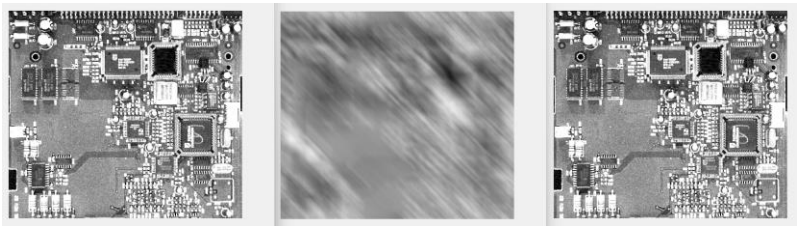
Práctica 8

Restauración de imágenes degradas o distorsionadas

Restaura una imagen movida y suavizada como la de la siguiente figura y calcula el ECM de restauración.



```
I0=imread('Fig1.14(a).jpg');
I=im2double(I0);
figure, imshow(I)
[M,N]=size(I);
F=fft2(I);
h1=fspecial('gaussian',7,20); % Crea un filtro gaussiano de tamaño 7x7 con una desviación estándar de 20.
H1=fft2(h1,M,N); % Calcula la transformada de Fourier del filtro gaussiano.
long=100;
ang=135;
h2=fspecial('motion',long,ang); % Crea un filtro de movimiento.
H2= fft2(h2,M,N); %  Calcula la transformada de Fourier del filtro de movimiento
G=H1.*H2.*F;
I1=real(ifft2(G));
figure,imshow(I1)

HR=(1./(H1.*H2)); % Calcula el filtro inverso combinando los inversos de los dos filtros aplicados anteriormente.
G_rest=HR.*G; % Aplica el filtro inverso a la imagen filtrada en el dominio de la frecuencia.
I2=real(ifft2(G_rest)); % Realiza la transformada inversa de Fourier para obtener la imagen restaurada I2.
figure,imshow(I2)
ECM=sum((I2-I).^2,"all"); % Calcula el error cuadrático medio (ECM).
```

ECM = 3.1011e-26

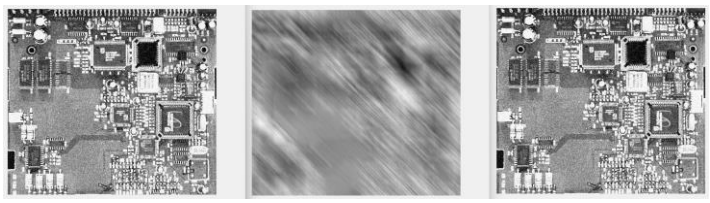
¿Qué método de restauración estamos usando? ¿por qué no es nulo el ECM?

Estamos utilizando un método de restauración basado en la deconvolución en el dominio de la frecuencia. No es nulo porque siempre hay una pérdida de información durante el proceso de convolución y su inversión. Además, la presencia de ruido en la imagen original contribuye al error en la restauración.

Prueba a variar el grado de suavizado, ¿varía significativamente el ECM? ¿Por qué?

Sí. Cuanto mayor sea el grado de suavizado, mayor será la pérdida de detalles finos en la imagen restaurada, lo que puede resultar en un mayor error en comparación con la imagen original. Por otro lado, un suavizado más suave podría preservar mejor los detalles, lo que podría resultar en un ECM meno

Añade ruido uniforme antes de desplazar una imagen y restáurala usando un filtro inverso, ¿se restaura sin ruido? ¿Por qué?

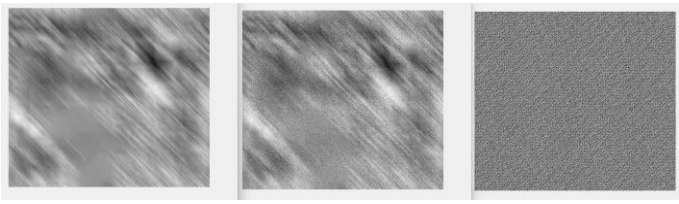


No, la imagen no se restaura sin ruido. De hecho, el uso de un filtro inverso en presencia de ruido suele amplificar el ruido en la imagen restaurada.

El ruido uniforme en el dominio espacial se transforma en ruido distribuido en todas las frecuencias en el dominio de la frecuencia. El filtro inverso amplifica este ruido, especialmente en las frecuencias donde el filtro original atenúa la señal.

```
%% 2.a)
I0=imread('Fig1.14(a).jpg');
I=im2double(I0);

% Añadir ruido uniforme a la imagen
ruido=0.1*randn(size(I));
B=I+ruido;
figure, imshow(B)

% Aplicar la transformada de Fourier a la imagen con ruido
F=fft2(B);

% Definir el filtro de movimiento
long=100;
ang=135;
h=fspecial('motion',long,ang);
[M,N]=size(I);

% Aplicar la transformada de Fourier al filtro de movimiento
H= fft2(h,M,N);

% Filtrar la imagen en el dominio de la frecuencia
G=H.*F;
I1=real(ifft2(G));
figure,imshow(I1)

% Restaurar la imagen usando un filtro inverso
HR=(1./H);
G_rest=HR.*G;
I2=real(ifft2(G_rest));
figure,imshow(I2)
```

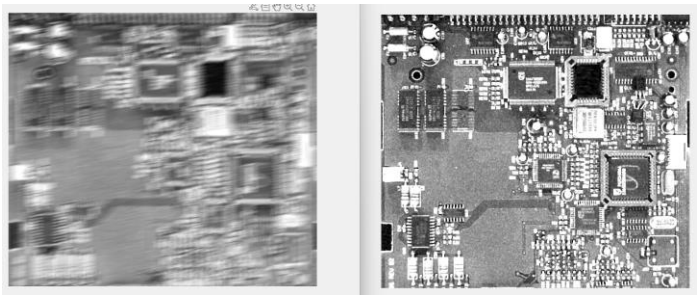
Y si añades ruido después de desplazar la imagen, ¿hay alguna diferencia con el caso anterior? ¿Por qué?



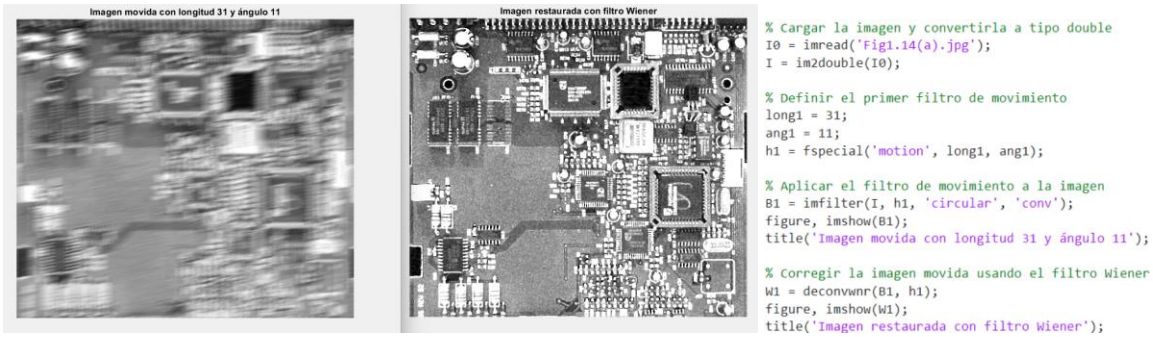
Sí, hay una diferencia significativa entre añadir el ruido antes y después de aplicar el filtro de movimiento. La diferencia principal radica en cómo el ruido interactúa con el proceso de filtrado y la posterior deconvolución.

```
%% 2.b)
I0=imread('Fig1.14(a).jpg');
I=im2double(I0);

% Aplicar la transformada de Fourier a la imagen original
F=fft2(I);

% Definir el filtro de movimiento
long=100;
ang=135;
h=fspecial('motion',long,ang);
[M,N]=size(I);

% Aplicar la transformada de Fourier al filtro de movimiento
H=fft2(h,M,N);

% Filtrar la imagen en el dominio de la frecuencia
G=H.*F;
I1=real(ifft2(G));
figure,imshow(I1)

% Añadir ruido gaussiano a la imagen filtrada
ruido=0.1*randn(size(I));
B=I1+ruido;
figure,imshow(B)

% Aplicar la transformada de Fourier a la imagen con ruido
F2=fft2(B);

% Restaurar la imagen usando un filtro inverso
HR=(1./H);
G_rest=HR.*F2;
I1=real(ifft2(G_rest));
figure,imshow(I1)
```

Distorsiona una imagen utilizando una función de distorsión que produzca el efecto de "imagen movida" y corrígela mediante el filtro Wiener.

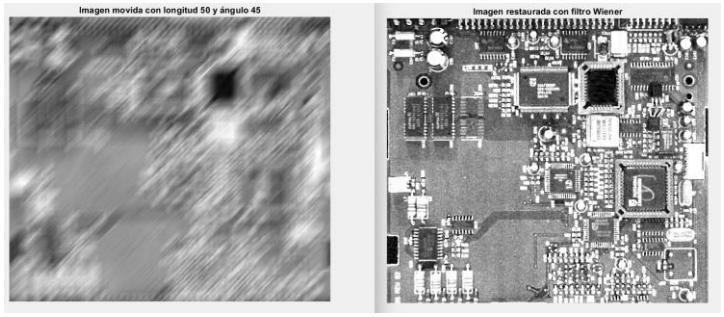


Prueba dos desplazamientos diferentes y comenta las diferencias si las hubiera.




```
% Cargar la imagen y convertirla a tipo double
I0 = imread('Fig1.14(a).jpg');
I = im2double(I0);

% Definir el primer filtro de movimiento
long1 = 31;
ang1 = 11;
h1 = fspecial('motion', long1, ang1);

% Aplicar el filtro de movimiento a la imagen
B1 = imfilter(I, h1, 'circular', 'conv');
figure, imshow(B1);
title('Imagen movida con longitud 31 y ángulo 11');

% Corregir la imagen movida usando el filtro Wiener
W1 = deconvwnr(B1, h1);
figure, imshow(W1);
title('Imagen restaurada con filtro Wiener');
```

Primer desplazamiento



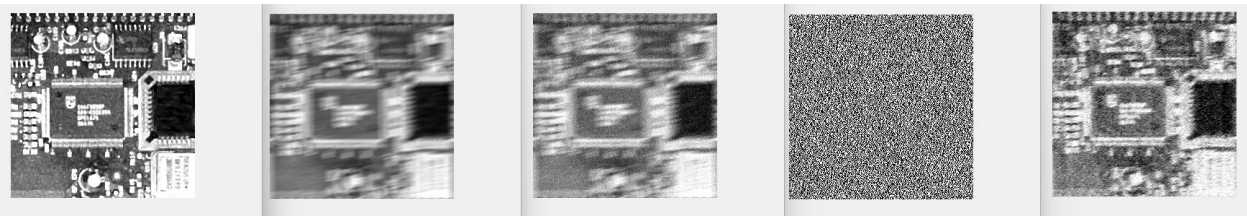

```
% Definir el segundo filtro de movimiento
long2 = 50;
ang2 = 45;
h2 = fspecial('motion', long2, ang2);

% Aplicar el filtro de movimiento a la imagen
B2 = imfilter(I, h2, 'circular', 'conv');
figure, imshow(B2);
title('Imagen movida con longitud 50 y ángulo 45');

% Corregir la imagen movida usando el filtro Wiener
W2 = deconvwnr(B2, h2);
figure, imshow(W2);
title('Imagen restaurada con filtro Wiener');
```

Segundo desplazamiento

- **Primer desplazamiento (longitud 31, ángulo 11):**
    - **Imagen movida:** La imagen presenta un desenfoque notable pero moderado en una dirección específica.
    - **Imagen restaurada:** La restauración es bastante efectiva, recuperando detalles y reduciendo el desenfoque. Sin embargo, pueden quedar algunos artefactos y ruido.
- **Segundo desplazamiento (longitud 50, ángulo 45):**
    - **Imagen movida:** La imagen presenta un desenfoque más pronunciado y diagonal, debido a la mayor longitud del desplazamiento y el ángulo.

- **Imagen restaurada:** La restauración es menos efectiva en comparación con el primer desplazamiento. El filtro Wiener recupera algunos detalles, pero el mayor desenfoque inicial resulta en más artefactos y ruido residual.

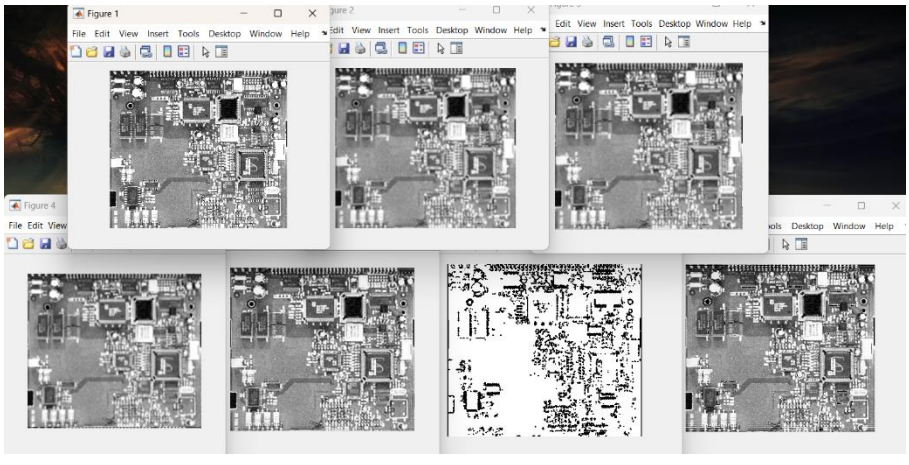
Restaura una imagen movida que tiene añadido además ruido gaussiano utilizando primero el filtro inverso y después el filtro de Wiener. ¿Ambos restauran bien la imagen? ¿Por qué?



El filtro inverso no restauraría bien la imagen debido a su alta El filtro de Wiener sin NSR ofrece una mejor restauración que el filtro inverso, pero no es óptimo porque no tiene en cuenta la relación señal-ruido.

```
I=imread('Fig1.14(a).jpg');
I=I(10+[1:256],222+[1:256],:);
%Cogemos una matriz vinculada
%cuadrada => debido a una limitación de
%de deconvwnr
imshow(I)

% Definir el filtro de movimiento
long=20;
ang=12;
h=fspecial('motion',long,ang);

% Aplicar el filtro de movimiento
J=imfilter(I,h,'circular','conv');
figure,imshow(J);


% Añadir ruido gaussiano
ruido=0.1*randn(size(J));
B=imadd(J,im2uint8(ruido));
figure,imshow(B)

% Restaurar con filtro de Wiener sin NSR
W=deconvwnr(B,h);
figure,imshow(W)

% Calcular NSR y restaurar con filtro de Wiener
NSR=sum(ruido(:).^2)/sum(im2double(I(:)).^2);
W1=deconvwnr(B,h,NSR);
figure,imshow(W1)
```

Restaura una imagen que ha sido suavizada con un filtro gaussiano utilizando el método de desconvolución a ciegas (algoritmo de Lucy-Richardson). ¿Cuál es la matriz/función de degradación?



```
I=imread('Fig1.14(a).jpg');
imshow(I)
h=fspecial('gaussian',7,10);
B=imfilter(I,h,'symmetric','conv');
figure,imshow(B);
h1=ones(size(h)-4);
%1ª restauración
[J1 P1]=deconvblind(B,h1);
figure, imshow(J1);
%2ª restauración
h2=padarray(h1,[4:4],'replicate','both');
[J2 P2]=deconvblind(B,h2);
figure, imshow(J2);
%3ª restauración
h3=padarray(h1,[2 2],'replicate','both');
[J3 P3]=deconvblind(B,h3);
figure, imshow(J3);
peso=edge(I,'sobel',.3);
ee=strel('disk',2);
peso=1-double(imdilate(peso,ee));
peso(:,[1:3 end-[0:2]])=0;
figure,imshow(peso)
[J P]=deconvblind(B,h3,30,[],peso);
figure,imshow(J)
```

La matriz/función de degradación utilizada en este ejercicio es un filtro gaussiano. Este filtro gaussiano de tamaño 7x7 y desviación estándar 10 es la función de degradación que se aplica a la imagen original.

¿ha sido necesario conocer el valor concreto de dicha matriz para restaurar la imagen? ¿por qué?

No, no ha sido necesario conocer el valor concreto de la matriz de degradación exacta **h** para restaurar la imagen en el proceso de deconvolución ciega. La deconvolución ciega es un método que intenta estimar simultáneamente la imagen original y la función de degradación (PSF) a partir de la imagen degradada.