Práctica 2: Sistemas LSI		Grupo	
		Puesto	
Apellidos, nombre		**Fecha**	
Apellidos, nombre			

El objetivo de esta práctica es presentar al alumno la aplicación de las herramientas que ofrece el análisis de los sistemas LSI (*Linear Shift Invariant*) al tratamiento digital de imágenes.

Desarrolle cada ejercicio en un fichero de comandos 'ejercicio_X.m' separado. Para conocer el funcionamiento preciso de los comandos que se introducen en este guión, utilice la ayuda de MATLAB. Para evitar posibles interferencias con otras variables o ventanas recuerde incluir siempre las instrucciones **`clear all`** y **`close all`** al principio de cada fichero de comandos.

Justo antes de finalizar la práctica, comprima el guión y los ficheros '.m' generados en un único fichero con el nombre '**practica_2_ApellidosNombre.zip**', conéctese al sistema de entrega de prácticas de la Intranet y entréguelo en el grupo de prácticas (grupo único).

1 Análisis frecuencial

1.1 Ejercicio 1: Transformada de Fourier de sinusoides; caso particular.

Genere imágenes de las versiones discretas de las siguientes cuatro funciones; para ello defina las funciones en la región $0 \le x < 1, 0 \le y < 1$ (observe que los intervalos están abiertos en su extremo superior), muestreando con un retículo ortogonal de vectores $\vec{v}_1 = (1/100, 0), \vec{v}_2 = (0, 1/100)$.

1. $\psi_1(x, y) = \sin(10\pi x)$
2. $\psi_2(x, y) = \sin(20\pi y)$
3. $\psi_3(x, y) = \sin(40\pi x + 30\pi y)$
4. $\psi_4(x, y) = \sin(10\pi x + 20\pi y) + \sin(30\pi x + 30\pi y)$

Represente las cuatro funciones a tamaño real, de modo que cada píxel de la imagen se corresponda con un píxel de la pantalla. Recuerde que para ello deberá representar cada una en una figura distinta, utilizando el comando:

```
>> imshow(f,[min(min(f)) max(max(f))], 'InitialMagnification',100);
```

La Transformada de Fourier (FT) de una señal es una función compleja de variable real. En el caso de señales bidimensionales o imágenes, la FT toma un valor complejo para cada punto del plano de frecuencias horizontales y verticales. Para poder representarla, una posibilidad es representar su módulo normalizado (también se podrían representar por separado sus partes real e imaginaria). Esta representación puede hacerse de dos modos: una representación 3D, definiendo el plano de frecuencias y utilizando el comando **`mesh`** para obtener la superficie que representa la FT; o una representación 2D, asumiendo que el módulo de la FT es una imagen discreta. Este ejercicio propone ilustrar ambos métodos.

REPRESENTACIÓN DEL MÓDULO DE LA FT EN 3D:

En primer lugar obtenga la FT de cada una de las cuatro señales anteriores. Para ello, utilice el comando:

```
>> F=fftshift(fft2(f));
```

Tenga en cuenta que mientras que cada una de las cuatro funciones definidas presentan su origen de coordenadas en la parte superior izquierda de su representación como imágenes, la FT así obtenida presenta su origen de coordenadas en el centro del plano de frecuencias en que está definida. Además, la máxima frecuencia horizontal de las imágenes discretas será la mitad de su número de columnas (**`image_w/2`**), y la máxima frecuencia vertical la mitad de su número de filas (**`image_h/2`**). Por lo tanto, la definición del plano de frecuencias, el cálculo del módulo normalizado y su representación 3D resulta:

```
>> fx=[-image_w/2:image_w/2]; fx=fx(1:end-1);
>> fy=[- image_h/2:image_h/2]; fy=fy(1:end-1);
>> F=abs(F)/max(max(abs(F)));
>> figure; mesh(fx, fy, F);
```

Vaya representando una a una las FT de las imágenes propuestas y compruebe que son coherentes con lo que cabía esperar de la definición de las funciones. A partir de la observación de las FT obtenidas, indique la expresión analítica del módulo normalizado de la FT de cada una de las cuatro funciones:

REPRESENTACIÓN DEL MÓDULO DE LA FT COMO UNA IMAGEN:

En esta situación, en lugar de definir el plano de frecuencias y representar la FT utilizando el comando **`mesh`**, se pretende representar directamente el módulo de la FT como una imagen. Dado que en la práctica una imagen sólo contiene 256 niveles o valores distintos de la señal que representa, en vez de representar el módulo normalizado de la FT resulta más conveniente representar su logaritmo[1]:

```
>> F=log(1+abs(F));
>> figure; imshow(F,[min(min(F)) max(max(F))],'InitialMagnification',100);
```

Compruebe que lo que observa es coherente con la representación 3D realizada antes.

1.2 Ejercicio 2: Transformada de Fourier; caso general.

Utilizando el mismo retículo del ejercicio anterior, genere ahora en la región $0 \le x < 1.22, 0 \le y < 1.22$ una versión discreta de las funciones:

1. $\psi_1(x,y) = \sin(40\pi x + 30\pi y)$
2. $\psi_2(x,y) = \sin(120x + 90y)$

[1] Observe que en vez de calcular directamente el logaritmo del módulo de la FT, se le suma una unidad para evitar que el resultado tome valores negativos.

Represente la FT de estas dos señales tanto en 3D como en 2D, siguiendo el esquema del ejercicio anterior. Indique cuál es el motivo de no obtener resultados similares a los del ejercicio anterior, sobre todo en el caso de $\psi_1(x, y)$, que es idéntica a la tercera función del citado ejercicio:

1.3 Ejercicio 3: Transformada de Fourier de imágenes naturales.

La conclusión del ejercicio anterior es que si la señal cuya FT se desea obtener, o no es periódica o no presenta un número entero de periodos en la región en la que está definida, la FT calculada según se propone en el Ejercicio 1 presenta componentes frecuenciales que no corresponden a la señal en cuestión. En ambas situaciones, aunque por distintas causas, el motivo es que la replicación periódica de la señal, implícita en la obtención de la FT a partir de la FFT, genera en la señal discontinuidades que dan lugar a componentes de alta frecuencia en las direcciones horizontal y vertical.

En el caso de imágenes reales o *naturales* (así llamadas por oposición a las imágenes sintéticas), la periodicidad es en la práctica inexistente, por lo que el efecto observado en el ejercicio anterior se produciría siempre, impidiendo de este modo una correcta interpretación de las componentes frecuenciales de la imagen.

Una posible solución a este problema consiste en generar una nueva imagen formada por réplicas especulares de la imagen cuya FT se desea obtener (ver Fig. 1), y obtener a continuación la FT de esta imagen. La FT de la imagen así extendida presentará componentes frecuenciales similares a las de la imagen inicial, pero sin los efectos derivados de la no periodicidad ya que la extensión hace que la imagen sea siempre periódica.

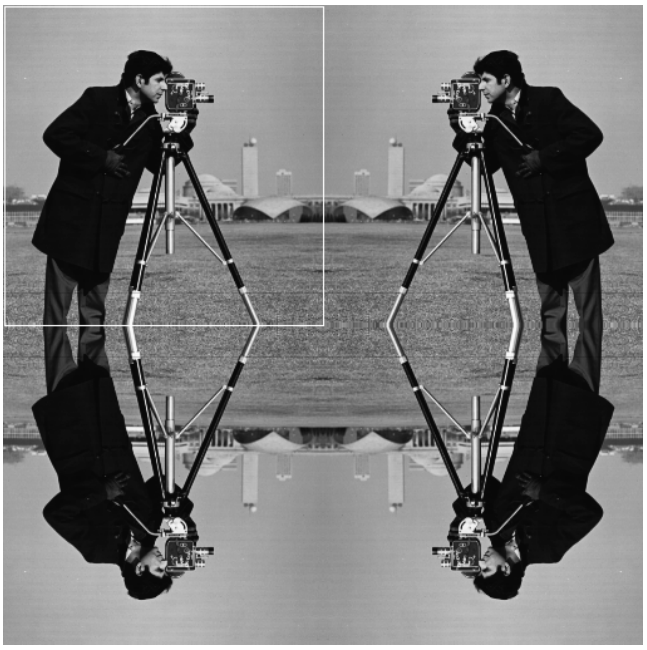

Fig.1: Imagen extendida especularmente en todas sus dimensiones

Para apreciar la dependencia de este efecto con respecto a la imagen particular con que se opere, proceda según se indica a continuación:

Obtenga la FT de la imagen **`paisaje_bw.bmp`**[2] tanto como se propone en los ejercicios anteriores como siguiendo el esquema comentado en los párrafos anteriores[3] (represente en ambos casos la imagen cuya FT calcula para asegurarse de que la extensión se realiza correctamente). Tenga en cuenta que para poder calcular la FT de la imagen tendrá que convertirla antes a tipo **`double`** (para ello use siempre **`double(ima)`**, que no modifica los valores de la imagen, en vez de usar la función **`im2double(ima)`** que además los escala). Represente en una misma ventana las imágenes 2D de las dos FTs obtenidas y describa y justifique todas las diferencias que observa entre ellas:

Repita el procedimiento anterior con la imagen **`cameraman.tif`** y anote de nuevo las diferencias entre las dos FTs obtenidas, indicando además si son o no tan acusadas cono en la imagen anterior y por qué:

Concluya, a la vista de las apreciaciones anteriores, si este método para evitar la aparición de falsas frecuencias horizontales y verticales por efecto de la replicación periódica inherente al cálculo de la FT, le parece o no adecuado para realizar una interpretación frecuencial de una imagen cualquiera:

2 Filtrado en frecuencia

Si en vez de una interpretación frecuencial de una señal, el objetivo que se persigue con el uso de la FT es utilizar su propiedad de convolución para hacer un filtrado en el dominio frecuencial, el efecto observado en los ejercicios anteriores no tiene importancia a efectos de cálculo, ya que la FT inversa lo asume.

Adicionalmente, si el filtrado se realiza generando el filtro frecuencial en vez de calculando la FT de su respuesta al impulso, no es necesario hacer *padding* de las imágenes que se convolucionan.

---

[2] Descárguese esta imagen de la página *web* de la asignatura.

[3] Para calcular la imagen propuesta en la Figura 1 utilice las funciones fliplr y flipud para reflejar la imagen original tanto en el eje x como en el eje y.

2.1 Ejercicio 4: Filtrado paso bajo y paso alto ideal.

El objetivo de este ejercicio es aprender a aplicar la técnica de filtrado frecuencial, y comprobar su validez sobre una imagen. Sin embargo no se pretende profundizar en el efecto visual de los distintos tipos de filtrado, asunto este que se tratará en prácticas posteriores.

Sea la función: $\psi_1(x,y) = \sin(2\pi 160x) + \sin(2\pi 40y)$

Genere una imagen a partir de ella muestreándola en la región $0 \le x < 1, 0 \le y < 1$ con un retículo ortogonal de vectores $\vec{v}_1 = (1/400, 0), \vec{v}_2 = (0, 1/400)$ y calcule su FT.

Un filtro 2D estará definido por una imagen de igual tamaño que la FT obtenida. Su origen estará en el centro de la imagen. Para generar filtros paso bajo o paso alto ideales deberá asignar valor unidad a sus regiones de soporte y valor nulo al resto. Tenga en cuenta que si los filtros no son simétricos respecto del origen el resultado será una imagen con valores complejos. Analice y comprenda el siguiente código de ejemplo, que genera filtros paso bajo y paso alto con frecuencia de corte horizontal en **`fc_x`** y vertical en **`fc_y`**:

```
>> v1=1/400; v2=1/400;
>> ...
>> fc_x=150*v1; fc_y=35*v2;
>> f0_x=0.5+v1; f0_y=0.5+v2;
>> fpb=(X>(f0_x-fc_x) & X<(f0_x+fc_x)) & (Y>(f0_y-fc_y) & Y<(f0_y+fc_y));
>> fpa=~fpb;
>> fpb=double(fpb); fpa=double(fpa);
```

Para obtener la imagen resultante de un proceso de filtrado deberá aplicar la FT inversa. Por motivos de precisión numérica, esta operación genera valores complejos aun en el caso de que el resultado sea una imagen real y valores muy pequeños pero no nulos aun en el caso de que el resultado teórico fuera nulo. Para resolver la primera situación suele tomarse la parte real del resultado; para resolver la segunda, suele redondearse al intervalo de cuantificación, que depende del rango de la imagen inicial (en este caso una suma de dos sinusoides, cuyo rango varía desde -2 a +2, es decir, 4 unidades) y de la posible ganancia del filtro aplicado (en nuestro caso de ganancia unidad):

```
>> ima_range=4; gain=1; quant=ima_range*gain/256;
>> filtered_ima=quant*round(real(ifft2(ifftshift(F)))/quant);
```

Tomando como ejemplo el código indicado, genere y represente filtros paso bajo y los paso alto correspondientes con frecuencias de corte en **`fc_x=fc_y=30`** y **`fc_x=fc_y=100`**. Compruebe visualmente que la extensión de su región de soporte se corresponde con las frecuencias de corte definidas. Represente a tamaño real la FT de la imagen propuesta. Aplique cada filtro sobre dicha imagen y represente la imagen resultante. Comente el resultado obtenido indicando su coherencia con la FT de la imagen de partida y con los distintos filtros aplicados:

2.2 Ejercicio 5: Grados de libertad en el filtrado 2D

El ejercicio anterior considera un filtro de forma rectangular, definido en base a frecuencias de corte horizontal y vertical. En este apartado se profundiza en otro aspecto del filtrado 2D: la forma de los filtros, con independencia de sus frecuencias de corte en las direcciones de los ejes principales.

Asumiendo que la frecuencia de corte es igual en ambas direcciones (**`fc_x=fc_y=fc`**), el siguiente código define un filtro paso bajo ideal cuadrado, otro romboidal y otro circular, todos con igual frecuencia de corte:

```
>> fpb_cuad=double((X>(f0_x-fc) & X<(f0_x+fc)) & (Y>(f0_y-fc) & Y<(f0_y+fc)));
>> fpb_romb=double((abs(X-f0_x)+abs(Y-f0_y))<fc);
>> fpb_circ=double((((X-f0_x).*(X-f0_x)+(Y-f0_y).*(Y-f0_y)).^(1/2))<fc);
```

Sea la función: $\psi_1(x,y)=\sin(2\pi 90x)+\sin(2\pi 60x+2\pi 60y)+\sin(2\pi 90x-2\pi 90y)$

Genere una imagen a partir de ella muestreándola en la región $0\le x<1, 0\le y<1$ con un retículo ortogonal de vectores $\vec{v}_1=(1/400,0), \vec{v}_2=(0,1/400)$ y calcule su FT.

Genere y represente en una misma figura filtros paso bajo de las tres formas indicadas con frecuencia de corte en **`fc=100`**. Compruebe visualmente que la extensión de su región de soporte se corresponde con las frecuencias de corte definidas. Represente a tamaño real la FT de la imagen propuesta. Aplique cada filtro sobre dicha imagen y represente la imagen resultante. Comente el resultado obtenido indicando su coherencia con la FT de la imagen de partida y con los distintos filtros aplicados:

3 Introducción al diezmado

El objetivo de este apartado es aprender a diezmar e interpolar imágenes con esquemas básicos, observar el efecto del submuestreo sobre imágenes naturales y aprender a evitarlo utilizando un filtrado *antialiasing* previo.

3.1 Ejercicio 6: Diezmado con retículos de celda unidad cuadrada

Para diezmar una imagen $\psi[n,m]$ de tamaño MxN con un retículo que muestrea tomando uno de cada D píxeles en cada dirección (horizontal y vertical), basta con crear una imagen $\psi_D[n,m]$ de tamaño $M/D\,x\,N/D$ y asignar los píxeles de esta nueva imagen según:

$$\psi_D[n,m]=\psi[nD,mD]$$

Efectúe esta operación sobre la imagen **edificio_bw.bmp**[4] para un factor $D = 2$. Para ello, cree una función **diezmar(imagen,factor)** que realice la operación de diezmado devolviendo la imagen resultante. Posteriormente, represente a tamaño real tanto la imagen original como la imagen diezmada. Observe la imagen diezmada e intente localizar algún defecto debido a aliasing (recuerde que este efecto consiste en una reflexión de frecuencias, y que se produce en zonas de la imagen con alta variación). Si lo encuentra, descríbalo:

NOTA: como comprobación del ejercicio, calcule la energía de la imagen original y diezmada resultante (de tipo **`double`**) y verifique que su valor es $6.6845 \cdot 10^7$ y $1.4251 \cdot 10^9$ respectivamente.

3.2 Ejercicio 7: Interpolación en la misma dirección del retículo de muestreo

En este caso la interpolación se puede abordar como una extensión de la que se realiza en una dimensión. Así, para interpolar una imagen $\psi[n,m]$ de tamaño MxN por un factor I basta con crear una nueva imagen $\psi_0[n,m]$ de tamaño $IMxIN$ y asignar los píxeles de esta nueva imagen según:

$$\psi_0[n,m] = \begin{cases} \psi[n/I, m/I] & \text{,si } n \text{ y } m \text{ son múltiplos de } I \\ 0 & \text{, resto} \end{cases}$$

A continuación, se aplica un filtrado paso bajo de frecuencias de corte normalizadas $1/2I$ (es decir, un filtro frecuencial de tamaño $IMxIN$ con frecuencias de corte igual a la mitad de las dimensiones de la imagen dividida por I), de ganancia I^2 y de forma lo más cercana posible al ideal. En este sentido, en vez de utilizar un filtro paso bajo ideal es más conveniente utilizar un filtro paso bajo de Butterworth de orden 5, cuya implementación ilustra el siguiente código:

```
>> D_f0=((X-f0_x).*(X-f0_x)+(Y-f0_y).*(Y-f0_y)).^(1/2);
>> fpb_circ=double(ones(image_h,image_w)./(1+(D_f0./fc).^(2*orden)));
```

El objetivo de este ejercicio es ampliar la imagen resultante del apartado anterior para observar con mayor facilidad las diferencias entre la imagen original, **`edificio_bw.bmp`**, y la imagen diezmada. Para ello deberá interpolar por un factor $I = D = 2$. Para la operación de interpolado cree una función **`interpolar(imagen,factor)`** que devuelve la imagen interpolada resultante. Recuerde que para calcular la FT inversa deberá aplicar el proceso descrito en el ejercicio 5.

Efectúe, por tanto, esta operación y represente tanto la imagen original como la imagen interpolada. Comente las diferencias que observa entre ambas, indicando nuevamente si observa defectos debidos al *aliasing* espectral.

---

[4] Descárguese esta imagen de la página *web* de la asignatura.

Adicionalmente, pruebe a incluir la ganancia del filtro paso bajo de Butterworth (I^2) en el proceso de cuantificación (descrito en el Ejercicio 4) y comente los resultados obtenidos:

NOTA: como comprobación del ejercicio, calcule la energía resultante tras el diezmado e interpolación (de tipo **`double`**) y verifique que su valor es $5.6937 \cdot 10^9$

3.3 Ejercicio 8: Filtrado antialiasing previo al diezmado.

El objetivo de este ejercicio es mostrar como la aplicación de un filtrado paso bajo previo sobre la imagen que se desea diezmar, evita los posibles defectos debidos al *aliasing* espectral. Según se ha visto en las clases teóricas, el filtro debe ser ideal con frecuencia de corte normalizada $1/2D$ (es decir, un filtro frecuencial del mismo tamaño MxN que la imagen que se diezma con frecuencias de corte igual a la mitad de las dimensiones de la imagen dividida por D), ganancia unidad, y región de soporte igual a la celda de Voronoi del retículo con el que se submuestrea (en este caso cuadrada).

Repita los dos ejercicios anteriores, pero esta vez filtrando previamente la imagen **`edificio_bw.bmp`**. Para evitar el efecto de *ringing* que introduce el filtrado paso bajo ideal, utilice un filtro de Butterworth de orden 5 (que, aunque no tiene una región de soporte cuadrada, presenta una región de soporte inscrita en ella, por lo que resulta una buena aproximación). Represente la imagen original y el resultado de la imagen interpolada y observe y comente las diferencias respecto la situación del ejercicio anterior:

NOTA: como comprobación del ejercicio, calcule la energía de la imagen obtenida tras el filtrado previo y de la imagen obtenida tras el diezmado e interpolación (ambas de tipo **`double`**) y verifique que su valor es $5.6908 \cdot 10^9$ y $5.6895 \cdot 10^9$ respectivamente.