lineales hay que recordar que pueden caracterizarse por su respuesta impulsional y por lo tanto la salida puede obtenerse por medio de una suma de convolución. La suma de convolución en 2D está especificada en la ecuación (19) del apéndice A. Matlab la tiene implementada en la función `conv2`.

En esta sección se implementarán filtros paso bajo para eliminar ruido, y filtros paso alto para realzado de bordes.

3.2.1. Filtrado de Ruido

Una imagen puede ser afectada por un ruido y/o interferencia debido a errores en el proceso adquisición, transmisión y/o almacenamiento. El ruido de imagen se aprecia como variaciones en

las intensidades de puntos aislados espacialmente e incorrelados. Los puntos afectados por el ruido aparecen claramente diferenciados de sus vecinos en la imagen. Básicamente existen dos tipos de ruido:

Gausiano : El ruido se superpone a los puntos de la imagen de forma aditiva. Su valor sigue una distribución normal.

Impulsivo : El ruido aparece en algunos puntos de la imagen distribuidos aleatoriamente. En estos puntos el valor de intensidad se modifica por valores binarios, es decir: negro o blanco.

Considerando una imagen de dimensión $N \times N$, determinada por la matriz $\mathbf{F}(n, m)$, veremos dos tipos de filtros espaciales:

- Filtros lineales paso bajo. *Media móvil*: se comportan como filtros paso bajo reduciendo el ruido de una imagen. Éste tiene componentes de alta frecuencia respecto a la de una imagen normal, debido a su falta de correlación espacial. Un filtrado paso bajo muy sencillo se puede obtener por medio de la convolución en dos dimensiones de la matriz de imagen con una matriz $L \times L$, $\mathbf{H}$, que podría ser cualquiera de éstas (para $L = 3$):

$$\mathbf{H} = \frac{1}{9}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \mathbf{H} = \frac{1}{10}\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \mathbf{H} = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix} \tag{4}$$

- Filtros no lineales. Una de las principales dificultades del filtrado lineal es que difumina los bordes y otros detalles delimitadores. Si el único objetivo es reducir ruido, una alternativa son los filtros no lineales. Este método es particularmente efectivo cuando el ruido consiste en puntos fuertes y separados, y cuando se quieren respetar los bordes de la imagen.

  - Filtros *mediana*. Estos filtros sustituyen el nivel de gris de cada punto por el valor mediano de su vecindario. El valor mediano de un conjunto de un conjunto de $2m + 1$ valores es aquel que está en el medio de los valores ordenados del conjunto, es decir, hay m valores más pequeños y m más grandes. Para implementarlo en el caso del vecindario más cercano (9 puntos vecinos), ordenamos los valores del punto y de sus ocho vecinos de menor a mayor, o viceversa, determinamos el valor de la mediana, y asignamos su valor al punto. De esta forma, en este vecindario 3×3 la mediana es el 5^o mayor (o menor) valor, y en uno 5×5 es el 13^o, etc.
  - Filtros *out-range*: Calculan la media del valor de los 8 puntos alrededor de cada punto de la imagen. Si la diferencia, en valor absoluto, de este valor medio con el valor del punto examinado es mayor que un cierto umbral, el punto en examen será sustituido por el valor medio calculado, en otro caso este no se modificará. Es decir:

$$v_{medio}(n, m) = \frac{1}{8}\left[\left(\sum_{k=-1}^{1}\sum_{l=-1}^{1}\mathbf{F}(n + k, m + l)\right) - \mathbf{F}(n, m)\right] \qquad \begin{cases} n = 0, \cdots, N - 1 \\ m = 0, \cdots, N - 1 \end{cases} \tag{5}$$

    si $|\mathbf{F}(n, m) - v_{medio}(n, m)| > \epsilon$, donde ϵ es un valor de umbral, entonces $\mathbf{F}(n, m) = v_{medio}(n, m)$. En caso contrario el valor de $\mathbf{F}(n, m)$ no se modificará[3].

Implemente una función, `pbajo.m`, que realice uno de los filtrados anteriores y que acepte como parámetros: la matriz de datos, el tamaño del vecindario y el tipo de filtro[4], y devuelva la matriz filtrada. Para probar esta función se pide que sobre alguna de las imágenes de las fotos:

[3] Este tipo de filtro puede implementarse sin utilizar bucles. Piensa que la ecuación (5) puede implementarse con una convolución 2D.

[4] El valor de ϵ en el filtro *out-range* será un parámetro más. Pruebe diferentes valores hasta encontrar uno que le convenza.

- Perturbe con ruido impulsivo la matriz de la imagen. Para hacer esto, considere un 5 % de puntos (aleatoriamente distribuidos) de la imagen distorsionados. En estos puntos, ponga un nivel de intensidad correspondiente al negro o al blanco de forma totalmente aleatoria.

  Aplique los distintos tipos de filtrado a la imagen con ruido. ¿Qué tipo de filtros se comportan mejor?

- Añada ruido gausiano a la matriz de la imagen. Para hacer esto, sume a la matriz de imagen una cuyos valores sigan una distribución gaussiana con varianza 256[5].

  Aplique los distintos tipos de filtrado a la imagen con ruido. ¿Qué tipo de filtros se comportan mejoren cada caso?

3.2.2. Realzado de bordes

El principal objetivo del realzado de bordes de una imagen es resaltar el detalle fino de ésta de forma que se visualicen las delimitaciones de ciertos atributos. De nuevo podemos hablar de filtrados lineales y no lineales.

- Filtrado lineal paso alto

  - *Básico*. La forma de la respuesta impulsional de estos filtros tiene coeficientes positivos y negativos. Estos filtros se pueden diseñar para detectar cambios o discontinuidades con una determinada orientación. Los filtros que podríamos considerar para $L = 3$, según la orientación del resaltado, son:

$$\text{NORTE:} \quad \mathbf{H} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{pmatrix} \qquad \text{SUR:} \quad \mathbf{H} = \begin{pmatrix} -1 & -1 & -1 \\ 1 & -2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

$$\text{ESTE:} \quad \mathbf{H} = \begin{pmatrix} -1 & 1 & 1 \\ -1 & -2 & 1 \\ -1 & 1 & 1 \end{pmatrix} \qquad \text{OESTE:} \quad \mathbf{H} = \begin{pmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{pmatrix} \tag{6}$$

  - *Por eliminación*. Un filtrado paso alto puede ser calculado como la diferencia entre la imagen original y una versión de ésta filtrada paso bajo, es decir

$$\text{Imagen filtrada paso alto} \quad = \quad \text{Imagen original} - \text{Imagen filtrada paso bajo} \tag{7}$$

    donde el filtro paso bajo a considerar podría ser cualquiera de los lineales utilizados en la sección anterior.

- Procesado no lineal: La detección de bordes se realiza en dos etapas a partir de la matriz de imagen $\mathbf{F}$.

  1. Resaltado de cambios en la imagen: Obtiene una diferenciación espacial de los puntos de la imagen en una dirección especificada. La matriz de diferenciación, $\mathbf{G}$, se obtendrá, según la dirección elegida, de la siguiente forma

     Horizontal $\mathbf{G}(n,m) = 2\mathbf{F}(n,m) - \mathbf{F}(n,m-1) - \mathbf{F}(n,m+1)$
     Vertical $\mathbf{G}(n,m) = 2\mathbf{F}(n,m) - \mathbf{F}(n-1,m) - \mathbf{F}(n+1,m)$

  2. Detector de umbral: Aplica un umbral a los puntos de la matriz $\mathbf{G}$ obteniendo la matriz de bordes resaltados $\mathbf{B}$, definida de la siguiente forma:

$$\mathbf{B}(n,m) = \begin{cases} 255 & si \quad \mathbf{G}(n,m) \geq \text{umbral} \\ 0 & si \quad \mathbf{G}(n,m) < \text{umbral} \end{cases} \tag{8}$$

[5]En matlab puede generar un ruido así con la instrucción `16 randn`

Implemente una función, `palto.m`, que realice uno de los filtrados anteriores y que acepte como parámetros: la matriz de datos y el tipo de filtro[6], y devuelva la matriz filtrada. Para probar esta función haga un realzado de bordes de las imágenes de las fotos con los diferentes tipos de filtros indicados.

4. Procesado Frecuencial de Imágenes

Este tipo de procesado se basa en la manipulación de la transformada de Fourier (TF) de las imágenes. La transformada de Fourier unidimensional puede ser fácilmente extendida al caso 2D. La expresión de la ecuación de análisis se encuentra en la ecuación (25), en el apéndice A. El fundamento de las técnicas de procesado frecuencial se basa en la propiedad de convolución de la transformada de Fourier que es la misma a la del caso unidimensional, es decir, la convolución 2D en el dominio espacial se convierte en un producto punto a punto en el dominio frecuencial. De esta forma, si queremos aplicar una transformación lineal a una imagen caracterizada por la respuesta impulsional del filtro a aplicar, podemos hacer las TF de la imagen y del filtro, multiplicar las matrices resultantes punto a punto y calcular la TF inversa.

Los conceptos de filtrado frecuencial aparecen de forma natural e intuitiva trabajando en el dominio de la frecuencia. Por ejemplo, un filtrado paso bajo consistirá en aumentar el módulo de las componentes de frecuencia bajas con respecto a las altas.

4.1. Cálculo y Visualización de la DFT en 2D

La DFT de una imagen representada por una matriz $\mathbf{F}$ de dimensión $N \times N$ es otra matriz $\mathcal{F}$ de las mismas dimensiones. Matlab tiene esta transformación, y su inversa, implementadas de forma eficiente en las funciones `fft2` e `ifft2`, respectivamente.

4.1.1. Compresión del Margen Dinámico

Los componentes de la DFT adquieren valores complejos cuyos módulos tienen una gran diversidad de valores. Por ello, para obtener una buena representación del módulo de la DFT es necesario realizar una compresión del rango dinámico como el realizado en el apartado 3.1.3, es decir, $\hat{\mathcal{F}} = c \log_{10}(1 + |\mathcal{F}|)$, con $c = \frac{255}{\text{maximo}(\log_{10}(1+|\mathcal{F}|))}$ y donde $\text{maximo}(|\mathcal{X}|)$ representa el valor máximo del módulo de los elementos de una matriz $\mathcal{X}$.

Para ver la utilidad de esta compresión, visualice el módulo de la DFT de la imagen del círculo de dos formas distintas: una mediante un escalado lineal cuyos valores queden en el rango $[0, 255]$ y la otra utilizando una compresión del margen dinámico como la especificada antes. ¿En cuál se observa un mayor número de matices?

A partir de aquí, todas las representaciones del módulo de la DFT en 2D se realizarán usando la compresión logarítmica del rango dinámico. *NOTA:* La DFT en 2D obtiene una matriz que representa los resultados de una función 3D y, aunque se podría visualizar como si de una imagen se tratase, para representarla es mejor utilizar curvas de nivel con la instrucción `contour`[7].

4.1.2. Centrado de la TF

Las frecuencias bajas de la imagen se sitúan en las esquinas de la DFT en 2D. Sin embargo, es más cómodo situarlas en el centro de la matriz $\mathcal{F}$, y para ello es necesario producir un desplazamiento frecuencial de la DFT de $N/2$ puntos en horizontal y en vertical. Esto se puede hacer de forma similar al caso unidimensional, modificando en este caso la matriz en el dominio espacial mediante

[6] El valor del umbral será un parámetro más. Pruebe con diferentes valores hasta encontrar uno que le convenza.
[7] En caso de interesarnos una representación 3D, se puede hacer uso de la instrucción `mesh`.

un producto de los elementos de $\mathbf{F}$ con una exponencial compleja de la siguiente forma:

$$\mathbf{F}_c(n,m) \quad = \quad \mathbf{F}(n,m)e^{j\frac{2\pi}{N}(\frac{N}{2}n+\frac{N}{2}m)} = \mathbf{F}(n,m)e^{j\pi(n+m)} = \mathbf{F}(n,m)(-1)^{n+m} \tag{9}$$

Para ver el efecto de este desplazamiento frecuencial, visualice el módulo de la DFT de la imagen del círculo antes y después del desplazamiento. A partir de ahora, todas las DFT en 2D que se consideren en la práctica, a no ser que se indique lo contrario, serán las centradas como indica (9). De todas formas, hay que tener en cuenta que este centrado frecuencial sólo es útil para visualizar de forma más intuitiva la DFT de una imagen ya que para el procesado en frecuencia no es necesario y, en caso de implementarlo, habría que deshacer este cambio.

Analice los módulos de las DFT (centrados y escalados) de las imágenes geométricas generadas anteriormente ¿Qué relación ve entre las imágenes y sus DFT? ¿En las imágenes de los rectángulos qué similitud encuentra con la DFT unidimensional?

Todas las técnicas de filtrado de las siguientes secciones consideran que la DFT de la imagen está centrada.

4.2. Filtrado Paso Bajo

Un filtrado en frecuencia se hace resaltando o eliminando componentes de frecuencia directamente en los elementos de la matriz de imagen en frecuencia, $\mathcal{F}$. Esto se puede hacer de la siguiente forma: $\mathcal{G} = \mathcal{F} \otimes \mathcal{H}$, donde $\mathcal{G}$, $\mathcal{F}$ y $\mathcal{H}$ son las DFT de la imagen filtrada, $\mathbf{G}$, la imagen original, $\mathbf{F}$, y la matriz del filtro, $\mathbf{H}$, respectivamente, y la operación $\otimes$ representa un producto elemento a elemento de dos matrices de las mismas dimensiones. Por lo tanto, las matrices $\mathcal{G}$, $\mathcal{F}$ y $\mathcal{H}$ serán del mismo tama no. Si queremos realizar un filtro paso bajo ideal, este puede ser definido en frecuencia de la siguiente forma:

$$\mathcal{H}(k,l) = \begin{cases} 1 & \text{si } d(k,l) \leq d_0 \\ 0 & \text{si } d(k,l) > d_0 \end{cases} \tag{10}$$

donde d_0 es una cantidad positiva y $d(k,l)$ es la distancia desde el punto (k,l) hasta el punto de origen del plano frecuencial (el punto central en una DFT centrada), es decir: $d(k,l) = \sqrt{k^2+l^2}$. Por lo tanto, un filtro paso bajo ideal consiste en dejar las frecuencias dentro de un círculo de radio d_0.

Implemente una función, `fpbajo2.m`, que haga un filtrado paso bajo frecuencial de una matriz con un valor d_0 pasado como parámetro.

Para probar este tipo de filtrado sume ruido impulsivo a la matriz de una imagen como en el apartado 3.2.1. La probabilidad de obtener un punto erróneo (con valor modificado) ha de ser 10^{-2}. Calcule la DFT en 2D de la imagen con ruido, aplique el filtro paso bajo en frecuencia[8], calcule la DFT inversa del resultado del filtrado en frecuencia y visualice la imagen resultante. Para comprobar que su filtro funciona correctamente, haga una representación del módulo de la DFT de la imagen antes y después de ser filtrada.

4.3. Filtrado Paso Alto

Razonando de forma similar al filtrado paso bajo, se puede obtener un filtro en frecuencia especificado en este caso, de la siguiente forma:

$$\mathcal{H}(k,l) = \begin{cases} 0 & \text{si } d(k,l) \leq d_0 \\ 1 & \text{si } d(k,l) > d_0 \end{cases} \tag{11}$$

es decir, se consideran las frecuencias fuera de un círculo de radio d_0.

Implemente una función, `fpalto2.m`, que haga un filtrado paso alto frecuencial de una matriz $\mathcal{F}$ con un valor d_0 pasado como parámetro.

[8] Elija un valor d_0 cuyo efecto le convenza.

Para probar este tipo de filtrado calcule la DFT en 2D de una imagen, aplique este filtro paso alto[9], calcule la DFT inversa del resultado del filtrado en frecuencia y visualice la imagen resultante. ¿Cuál ha sido el resultado?. Para comprobar que su filtro funciona correctamente, haga una representación del módulo de la DFT de la imagen antes y después de ser filtrada.

4.4. Compresión de Imágenes

Las imágenes concentran la mayoría de su energía en determinadas frecuencias. Esta característica puede ser aprovechada para comprimir éstas, con una ligera pérdida de calidad. Los sistemas de compresión de imagen, como en el caso de la técnica *JPEG*, almacenan los puntos de la DFT en 2D que tienen mayor concentración de energía. En este apartado se pretende realizar un sencillo compresor de imágenes basado en el procesado frecuencial.

El compresor se basa en un filtro definido en frecuencia de la forma:

$$\mathcal{H}(k,l) = \begin{cases} 1 & \text{si } |\mathcal{F}(k,l)| \geq \mathcal{F}_0 \\ 0 & \text{si } |\mathcal{F}(k,l)| < \mathcal{F}_0 \end{cases} \tag{12}$$

donde $\mathcal{F}_0$ es un umbral que determina la relación de compresión/calidad de imagen. Implemente una función, `comprime.m`, que implemente este tipo de filtrado para compresión, y que acepte como parámetro $\mathcal{F}_0$.

Para probar este compresor, obtenga la DFT de una imagen y aplíquele el filtro compresor. El tama no que ocupará la imagen está directamente relacionado con el número de puntos de la DFT distintos de cero. ¿Cuál es el número de puntos diferentes de cero de la DFT original? ¿Y de la DFT filtrada? NOTA: En matlab la instrucción `nnz()` devuelve el número de puntos diferentes de cero en un vector. Calcule la inversa de la DFT filtrada y visualice la imagen resultante. ¿Se consigue un buen factor de compresión antes de perder mucha calidad en la imagen?

4.5. Filtrado Banda Eliminada

Uno de los casos más comunes y simples de perturbación de imagen es el provocado por sinusoides en 2D que superponen a la imagen deseada. Este tipo de ruido, llamado ruido coherente, es comúnmente provocado por motores cercanos a un receptor de imagen. La matriz de imagen distorsionada será de la forma:

$$\mathbf{G}(n,m) = \mathbf{F}(n,m) + \mathbf{N}(n,m) \tag{13}$$

donde $\mathbf{F}(n,m)$ es la matriz de imagen original y

$$\mathbf{N}(n,m) = A \;\; sin(w_1 n + w_2 m) \tag{14}$$

es la matriz del ruido senoidal. Caculando la DFT en 2D de (14) se obtiene

$$\mathcal{N}(k,l) = \frac{-jA}{2} * \left[\delta\left(k - \frac{w_1}{2\pi/N}, l - \frac{w_2}{2\pi/N}\right) - \delta\left(k + \frac{w_1}{2\pi/N}, l + \frac{w_2}{2\pi/N}\right) \right] \tag{15}$$

para $k = 0, \cdots, N-1$ y $l = 0, \cdots, N-1$ y donde N es el número de filas y columnas de la imagen y de su DFT 2D. Tenga en cuenta que estamos trabajando con la DFT y que por lo tanto, al igual que en el caso unidimensional, los desplazamientos son circulares con módulo N.

Se pide que realice en un programa, `belimina.m`, lo siguiente:

- Genere y visualice una matriz de imagen de ruido senoidal, de tama no 256×256, con $w_1 = w_2 = pi/4$ y $A = 256$.
- Calcule la DFT 2D y represente su módulo. Relacione el resultado con (15).

[9] Elija un valor d_0 cuyo efecto le convenza.

- Añada este ruido a una imagen de una foto. Visualice la imagen perturbada con el ruido.
- ¿Cómo filtraría la imagen perturbada para eliminar totalmente el ruido senoidal? Fíltrela y visualice la imagen resultante.

Apartados Opcionales

4.6. Importancia de la Fase en la DFT de una Imagen (OPCIONAL)

En las secciones anteriores se ha visualizado siempre el módulo de la DFT de una imagen. Sin embargo, la información de la fase es fundamental para recuperar la imagen original a partir de la DFT.

Para ver la importancia de la fase haga un programa, `fase.m`, que realice los siguientes pasos:

- Seleccione dos de las imágenes de fotos y calcule sus DFT que llamaremos $\mathcal{F}_1$ y $\mathcal{F}_2$.
- Visualice sus fases (recuerde que en matlab la instrucción `angle` obtiene el valor de la fase de un complejo). En este caso no es necesario comprimir el rango dinámico.
- Mezcle el módulo de una con la fase de la otra de forma que queden dos nuevas DFT, es decir:

$$\mathcal{G}_1 = |\mathcal{F}_1|e^{j\angle\mathcal{F}_2} \quad (16)$$

$$\mathcal{G}_2 = |\mathcal{F}_2|e^{j\angle\mathcal{F}_1} \quad (17)$$

- Calcule las DFT inversas de $\mathcal{G}_1$ $\mathcal{G}_2$.
- Visualice los módulos de las imágenes resultantes.

¿Qué observa? ¿Qué imagen predomina, la que aporta la fase o la que aporta el módulo? ¿Qué cree que tiene más información: la fase o el módulo de la DFT?

4.7. Desplazamientos Espaciales (OPCIONAL)

La propiedad de desplazamiento de la DFT unidimensional dice que un desplazamiento temporal provoca en frecuencia un producto por una exponencial compleja. En la DFT 2D esta propiedad también se cumple. Si la DFT de una imagen $N \times N$, $\mathbf{F}(n,m)$, es $\mathcal{F}(k,l)$, entonces la de $\mathbf{F}(n-n_0, m-m_0)$ será:

$$\mathcal{F}(k,l)e^{-j*2*pi*(kn_0+lm0)/N} \quad (18)$$

Se pide que realice en un programa, `desplaza.m`, lo siguiente:

- Considere la imagen de un círculo ($N = 64$ y $M = 20$).
- Calcule la DFT de la imagen anterior y multiplique los elementos de la matriz de DFT por exponenciales de la misma forma que en (18) con un desplazamiento $n_0 = m_0 = N/2$.
- Calcule la DFT inversa de la matriz de DFT anterior.
- Visualice el resultado. ¿Qué Observa?.

5. Implementación de la Convolución en 2D (OPCIONAL)

Se pretende implementar una alternativa a la función de matlab `conv2`. Para ello, ha de implementar una nueva función, `convol2.m`, que implemente la convolución en 2D según el método matricial explicado en el apéndice. Pruebe la nueva función sobre alguna de las imágenes con figuras geométricas y compárelas con las de matlab. NOTA: Escoja un tamaño de imagen suficientemente pequeño para que la matriz $\mathbf{D}$ pueda ser manejada por matlab.

6. Implementación de la DFT en 2D (OPCIONAL)

Se pretende implementar una alternativa a la función de matlab `fft2`. Para ello, ha de implementar una nueva función, `dft2.m`, que implemente la DFT en 2D según el método matricial explicado en el apéndice. Pruebe la nueva función sobre alguna de las imágenes con figuras geométricas y compárelas con las de matlab.

7. Documentación y Fecha de entrega

La fecha de depósito de la práctica será el día **3 de junio**. Se depositarán los ficheros `*.m` de los programas de matlab utilizando el mecanismo de recogida de prácticas del CECAFI. Para una sencilla identificación de los programas realizados, **es MUY IMPORTANTE que los programas RESPETEN el nombre especificado** en cada apartado y que los códigos de éstos estén debidamente comentados.

Nota: Las respuestas planteadas a lo largo de la práctica han de incluirse en un fichero de texto llamado `respuestas.txt`.

Examen: Se recuerda que el **EXAMEN DE PRÁCTICAS** será el **lunes 7 de junio** en el laboratorio y horario de prácticas.

A. Apéndice: Definición de operadores en 2D

- **Convolución** La suma de convolución entre la imagen definida por una matriz $N \times N$ $\mathbf{F}(n,m)$ ($n = 1,\cdots,N$ y $m = 1,\cdots,N$) y una matriz de la respuesta impulsional del operador $\mathbf{H}(n,m)$ de dimensiones $L \times L$ ($n = 1,\cdots,L$ y $v = 1,\cdots,L$), será una matriz de tama no $M \times M$ de la forma

$$\mathbf{G}(m_1,m_2) = \sum_{n_1=1}^{m_1}\sum_{n_2=1}^{m_2}\mathbf{F}(n_1,n_2)\mathbf{H}(m_1-n_1+1,m_2-n_2+1) \qquad (19)$$
$$m_1 = 1,\cdots,M$$
$$m_2 = 1,\cdots,M$$

donde $M = N + L - 1$ y se ha asumido que $\mathbf{F}$ y $\mathbf{H}$ son cero fuera del rango de sus índices.
Si las columnas de las matrices $\mathbf{F}$ y $\mathbf{G}$ son concatenadas en vectores columna $N^2 \times 1$, $\mathbf{f}$, y $M^2 \times 1$, $\mathbf{g}$, respectivamente, entonces la convolución puede ser obtenida como

$$\mathbf{g} = \mathbf{D}\mathbf{f} \qquad (20)$$

donde $\mathbf{D}$ es una matriz $M^2 \times N^2$ conteniendo los elementos de la respuesta impulsional. Para especificar esta matriz es conveniente hacerlo por medio de submatrices de tama no $M \times N$. Observando los limites de los sumatorios en (19) se puede ver que:

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_{1,1} & \mathbf{0} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} \\ \mathbf{D}_{2,1} & \mathbf{D}_{2,2} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \ddots & \cdots & \cdots & \vdots \\ \mathbf{D}_{L,1} & \mathbf{D}_{L,2} & \cdots & \mathbf{D}_{L,L} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{L+1,2} & \cdots & \cdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \cdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \ddots & \cdots & \ddots & \vdots \\ \vdots & \vdots & \cdots & \mathbf{0} & \mathbf{D}_{N,N-L+1} & \cdots & \mathbf{D}_{N,N} \\ \vdots & \vdots & \cdots & \cdots & \mathbf{0} & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} & \mathbf{D}_{M,N} \end{pmatrix} \qquad (21)$$

donde $\mathbf{0}$ representa una submatriz de todos ceros y las submatrices que no son todo ceros se definen como

$$\mathbf{D}_{m_2,n_2}(m_1,n_1) = \mathbf{H}(m_1-n_1+1,m_2-n_2+1) \qquad \begin{cases} 1 \le n_i \le N \\ n_i \le m_i \le n_i + L - 1 \end{cases} \qquad (22)$$

Para ilustrar la estructura general de la matriz $\mathbf{D}$ considere el siguiente ejemplo con una matriz de imagen con $N = 3$ y una respuesta impulsional de un operador con $L = 2$, definido por la matriz

$$\mathbf{H} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \qquad (23)$$

Por lo tanto, $M = 3 + 2 - 1 = 4$ obteniendo una matriz 16×9

$$\mathbf{D} = \left(\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 4 & 2 & 0 & 3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 4 & 2 & 0 & 3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 3 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 4 & 2 & 0 & 3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 4 & 2 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 3 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \end{array}\right) \qquad (24)$$

- **DFT** Consiste en otra matriz $\mathcal{F}(k,l)$ de dimensión $N \times N$ definida como

$$\mathcal{F}(k,l) = \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \mathbf{F}(n,m) e^{-j\frac{2\pi}{N}(nk+ml)} \tag{25}$$

La matriz $\mathcal{F}$ puede ser calculada mediante el producto de matrices de la forma:

$$\mathcal{F} = \mathbf{AFA} \tag{26}$$

donde $\mathbf{A}$ es una matriz $N \times N$ de la forma

$$\mathbf{A} = \begin{pmatrix} W^0 & W^0 & W^0 & \cdots & W^0 \\ W^0 & W^1 & W^2 & \cdots & W^{N-1} \\ W^0 & W^2 & W^4 & \cdots & W^{2(N-1)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ W^0 & W^{N-1} & W^{2(N-1)} & \cdots & W^{(N-1)^2} \end{pmatrix} \tag{27}$$

siendo $W = e^{-j\frac{2\pi}{N}}$

- **IDFT**

$$\mathbf{F}(n,m) = \frac{1}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} \mathcal{F}(k,l) e^{j\frac{2\pi}{N}(nk+ml)} \tag{28}$$

La IDFT también puede ser calculada mediante el producto de matrices de la forma

$$\mathbf{F} = \mathbf{A}^{-1} \mathcal{F} \mathbf{A}^{-1} \tag{29}$$

donde $\mathbf{A}^{-1}$ es la inversa de la matriz $\mathbf{A}$ definida en (27).

Referencias

[1] Manual *online* de `matlab`. Instrucción `help`.