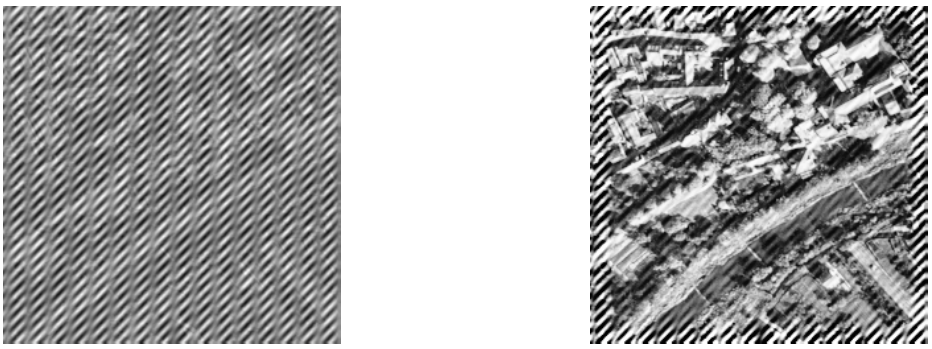
IEE239 - Procesamiento de Señales e Imágenes Digitales
Laboratorio 05 - Ejercicios propuestos
Segundo Semestre 2017

1. **Filtrado de patrones repetitivos en una imagen**

   Por errores de compresión durante el envío de datos, una imagen panorámica correspondiente a una serie de adquisiciones hechas con un drone resultó dañada. Como consecuencia, se evidencia la presencia de franjas verticales igualmente espaciadas a lo largo de la imagen, siendo imposible distinguir la información contenida. Para restaurarla, se propone lo siguiente.



(a) (a)

Figura 1: Proceso de filtrado de una imagen en el dominio de la frecuencia, (a) Imagen de entrada cuyo ruido sigue un patrón. (b) Imagen sin ruido filtrada en frecuencia.

   a. Leer la imagen **'panoramica.png'**[1] usando el comando **imread()**. Realizar zero padding a la imagen leída. A continuación, mostrar el espectro de magnitud y de fase en frecuencia, empleando **fftshift()**, **fft2()**, **unwrap()**, **abs()** y **angle()**. Por último, graficar ambos usando **imagesc()**. ¿Qué significado tiene el espectro de fase?

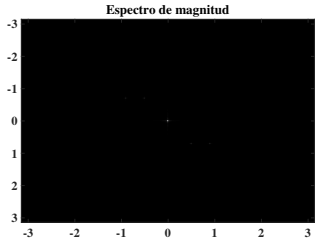
      Solución:

      Luego de leer la imagen, se pide realizar zero padding. Esto siempre se realiza antes de efectuar la DFT a una imagen. Recordar que la multiplicación en frecuencia es equivalente a la convolución circular en el espacio de muestras. Esto significa que en caso de no realizar zero padding, el resultado obtenido de un lado de la imagen podría afectar los valores del lado opuesto de la misma.

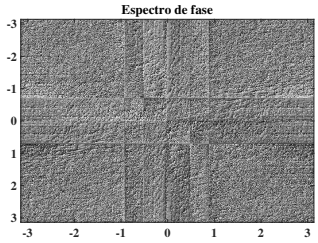
[1] La imagen está almacenada en la carpeta 'laboratorio/lab05/Propuestos/'

Posteriormente, se calculan los vectores de frecuencia normalizados ($[0 - \pi]$, para ambas direcciones), que serán los ejes coordenados del espectro a calcular. El espectro de magnitud indica la proporción de cada componente sinusoidal presente en la imagen, mientras que el espectro de fase indica la localización de cada componente sinusoidal.

```
1  %leemos la imagen
2  I = imread('panoramica.png');
3  %hacemos zero padding
4  [M N] = size(I); %A partir de la imagen, se extrae el numero de ...
       filas en la variable M y el numero de columnas en la variable N.
5  M = 2*M; N = 2*N;
6  %calculamos la FFT y los vectores de frecuencia
7  F = fftshift(fft2(I,M,N)); %con fftshift() se centra la DFT.
8  u_v = 2* pi* (0: M- 1)/ M;
9  u_v = unwrap(fftshift(u_v)-2*pi);%se obtiene el vector de ...
       frecuencia normalizado
10 v_v = 2* pi* (0: N- 1)/ N;
11 v_v = unwrap(fftshift(v_v)-2*pi);%se obtiene el vector de ...
       frecuencia normalizado
12 %graficamos el espectro de magnitud
13 subplot(121),imagesc(u_v,v_v,abs(F));colormap gray
14 subplot(122),imagesc(u_v,v_v,angle(F));colormap gray
```



(a)



(a)

Figura 2: Aplicación de la DFT sobre la imagen de entrada, obteniendo (a) espectro de magnitud y (b) espectro de fase.

b. Generar nuevamente el espectro de magnitud, esta vez usando logaritmo para visualizar mejor los detalles del mismo (como la localización del ruido). Para ello, usar **log()** sobre el espectro de magnitud para mayor claridad en la visualización. A continuación, normalizar el resultado. Luego, guardar en variables las coordenadas de cada pico correspondiente a la componente ruidosa.

Solución:

El rango dinámico del espectro de Fourier es mayor al rango típico de las imágenes (0-255). Para una mejor visualización, se puede mostrar el logaritmo del espectro de magnitud hallado. Así, se aplica la transformación logarítmica

$$F = c \log(EM + 1),$$

donde $c = 255/log(1 + 255)$ y EM hace referencia al espectro de magnitud. Luego, se normaliza el resultado F. Si se compara este resultado con el obtenido en el ítem anterior, se hace evidente la facilidad que ahora presenta para analizar la localización del ruido (representado como regiones o puntos blancos). Si bien es cierto que el ruido en este caso presenta una apariencia de destello, para eliminarlo en gran medida bastará con quitar la región central de cada destello.

```
% calculamos la FFT
[M N]=size(I);
ft=fftshift(fft2(I,M,N));
EM=abs(ft);
c=255/log(1+255);
%usamos logaritmo para visualizar mejor los detalles del espectro
F=c*log(EM+1);
F_norm=(F - min(F(:))) / (max(F(:)) - min(F(:)));
imshow(F_norm)
% Usando el cursor, se localiza las coordenadas del centro del espectro
cx = 129;
cy = 129;
% Usando nuevamente el cursor, se calcula las coordenadas del ...
    centro de cada destello visualizado en el espectro de magnitud ...
    (corresponfiente al ruido) y las guardamos en variables.
wx1 = 149.5-129;
wx2 = 165.5-129;
wy  = 157.5-129;
p1=[cx+wx1;cy+wy];
p2=[cx+wx2;cy+wy];
p3=[cx-wx1;cy-wy];
p4=[cx-wx2;cy-wy];
```

c. Se debe eliminar las componentes de ruido de la imagen. Para ello, implementar un filtro rechazabanda, dado por

$$H(u,v) = 1 - \exp\left(- \frac{(u-u_0)^2 + (v-v_0)^2}{\sigma^2} \right), \tag{1}$$

donde (u_0, v_0) es la coordenada de cada pico de ruido y considerar $\sigma = 5$. Para obtener u, v usar **meshgrid()**. Multiplicar el filtro rechazabanda resultante por la DFT 2D.

Solución:

```
% preparando variables para W
u = 0:(M-1);
v = 0:(N-1);
%Se usa meshgrid para convertir los vectores u y v en matrices, de ...
    forma que sea mas conveniente evaluar funciones de dos ...
    variables, y sea mas practico generar los filtros pedidos.
[WX,WY] = meshgrid(v, u);
sigma=5;
%se genera cuatro filtros, cada uno correspondiente a la posicion ...
    de un pico de ruido.
filter1=1-exp(-((WX-p1(1)).^2+(WY-p1(2)).^2)/sigma^2);
filter2=1-exp(-((WX-p2(1)).^2+(WY-p2(2)).^2)/sigma^2);
filter3=1-exp(-((WX-p3(1)).^2+(WY-p3(2)).^2)/sigma^2);
filter4=1-exp(-((WX-p4(1)).^2+(WY-p4(2)).^2)/sigma^2);
mask=filter1+filter2+filter3+filter4 -3; %se suman los cuatro ...
    filtros para generar un filtro total, pero se resta 3 a la ...
    suma, con la finalidad de que los valores finales queden entre ...
    0 y 1.
FF=ft.*mask; %se multiplica la DFT 2D por el filtro total, para ...
    eliminar los picos de ruido.
%se procede a graficar el espectro antes y despues de la aplicacion ...
    de los filtros. Notar que solo se aplica la transformacion ...
    logaritmica al espectro luego de ser multiplicado por el filtro ...
    (el anterior ya estaba transformado).
```

```
15 subplot(121),imagesc(u_v,v_v,abs(F)) ; colormap gray
16 subplot(122),imagesc(u_v,v_v,log(abs(FF)+1)) ; colormap gray
```

d. Calcular la IDFT 2D luego de aplicar el filtro rechazabanda sobre la DFT 2D de la imagen de entrada. Extraer la parte real del resultado usando **real()** y normalizar respecto a la intensidad. Luego, ecualizar el histograma usando **histeq()** y mostrar la imagen. ¿Se logra eliminar el ruido de la imagen? ¿Qué defectos presenta la imagen final y cómo pueden ser solucionados? Explicar.

Solución:

Luego de la aplicación del filtro propuesto, se logra eliminar gran parte del ruido en la imagen, aunque aún hay tenue presencia de las franjas de ruido. En los extremos de la imagen éste sigue presente. Si se visualiza el espectro después de la aplicación del filtro rechazabanda (mostrado en el ítem anterior), se puede comprobar que aún hay destellos pronunciados, lo cual significa que habrá ruido. Una forma de eliminar el ruido en mayor medida es usando un filtro rechazabanda con un valor de σ más grande, pues permitirá enmascarar una mayor región perteneciente al ruido. Por otro lado, para poder eliminar de forma más efectiva el ruido, sería necesaria una máscara que tenga la misma forma de los destellos, aunque su generación requeriría algo de trabajo adicional.

```
1 %calculando la IFFT y extrayendo la parte real
2 FF2 = real(ifft2(ifftshift(FF)));
3 %normalizando respecto a la intensidad
4 F_norm2=(FF2 - min(FF2(:))) / (max(FF2(:)) - min(FF2(:)));
5 %ecualizando el histograma
6 img_res = histeq(F_norm2);
7 imshow(img_res);
```

e. Rotar la imagen final (obtenida en el ítem anterior) un ángulo de 35° usando la función imrotate(). A continuación, calcular el nuevo espectro de magnitud. Antes de realizar esto, responda lo siguiente: ¿Qué espera que suceda con el espectro? Realice lo señalado en el enunciado. ¿Se comprueba su respuesta inicial?

Solución:

Por teoría, se sabe que la transformada de Fourier no es invariante ante rotaciones, esto es, si se gira una imagen un determinado ángulo, el espectro de magnitud de la transformada de Fourier también debería rotar en la misma proporción. Se procede a su implementacion.

```
1 A=img_res;%renombramos la imagen resultante
2 B = imrotate(A,35);%rotamos la imagen resultante
3 [M N] = size(A);
4 M = 2*M; N = 2*N;
5 % Calculamos la DFT 2D
6 F_A= fftshift(fft2(A,M,N)) ;
7 F_A=log(F_A+1); %aplicamos transformacion logaritmica
8 F_B= fftshift(fft2(B,M,N)) ;
9 F_B=log(F_B+1); %aplicamos transformacion logaritmica
10 % calculamos los vectores de frecuencia
11 u_v = 2* pi* (0: M- 1)/ M;
12 u_v = unwrap(fftshift(u_v)-2*pi);% se obtiene el vector de frecuencia ...
     normalizado
13 v_v = 2* pi* (0: N- 1)/ N;
14 v_v = unwrap(fftshift(v_v)-2*pi);% se obtiene el vector de frecuencia ...
     normalizado
```

```
15 subplot(121),imagesc(u_v,v_v,F_A);colormap gray
16 subplot(122),imagesc(u_v,v_v,F_B);colormap gray
```

Con esto, se comprueba que en efecto el espectro de magnitud rota de manera similar a la rotación de la imagen.

2. **Segmentación de un tumor**

Con la finalidad de conocer con mayor exactitud el estado del cáncer, frecuentemente se requiere medir el tamaño de un tumor. Con el empleo de resonancia magnética se obtuvo la siguiente imagen de la cabeza de una persona, donde se revela la presencia de un tumor. Antes de medir su tamaño, es necesario segmentarlo.

a. En primer lugar, leer la imagen **'mri.jpg'**[2]. Luego, recortar 5 pixeles en cada lado(5 filas al inicio y al final, 5 columnas el inicio y al final), con la finalidad de eliminar el borde blanco que presenta.

b. Calcular el histograma de la imagen, y calcular 3 umbrales a partir de él usando **multithresh()**. Usando el segundo umbral, binarizar la imagen.

c. Generar un disco de tamaño 5×5 de la siguiente forma:

```
1 se = strel('disk',5);
```

Para unir puntos inconexos en la imagen, ejecutar la siguiente línea de comando sobre la imagen obtenida del ítem anterior:

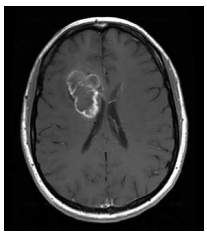
```
1 bw2 = imclose(seg_I,se);
```

A continuación, sobre 'bw2' emplear **bwlabel()**, para asignarle un valor al cráneo, y otro al fondo y demás regiones. Luego, de acuerdo a los valores asignados, generar una máscara correspondiente sólo al cráneo. Multiplicar esta máscara con la imagen original, de forma que sólo quede el cerebro.

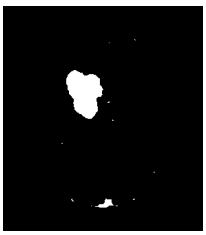
d. Con el cráneo fuera (era un gran problema, por tener intensidad similar al tumor) es turno de segmentar el tumor. Calcular el histograma de la imagen del cerebro, y a partir de éste definir dos umbrales con **multithresh()**. Usando el segundo umbral, binarizar la imagen del cerebro, con lo cual debe quedar sólo el tumor, cuyo interior no está completo. Aplicar nuevamente **imclose()** de forma similar al ítem anterior para conectar segmentos inconexos. Para rellenar el espacio vacío, usar **imfill()** con la opción **'holes'**. Finalmente, presentar la máscara con el tumor segmentado.

e. Añadir ruido gaussiano a la imagen original **'mri.jpg'**[3]. Para ello, emplear la función **imnoise()** con $\sigma = 0.05, 0.2, 0.4$ y $\mu = 0$. A continuación, sobre cada imagen con ruido aplicar un detector de bordes usando **edge()**, especificando la opción 'log' (laplaciano de gaussiano) y usando un umbral de 0.008. Indicar cómo es la calidad de la detección a medida que la varianza del ruido aumenta. Repetir el procedimiento pero ahora especificando la opción 'canny'.

[2]La imagen está almacenada en la carpeta 'laboratorio/lab05/Propuestos/'
[3]La imagen está almacenada en la carpeta 'laboratorio/lab05/Propuestos/'



(a)



(a)

Figura 3: Resultado de la segmentación del tumor, donde (a) Imagen de resonancia magnética de la cabeza y (b) máscara de la segmentación del tumor.

3. Análisis de fase y filtrado en el dominio de la frecuencia

a. Leer la imagen **chess.mat**[4] usando el comando **load** y calcular la transformada de Fourier 2D. Mostrar el espectro de magnitud y de fase en frecuencia empleando **imagesc**. Normalizar el espectro de magnitud y luego utilizar una transformación logaritmica $c \cdot \log(|FFT| + 1)$ con $c = 1$ para mejorar la visualización. Utilizar las funciones **fftshift, fft2, unwrap, abs** y **angle**. ¿Analizando el espectro de fase, puede concluir que la imagen presenta ruido? Justifique su respuesta. NOTA: para las transformaciones en frecuencia utilizar como número de mustras en frecuencia, el tamaño de la imagen original $M = N = 512$.

Solución: Se lee la imagen y se calculan los vectores de frecuencia normalizados ($[-\pi - \pi]$), que serán los ejes coordenados del espectro a calcular. Como se observa en la Figura 4, el espectro de magnitud indica la proporción de cada componente sinusoidal presente en la imagen, mientras que el espectro de fase da información de bordes de la imagen y orientación de las componentes frecuenciales de la misma. Asimismo, del espectro de fase concluimos que la imagen presenta ruido debido a que no solo se observa el patrón repetitivo de ajedrez sino también componentes aleatorias en otras frecuencias.

```
1 % leemos la imagen
2 load('chess.mat');
3 % hacemos zero padding
4 [M,N] = size(I); %A partir de la imagen, se extrae el numero de ...
      filas en la variable M y el numero de columnas en la variable N.
5 % calculamos la FFT y los vectores de frecuencia
6 F = fftshift(fft2(I,M,N)); % con fftshift() se centra la DFT.
7 u_v = 2* pi* (0: M- 1)/ M;
8 u_v = unwrap(fftshift(u_v)-2*pi);% se obtiene el vector de frecuencia ...
      normalizado
9 v_v = 2* pi* (0: N- 1)/ N;
10 v_v = unwrap(fftshift(v_v)-2*pi);% se obtiene el vector de frecuencia ...
      normalizado
11 % graficamos el espectro de magnitud
12 c = 1;
13 Flog = abs(F);
14 Flog = (Flog-min(Flog(:)))/(max(Flog(:))-min(Flog(:))); % ...
      normalizamos el espectro magnitud
```

[4] La imagen está almacenada en la carpeta laboratorio/lab05/Propuestos/'

```
Flog = c*log(Flog+1); % calculamos el logaritmo por razones de ...
    visualizacion
figure,
subplot(121),imagesc(u_v,v_v,Flog);colormap gray; title('Espectro de ...
    magnitud');
subplot(122),imagesc(u_v,v_v,angle(F));colormap gray; title('Espectro ...
    de fase');
```

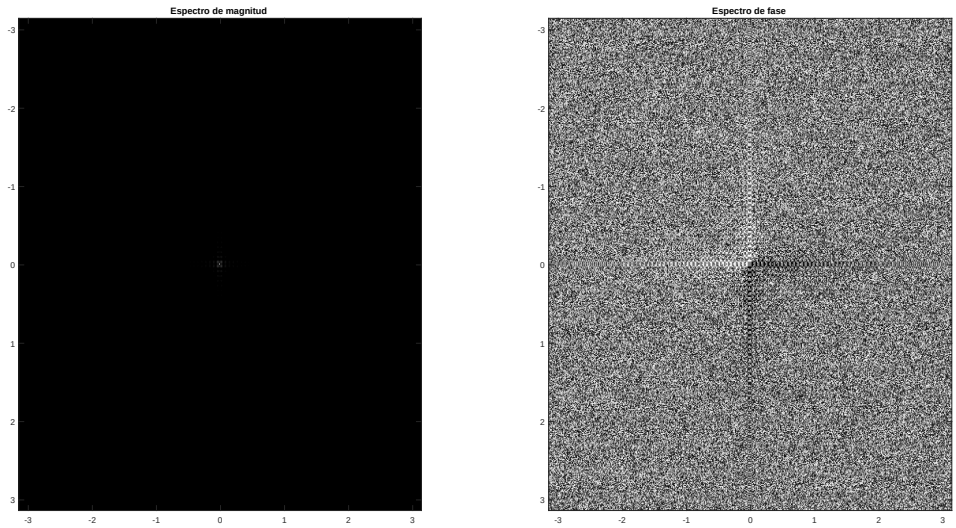


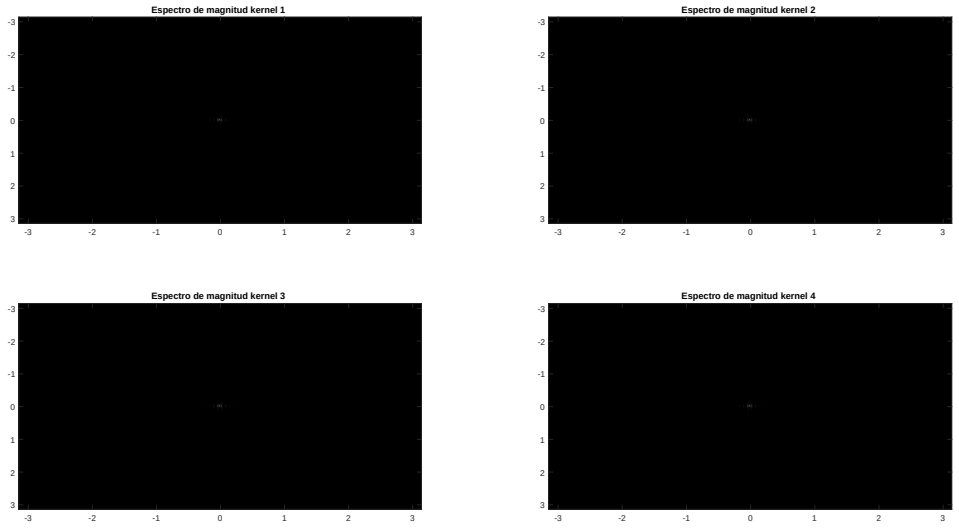
Figura 4: Aplicación de la DFT sobre la imagen de entrada, obteniendo espectro de magnitud y espectro de fase.

b. Se conoce que la imagen ha sido corrompida con ruido gaussiano aditivo; sin embargo no se conocen los parámetros del ruido. Utilizar la función **fspecial** para definir cuatro kernels que corresponden a filtros gaussianos de $\mu = 0$. Los kernel 1 y 2 deben tener $\sigma^2 = 1$ con tamaño 3×3 y 7×7 respectivamente. Los kernel 3 y 4 deben tener $\sigma^2 = 2$ con tamaño 11×11 y 13×13 respectivamente. Calcular el resultado de aplicar dichos filtros a la imagen del item anterior (utilizar la función **conv2** con la opción **'same'**). Calcular los espectros de magnitud y fase para cada uno de los filtros. ¿Qué efecto tiene incrementar σ^2 y el tamaño del kernel? Utilizar la misma transformación logaritmica que en el ítem anterior para el espectro de magnitud.

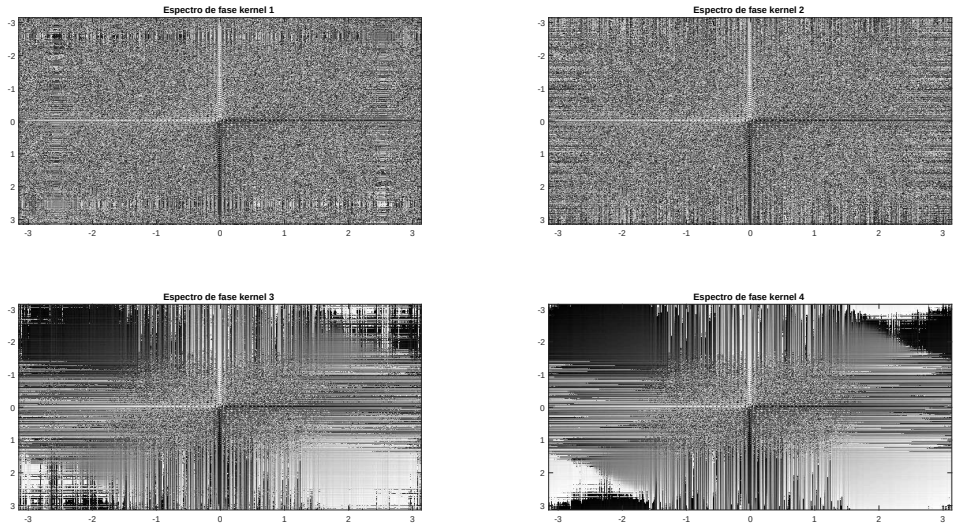
Solución: Se puede observar en las imágenes de magnitud en la Figura 5 que tanto el incremento del σ^2 como el tamaño del kernel se reflejan en una banda de paso más estrecha. Asimismo, para el caso de un σ^2 mayor, se puede observar que se empiezan a notar las réplicas debido a la periodicidad de la transformada de Fourier.

```
% filtro gaussiano
k1 = fspecial('gaussian',[3 3],1);
k2 = fspecial('gaussian',[7 7],1);
k3 = fspecial('gaussian',[11 11],2);
k4 = fspecial('gaussian',[13 13],2);
% se obtiene el resultado en espacio
ib1 = conv2(I,k1,'same');
```

```
ib2 = conv2(I,k2,'same');
ib3 = conv2(I,k3,'same');
ib4 = conv2(I,k4,'same');
% calculamos la FFT2
Fib1 = fftshift(fft2(ib1)); % centramos la DFT
Fib2 = fftshift(fft2(ib2));
Fib3 = fftshift(fft2(ib3));
Fib4 = fftshift(fft2(ib4));
% graficamos los espectros de magnitud y de fase
figure,
% mostramos el logaritmo del espectro magnitud para poder visualizar.
logFib1 = abs(Fib1);
logFib1 = (logFib1 - min(logFib1(:)))/(max(logFib1(:)) - ...
    min(logFib1(:))); % normalizamos el nuestro espectro de magnitud
logFib1 = log(logFib1+1);
logFib2 = abs(Fib2);
logFib2 = (logFib2 - min(logFib2(:)))/(max(logFib2(:)) - ...
    min(logFib2(:))); % normalizamos el nuestro espectro de magnitud
logFib2 = log(logFib2+1);
logFib3 = abs(Fib3);
logFib3 = (logFib3 - min(logFib3(:)))/(max(logFib3(:)) - ...
    min(logFib3(:))); % normalizamos el nuestro espectro de magnitud
logFib3 = log(logFib3+1);
logFib4 = abs(Fib4);
logFib4 = (logFib4 - min(logFib4(:)))/(max(logFib4(:)) - ...
    min(logFib4(:))); % normalizamos el nuestro espectro de magnitud
logFib4 = log(logFib4+1);
figure,
subplot(221), imagesc(u_v,v_v,logFib1);colormap gray; title('Espectro ...
    de magnitud kernel 1');
subplot(222), imagesc(u_v,v_v,logFib2);colormap gray; title('Espectro ...
    de magnitud kernel 2');
subplot(223), imagesc(u_v,v_v,logFib3);colormap gray; title('Espectro ...
    de magnitud kernel 3');
subplot(224), imagesc(u_v,v_v,logFib4);colormap gray; title('Espectro ...
    de magnitud kernel 4');
figure,
subplot(221),imagesc(u_v,v_v,angle(Fib1));colormap gray; ...
    title('Espectro de fase kernel 1');
subplot(222),imagesc(u_v,v_v,angle(Fib2));colormap gray; ...
    title('Espectro de fase kernel 2');
subplot(223),imagesc(u_v,v_v,angle(Fib3));colormap gray; ...
    title('Espectro de fase kernel 3');
subplot(224),imagesc(u_v,v_v,angle(Fib4));colormap gray; ...
    title('Espectro de fase kernel 4');
```



(a)



(b)

Figura 5: Proceso de filtrado de una imagen en el dominio del espacio, (a) Espectro magnitud utilizando diferentes kernels. (b) Espectro de fase utilizando diferentes kernels.

c. Utilizar solo los espectros de fase calculados en el ítem anterior para recontruir las imágenes. Utilizar las funciones **ifftshift, ifft2**. Mostrar en una gráfica las 4 imágenes recontruidas. ¿Qué puede concluir del efecto de los filtros evaluando la reconstrucción de fase?

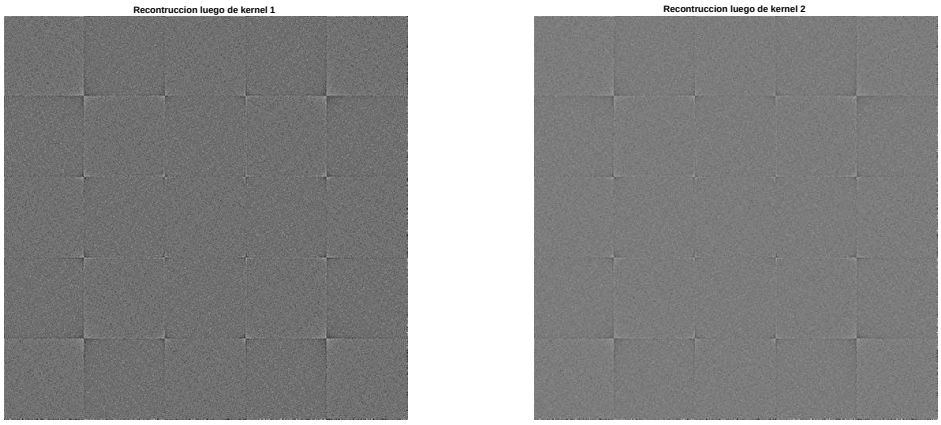
Solución: Se observa en la Figura 6 que la fase mantiene información sobre los bordes de la imagen. Asimismo, para los kernels 2 y 4 se puede observar que los bordes son mas tenues, lo que indica que en la imagen se ha logrado eliminar parte del ruido; sin embargo también se ha generado un efecto de difuminado.

```
%recontruimos utilizando solo informacion de fase
imrec01 = real(ifft2(exp(1i*(ifftshift(angle(Fib1))))));
%con angle calculamos la fase, con ifftshift eliminamos el efecto del
%ffthshift y genero la funcion compleja solo con la fase ... exp(1i*fase) y
%finalmente calculo la transformada inversa y me quedo solo con la parte
%real. Revisar la variable y notaran que la parte compleja es 0.000i
imrec01 = im2uint8(mat2gray(imrec01));
imrec02 = real(ifft2(exp(1i*(ifftshift(angle(Fib2))))));
imrec02 = im2uint8(mat2gray(imrec02));
imrec03 = real(ifft2(exp(1i*(ifftshift(angle(Fib3))))));
imrec03 = im2uint8(mat2gray(imrec03));
imrec04 = real(ifft2(exp(1i*(ifftshift(angle(Fib4))))));
imrec04 = im2uint8(mat2gray(imrec04));
figure,
subplot(221),imshow(imrec01,[]), title('Recontruccion luego de kernel 1')
subplot(222),imshow(imrec02,[]), title('Recontruccion luego de kernel 2')
subplot(223),imshow(imrec03,[]), title('Recontruccion luego de kernel 3')
subplot(224),imshow(imrec04,[]), title('Recontruccion luego de kernel 4')
```

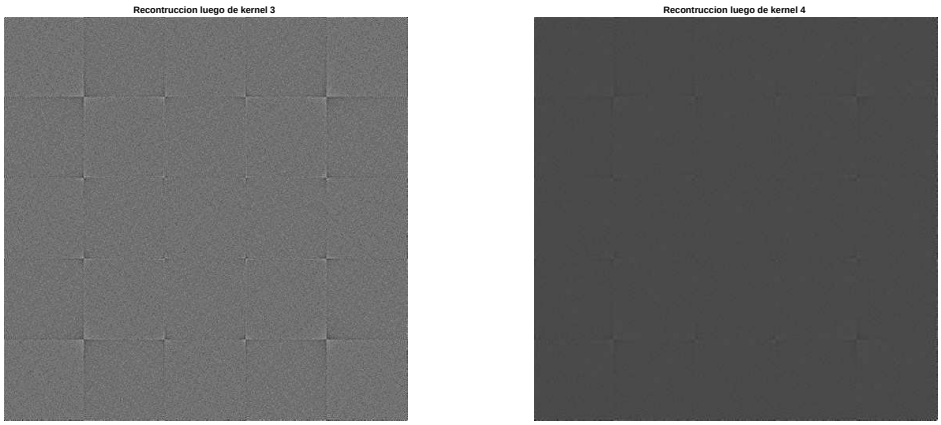
d. Uno de los filtros a utilizar para la detección de bordes es el filtro laplaciano. Utilizar la función **fspecial** para generar un filtro laplaciano con $\alpha = 0{,}5$. Realice el filtrado en el dominio de la frecuencia utilizando para ello la propiedad de Fourier $f(x,y) * g(x,y) \leftrightarrow F(u,v) \cdot G(u,v)$. Realizar un zero-padding al filtro y la imagen para poder realizar la multiplicación en el dominio de la frecuencia; asimismo, considerar la imagen resultante de la convolución con el kernel 4 del item anterior. Mostrar en una sola gráfica el resultado de la reconstrucción utilizando solo la fase de $F(u,v) \cdot G(u,v)$ y la reconstrucción completa utilizando magnitud y fase. Comentar sobre las diferencias presentes y su relación con la reconstrucción realizada en el ítem anterior.

Solución: Se observa en la Figura 7 que la reconstrucción utilizando solo la fase es similar a la reconstrucción vista en el ítem anterior. Esto debido a que la información en fase no se modifica con el filtro laplaciano. Sin embargo, se observa que en la reconstrucción completa solo se aprecian los bordes difuminados de la imagen y esto es debido a que el filtro laplaciano solo detecta variaciones de intensidad por lo que zonas homogeneas no presentan valores de magnitud considerables. También se observa presencia de ruido y es debido a que el filtro gaussiano no puede eliminar fuertes niveles de ruido sin perjudicar la textura de la imagen.

```
%generar filtro laplaciano con alpha=0.5
lapfilt = fspecial('laplacian',0.5);
[M,N] = size(ib4); %A partir de la imagen, se extrae el numero de ... filas en la variable M y el numero de columnas en la variable N.
FouI = fft2(double(ib4), M+2, N+2);
Hlap = fft2(double(lapfilt), M+2, N+2); %zero-padding del filtro se ... realiza aumentando las muestras de la transformada de Fourier.
F_fH = Hlap.*FouI;
ffi1 = ifft2(F_fH);
ffi2 = real(ifft2(exp(1i*angle(F_fH)))); %reconstruccion con fase
figure,
subplot(121),imagesc(ffi1), colormap gray, title('Reconstruccion ... completa')
subplot(122),imshow(im2uint8(mat2gray(ffi2)),[100 255]), colormap ... gray, title('Reconstruccion solo con fase')
```

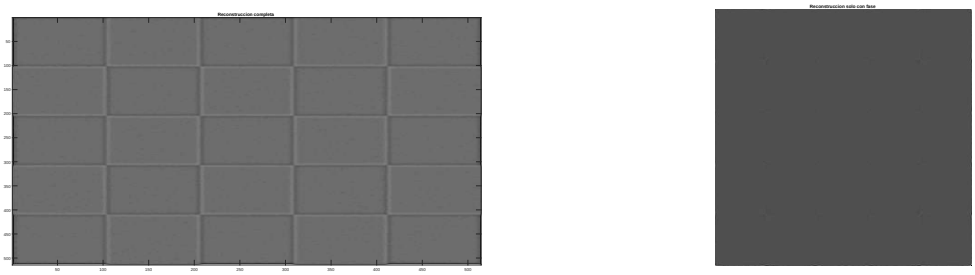


(a)



(b)

Figura 6: Reconstrucción de imágenes utilizando solo información en fase. (a) Reconstrucción luego de utilizar kernel de 3×3, $\sigma^2 = 1$ (izq.) y $\sigma^2 = 2$ (der.) (b) Reconstrucción luego de utilizar kernel de 7×7, $\sigma^2 = 1$ (izq.) y $\sigma^2 = 2$ (der.)



(a) (b)

Figura 7: Reconstrucción de imágenes con filtro laplaciano, (a) Reconstrucción utilizando espectro de magnitud y fase. (b) Reconstrucción utilizando espectro de fase.

e. Otro filtro para la detección de bordes es el detector Marr-Hildreth definido por:

$$\nabla^2 g(x,y) = (\frac{x^2+y^2-2\sigma^2}{\sigma^4})e^{-(\frac{x^2+y^2}{2\sigma^2})}$$

Realice el filtrado en el dominio de la frecuencia utilizando para ello la propiedad de Fourier $f(x,y) * g(x,y) \leftrightarrow F(u,v) \cdot G(u,v)$. Realizar un zero-padding al filtro y la imagen para poder realizar la multiplicación en el dominio de la frecuencia; asimismo, considerar la imagen resultante de la convolución con el kernel 4 del item anterior. Mostrar en una gráfica el resultado de la reconstrucción de $F(u,v) \cdot G(u,v)$. Comentar sobre las diferencias con el filtro laplaciano del ítem anterior. Utilizar **meshgrid, exp, fft2, ifft2**

Solución: Se observa en la Figura 8 que los bordes del tablero de ajedrez son detectados sin error. Se observa también que debido a que el filtro gaussiano no eliminó todo el ruido presente, se detectan como bordes algunos manchas en el tablero. Debido a la imagen, ambos detectores presentan la misma respuesta. Las diferencias son sutiles y se muestran al momento de realizar la binarización para extraer solo los bordes.

```
% Definir las coordenadas para el filtro de orden 11x11
[x,y]=meshgrid(-5:5,-5:5);
s2 = 1;
LoG = ((x.^2+y.^2-2*s2)/(s2^2)).*exp(-(x.^2+y.^2)/(2*s2));
[M,N] = size(ib4); %A partir de la imagen, se extrae el numero de ...
    filas en la variable M y el numero de columnas en la variable N.
FI = fft2(double(ib4), M+10, N+10); %zero-padding de la imagen se ...
    realiza aumentando las muestras de la transformada de Fourier.
Hlog = fft2(double(LoG), M+10, N+10); %zero-padding del filtro se ...
    realiza aumentando las muestras de la transformada de Fourier.
F_fH2 = Hlog.*FI;
ffi1LoG = ifft2(F_fH2);
ffi2LoG = real(ifft2(exp(1i*angle(F_fH))));%reconstruccion ...
    utilizando solo fase
figure,
imagesc(ffi1LoG), colormap gray, title('Reconstruccion completa')
```

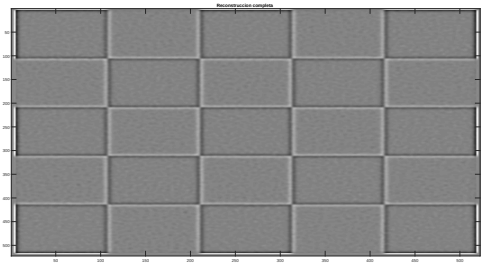


Figura 8: Detector Marr-Hildreth.

4. **Highboost Filtering y Segmentación por histograma**

El método **Highboost Filtering** puede ser visto en frecuencia de la siguiente manera:

$$g(x,y) = \mathcal{F}^{-1}\{[k_1 + k_2 \cdot H_{HP}(u,v)] \cdot F(u,v)\}$$

a) Leer la imagen **peppers.png**[5]. Calcular su transformada de Fourier 2D utilizando el tamaño de la imagen como número de muestras en frecuencia (M, N) y mostrar en una gráfica su espectro de magnitud y fase.

b) Utilizando la función **fspecial** definir un filtro pasa altos gaussiano con $\mu = 0$ y $\sigma^2 = 1$. Realizar un zero-padding a dicho filtro y calcular su transformada de Fourier. Luego, calcular $g(x, y)$ utilizando la ecuación inicial. Utilizar funciones **fft2, ifft2, fftshift, ifftshift, abs, angle**. Los valores de k serán $k_1 = 2$ y $k_2 = 5$. El tamaño del kernel del filtro sera de 5×5. ¿La imagen $g(x, y)$ presenta mejoras en los bordes?

c) Para diferentes valores de $k_2 = [1,\ 5,\ 10,\ 20]$ y $k_1 = 1$ calcular $g(x, y)$ y su transformada de Fourier. Explicar utilizando los espectros de magnitud y fase que es lo que ocurre al incrementar el valor de k_2. Utilizar funciones **fft2, ifft2, fftshift, ifftshift, abs, angle, exp**

d) Ahora incremente pruebe diversos valores de $k_1 = [1,\ 5,\ 10,\ 20]$ para $k_2 = 5$. Explicar utilizando los espectros de magnitud y fase que es lo que ocurre al incrementar el valor de k_1.

e) Calcular el histograma de la imagen resultante de aplicar el filtro highboost con $k_1 = 1$ y $k_2 = 5$. Seleccionar un valor de umbral en la gráfica del histograma teniendo en consideración la forma del histograma. Utilizando dicho umbral convierta la imagen resultante en una imagen binaria. Mostrar en una gráfica la imagen resultante del filtrado y la binarización. Utilizar las funciones **hist, im2bw**.

f) Utilizar la función **graythresh** para calcular un nuevo valor umbral. Binarizar la misma imagen del item anterior. ¿Qué método de segmentación utiliza la función **graythresh** (ver documentación)? ¿Qué diferencias encuentra en la binarización respecto al ítem anterior? Mostrar en una gráfica la imagen original y la binarización.

[5]La imagen está almacenada en la carpeta de instalación de MATLAB. Utilizar solo **imread('peppers.png')**