



**UAI**

UNIVERSIDAD ADOLFO IBÁÑEZ  
FACULTAD DE INGENIERÍA Y CIENCIAS



**iUAI**  
UNIVERSIDAD ADOLFO IBÁÑEZ  
FACULTAD DE INGENIERIA Y CIENCIAS

MDS<sup>2019</sup> PROCESAMIENTO  
DE IMÁGENES

RESTAURACIÓN + DEGRADACIÓN

Miguel Carrasco  
[miguel.carrasco@uai.cl](mailto:miguel.carrasco@uai.cl)  
1er Semestre 2020

### ▶ Dominio Espacial

- Filtros de orden estadísticos
  - Mediana
  - Moda
  - Max-min
  - Punto medio
  - Promedio alfa-acotado

- Filtros Adaptivos

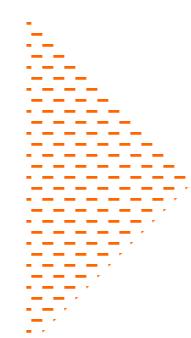
- Ruido Local
- Mediana Adaptiva

- Filtros lineales

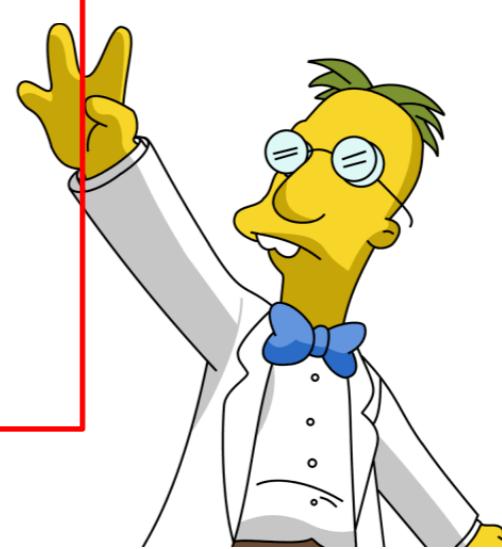
- Media
- Media geométrica
- Media armónica
- Media contra-armónica

### ■ Dominio Frecuencial

- Filtro Ideal
- Filtro Gaussiano
- Filtro Butterworth



- Parabanda
- Pasabanda
- Puntual
- Puntual óptimo



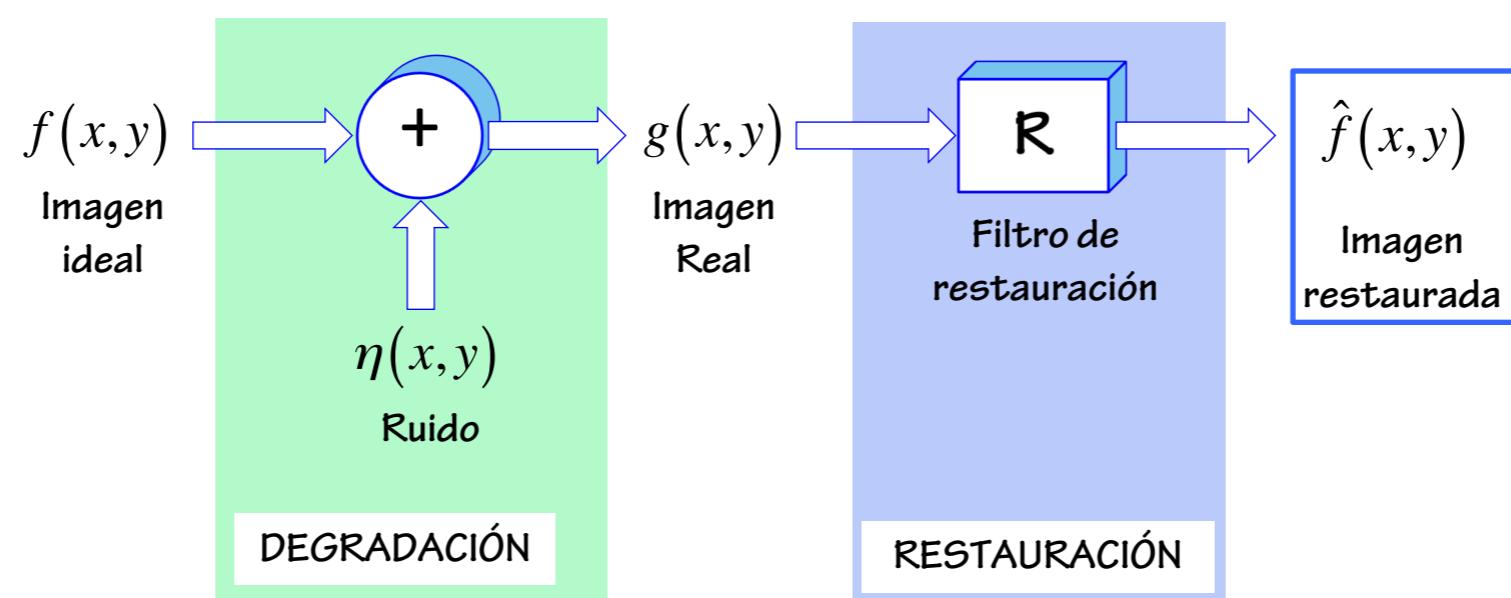
## ▶ Dominio Frecuencial

- Parabandas
- Pasabandas
- Puntuales
- Puntuales óptimos

### Definición

Los filtros en el dominio del espacio emplean el análisis en la frecuencia para mejorar la imagen, reducir o eliminar el ruido.

Para emplear este tipo de filtros es necesario transformar la imagen con la Transformada Discreta de Fourier



## ▶ Filtros en la frecuencia (Parabandas)

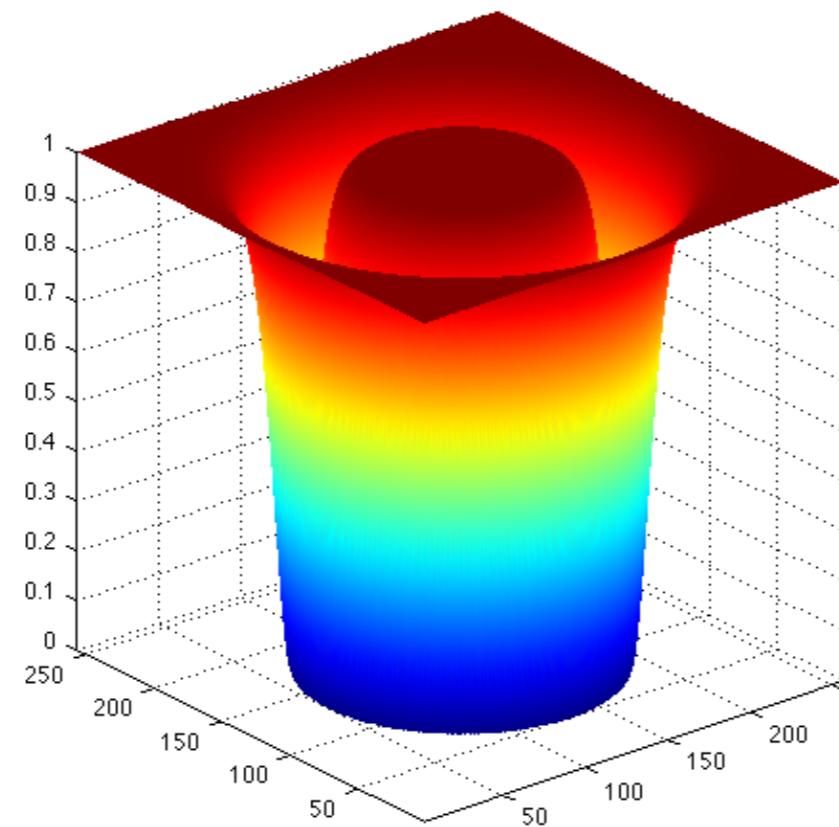
Parabandas

Pasabanda

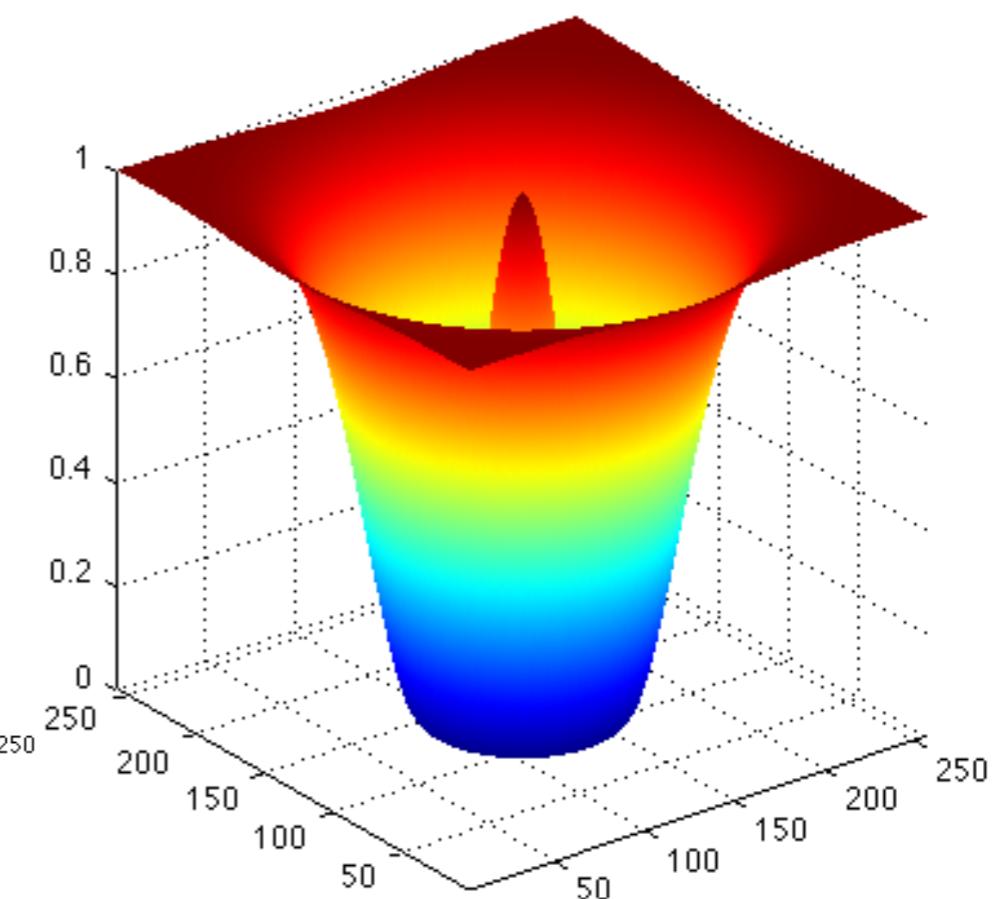
Puntuales

Puntuales - óptimo

- Bloquea o limita el paso de cierta frecuencia empleando una combinación de un filtro pasa alto y un filtro pasa bajo.



Parabanda Butterworth



Parabanda Gaussiano

## ▶ Filtros en la frecuencia (Parabandas)

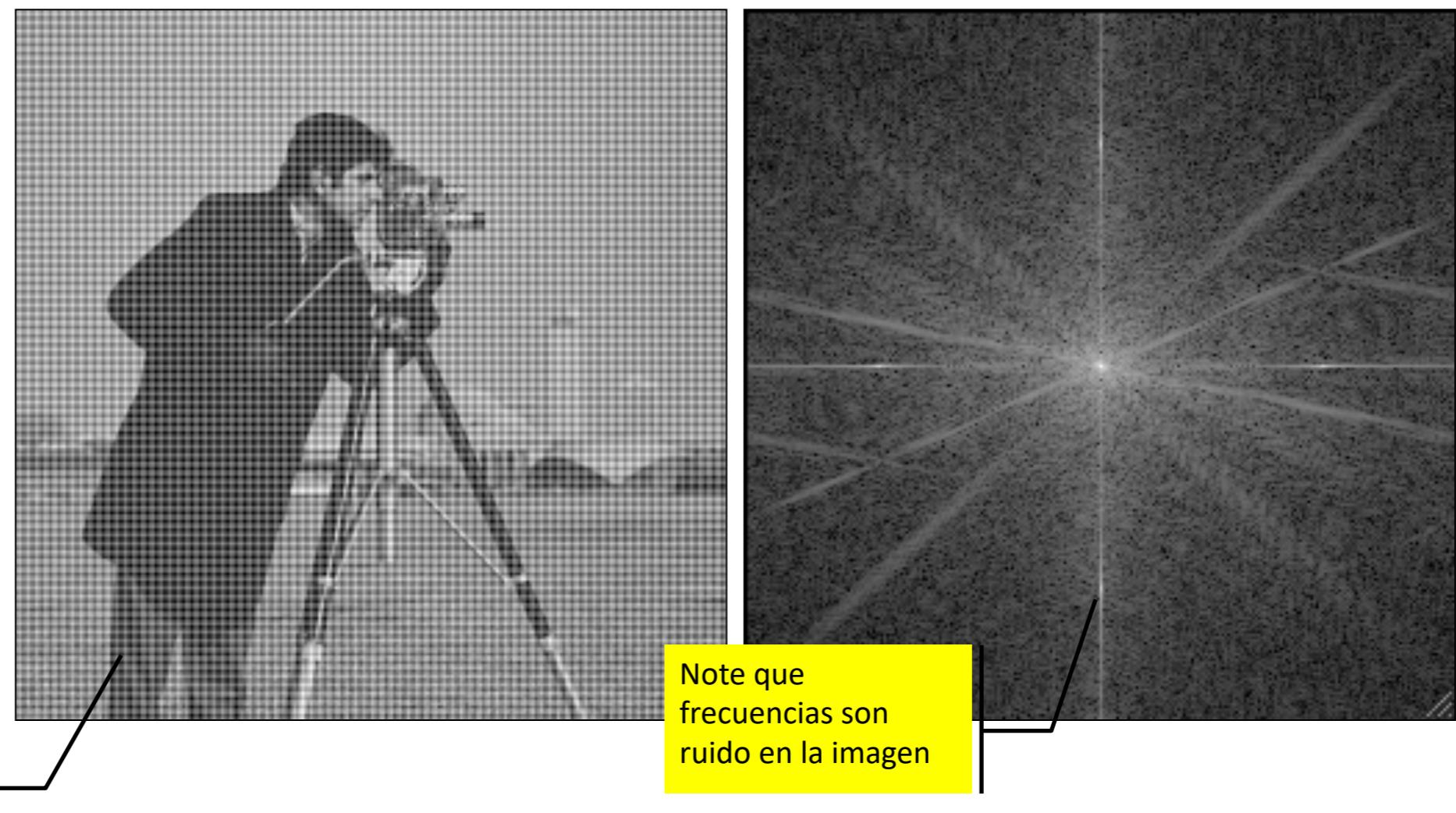
Parabandas

Pasabanda

Puntuales

Puntuales -  
óptimo

- Bloquea o limita el paso de cierta frecuencia empleando una combinación de un filtro pasa alto y un filtro pasa bajo.



## ▶ Filtros en la frecuencia (Parabandas)

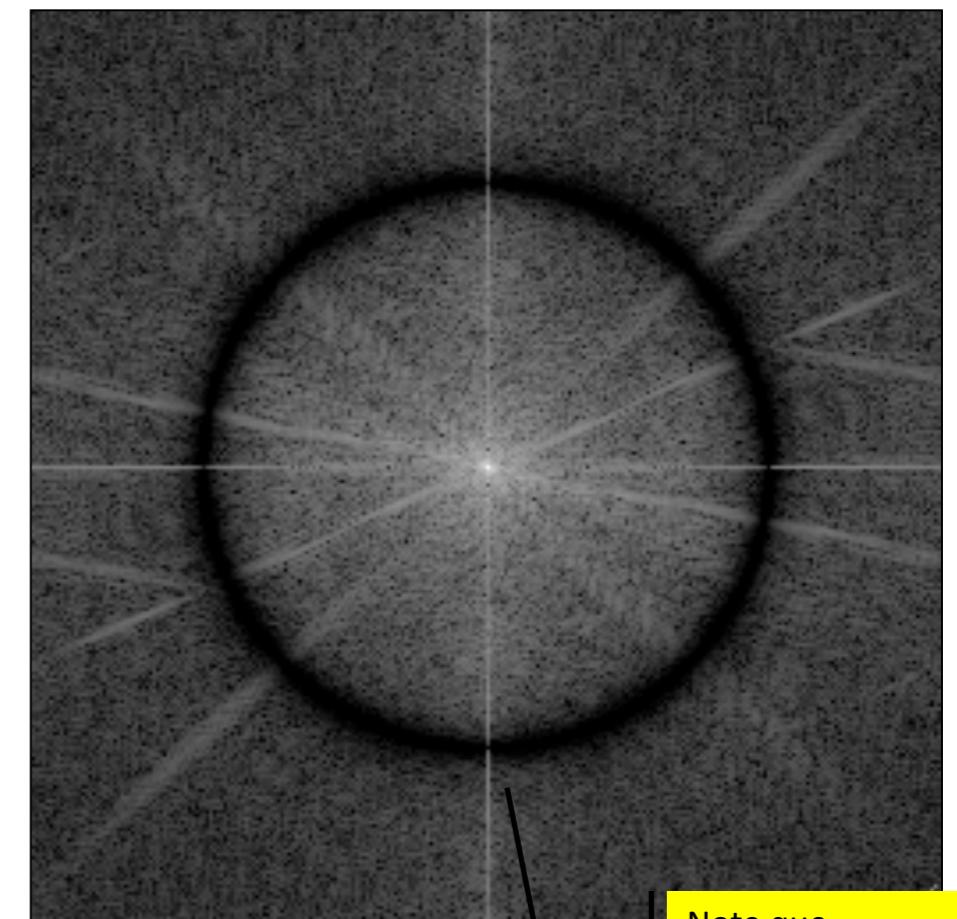
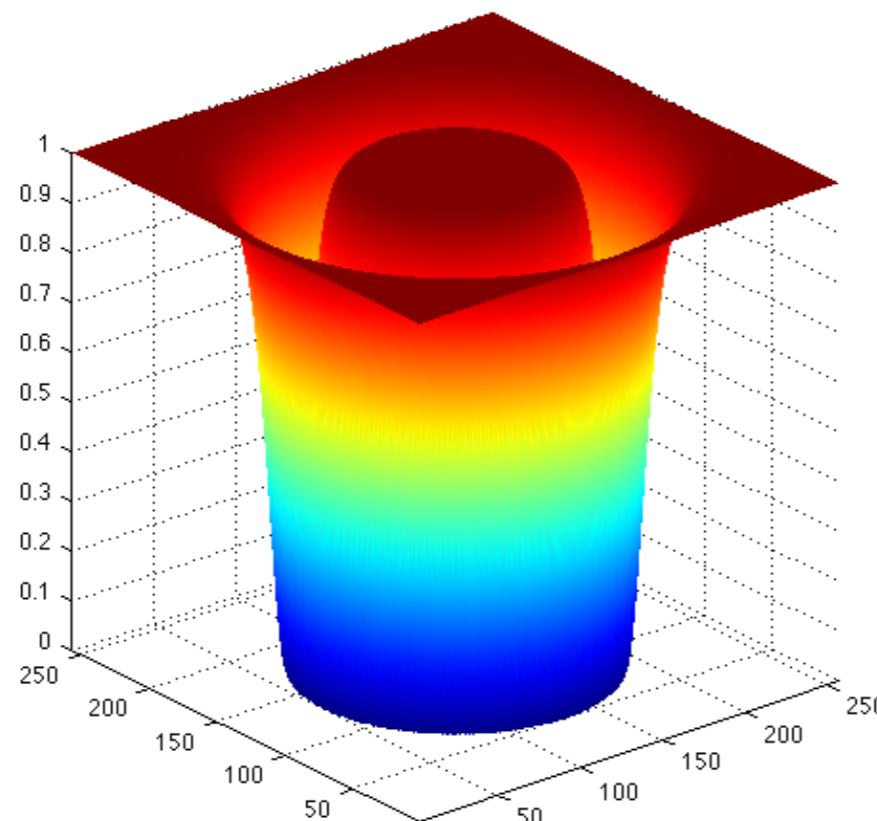
Parabandas

Pasabanda

Puntuales

Puntuales -  
óptimo

- Bloquea o limita el paso de cierta frecuencia empleando una combinación de un filtro pasa alto y un filtro pasa bajo.



Note que  
frecuencias son  
ruido en la imagen

## ▶ Filtros en la frecuencia (Parabandas)

Parabandas

Pasabanda

Puntuales

Puntuales -  
óptimo

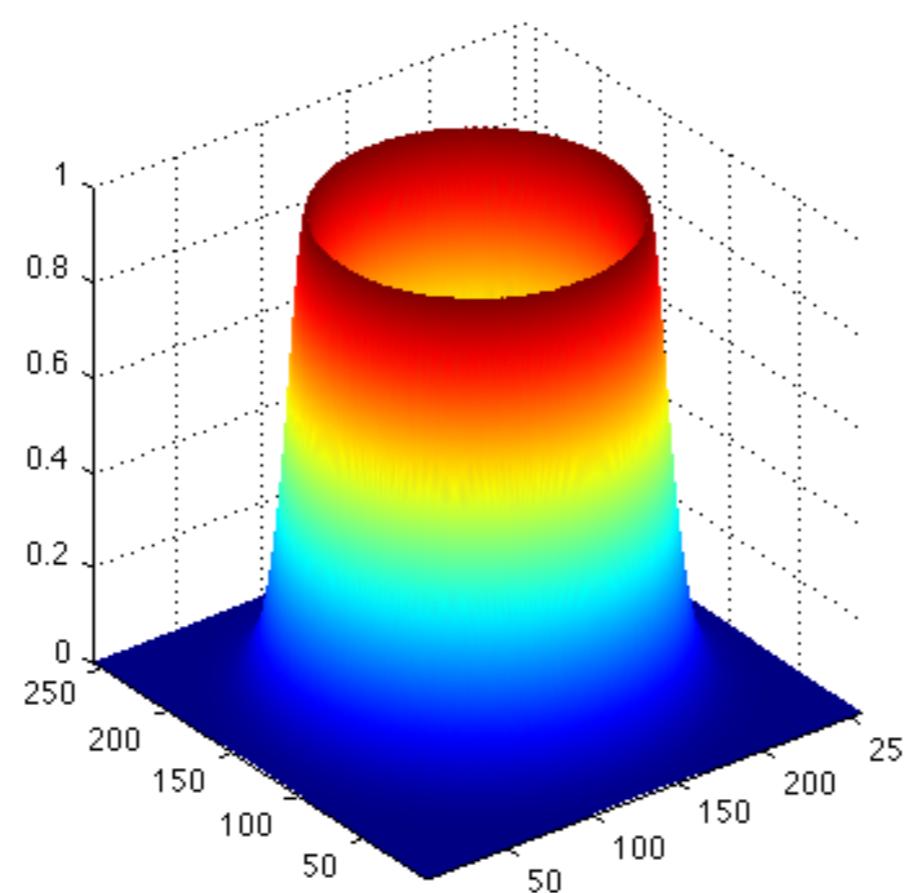
- Bloquea o limita el paso de cierta frecuencia empleando una combinación de un filtro pasa alto y un filtro pasa bajo.



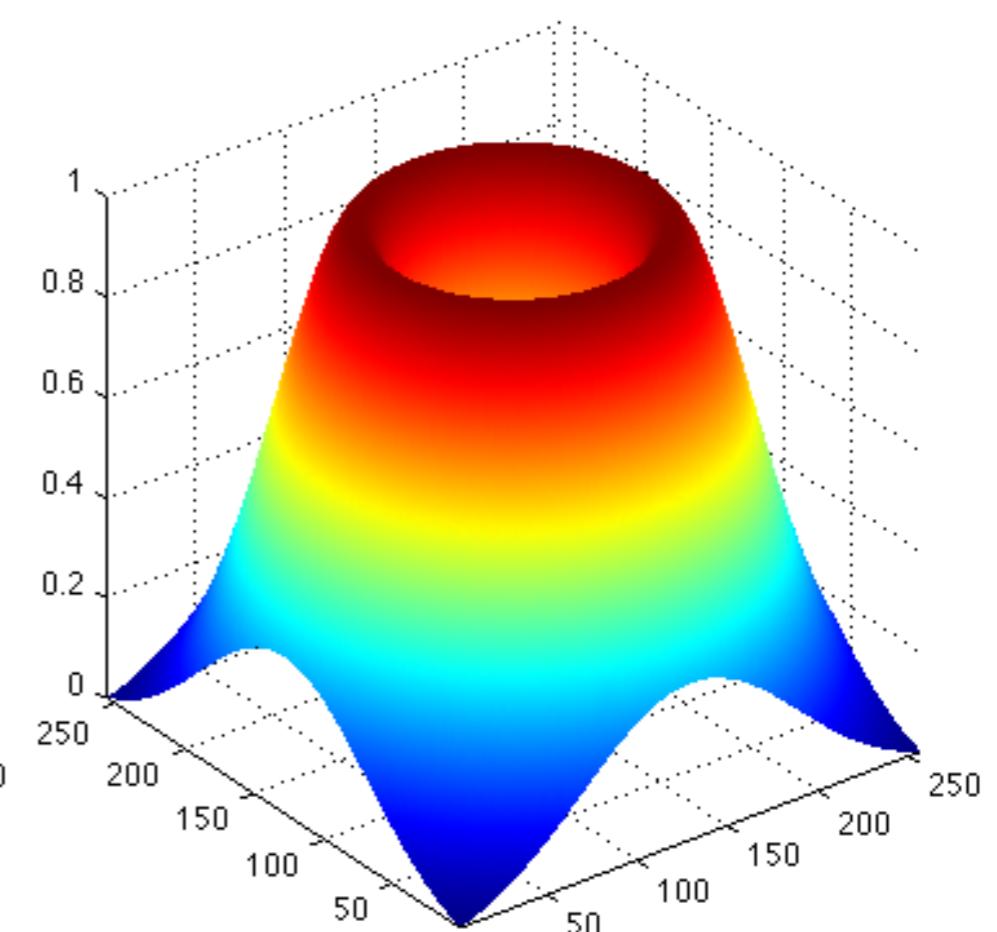
## ▶ Filtros en la frecuencia (Pasabandas)

- Parabandas
- Pasabanda
- Puntuales
- Puntuales - óptimo

- Permite el paso de un grupo de frecuencias específicas.



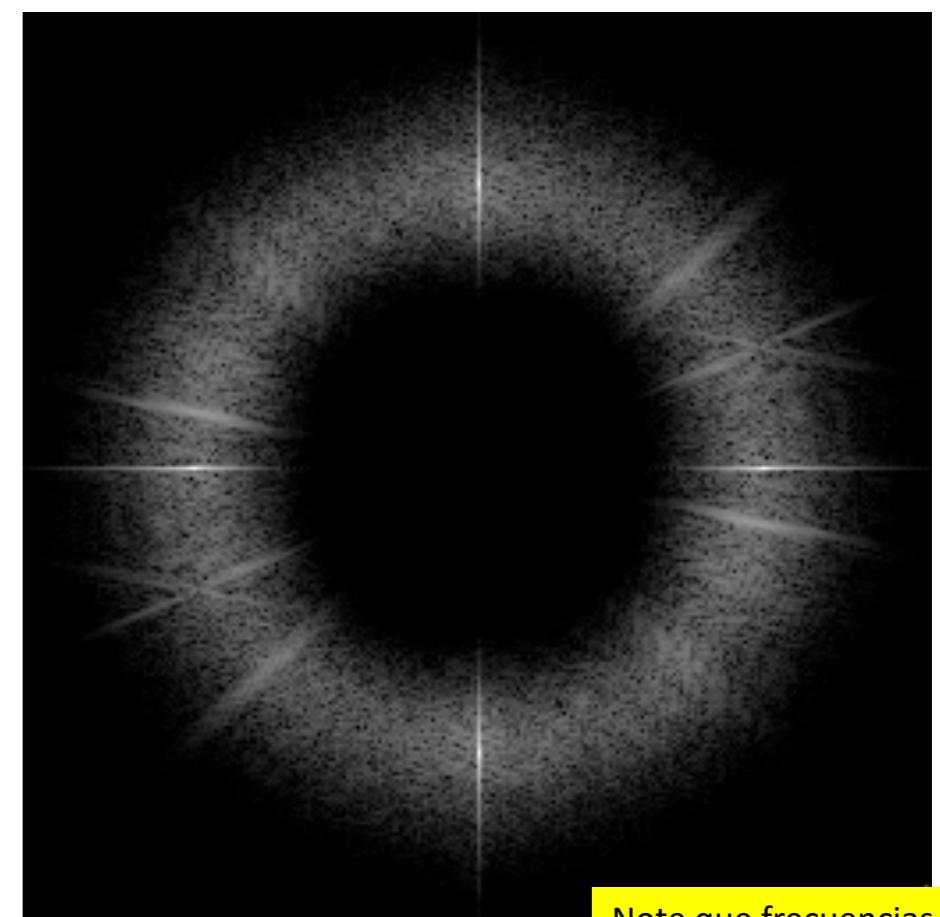
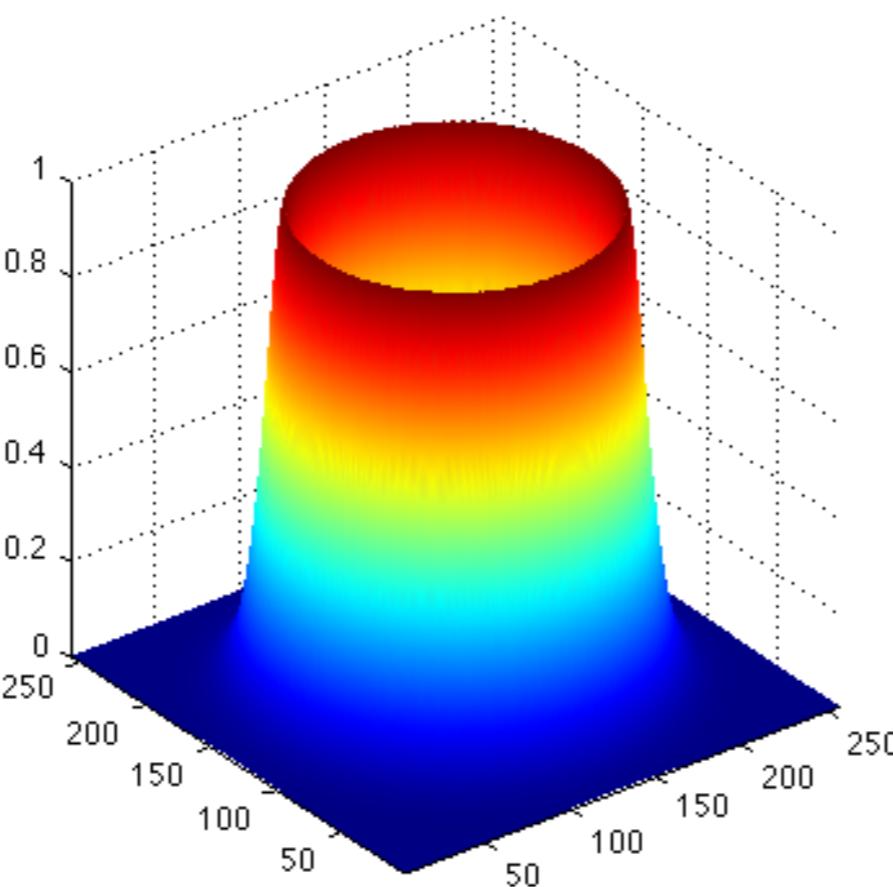
Pasabanda Butterworth



Pasabanda Gaussiano

## ▶ Filtros en la frecuencia (Pasabandas)

- Parabandas
- Pasabanda**
- Puntuales
- Puntuales - óptimo



Note que las frecuencias que el filtro dejó pasar corresponden al ruido

## ▶ Filtros en la frecuencia (Pasabandas)

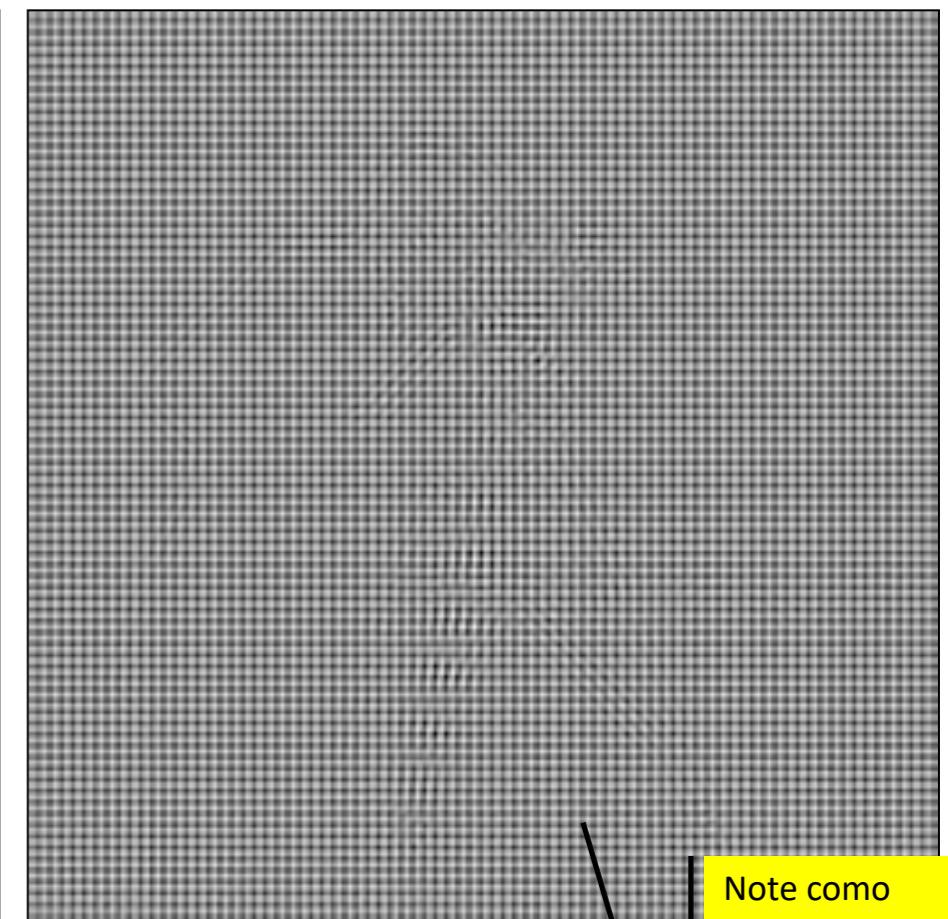
Parabandas

Pasabanda

Puntuales

Puntuales -  
óptimo

- Permite el paso de un grupo de frecuencias específicas. En el ejemplo, vemos que solo puede pasar el ruido



Note como  
obtenemos el  
ruido de la  
imagen

## ▶ Filtros en la frecuencia (Puntuales)

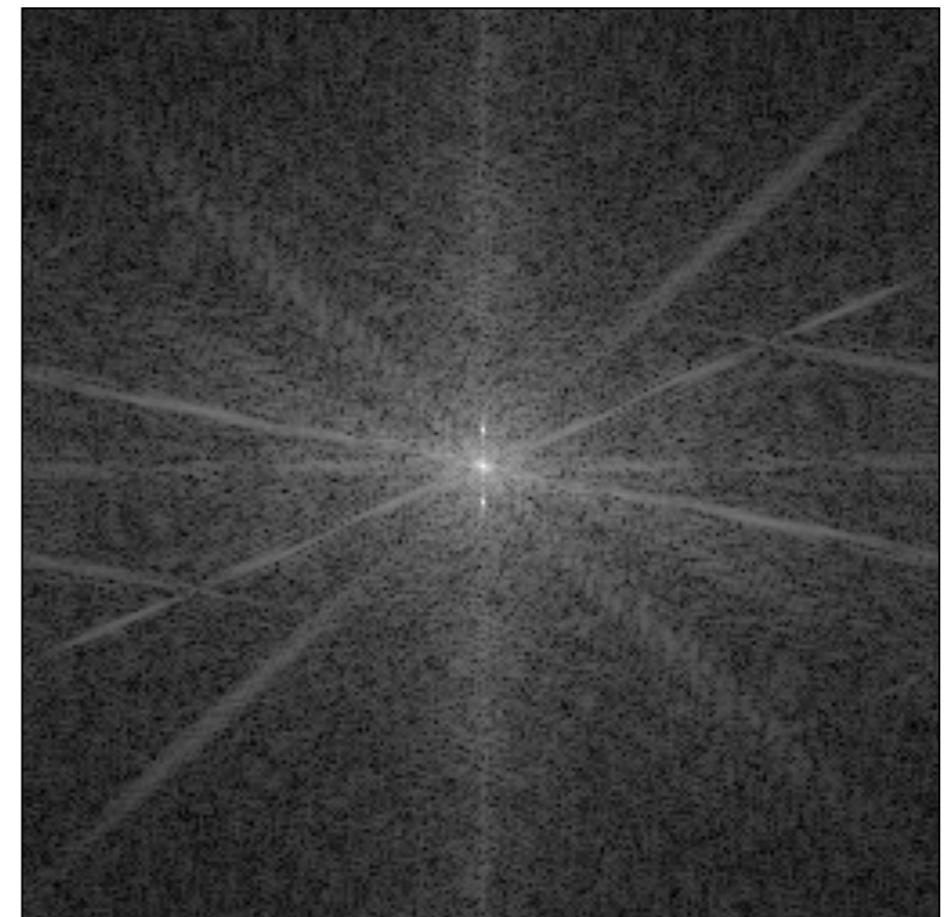
Parabandas

Pasabanda

**Puntuales**

Puntuales - óptimo

- Bloque el paso de un grupo de frecuencias específicas. Es empleado cuando conocemos las frecuencias en el espectro que debemos bloquear.



## ▶ Filtros en la frecuencia (Puntuales)

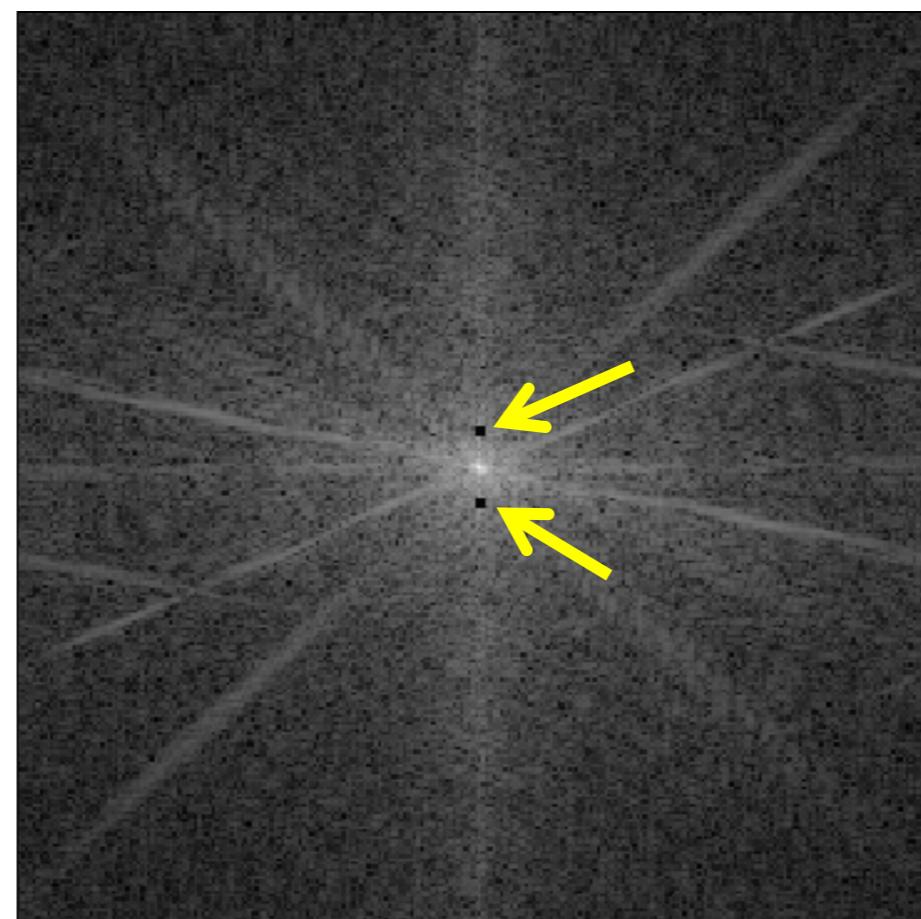
Parabandas

Pasabanda

**Puntuales**

Puntuales - óptimo

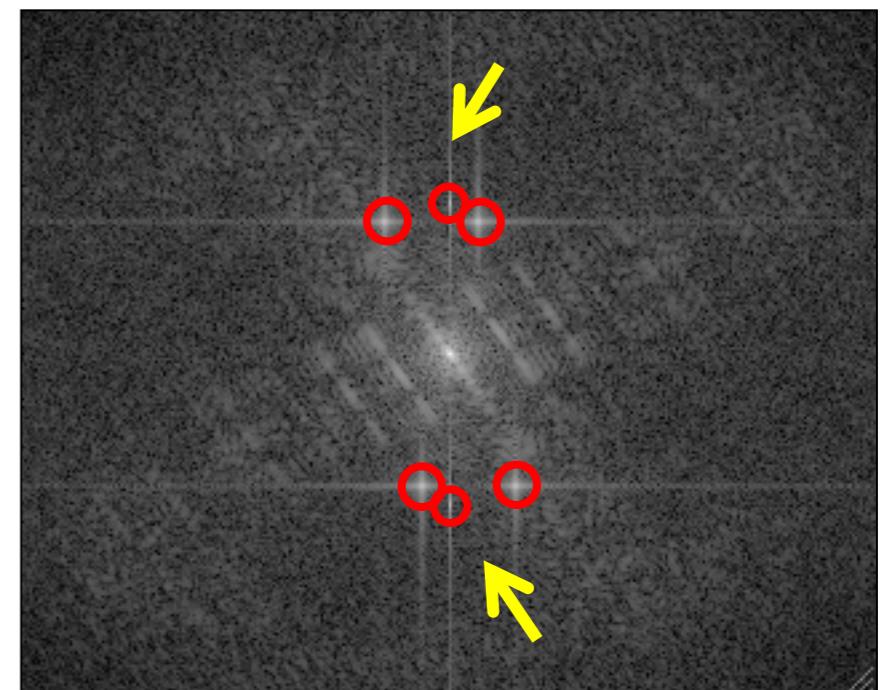
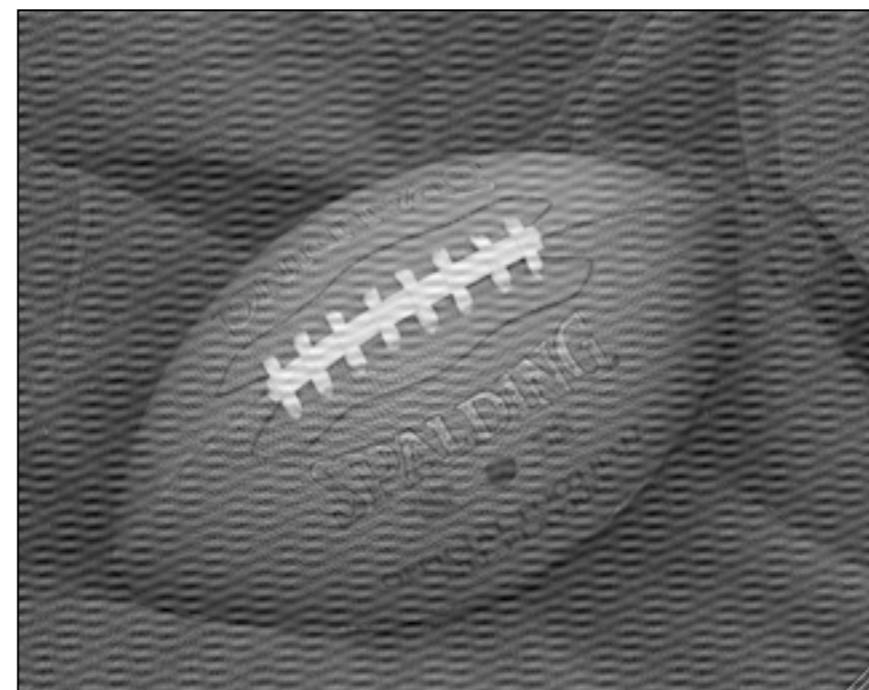
- Bloque el paso de un grupo de frecuencias específicas. Es empleado cuando conocemos las frecuencias en el espectro que debemos bloquear.



## ▶ Filtros en la frecuencia (Puntuales óptimos)



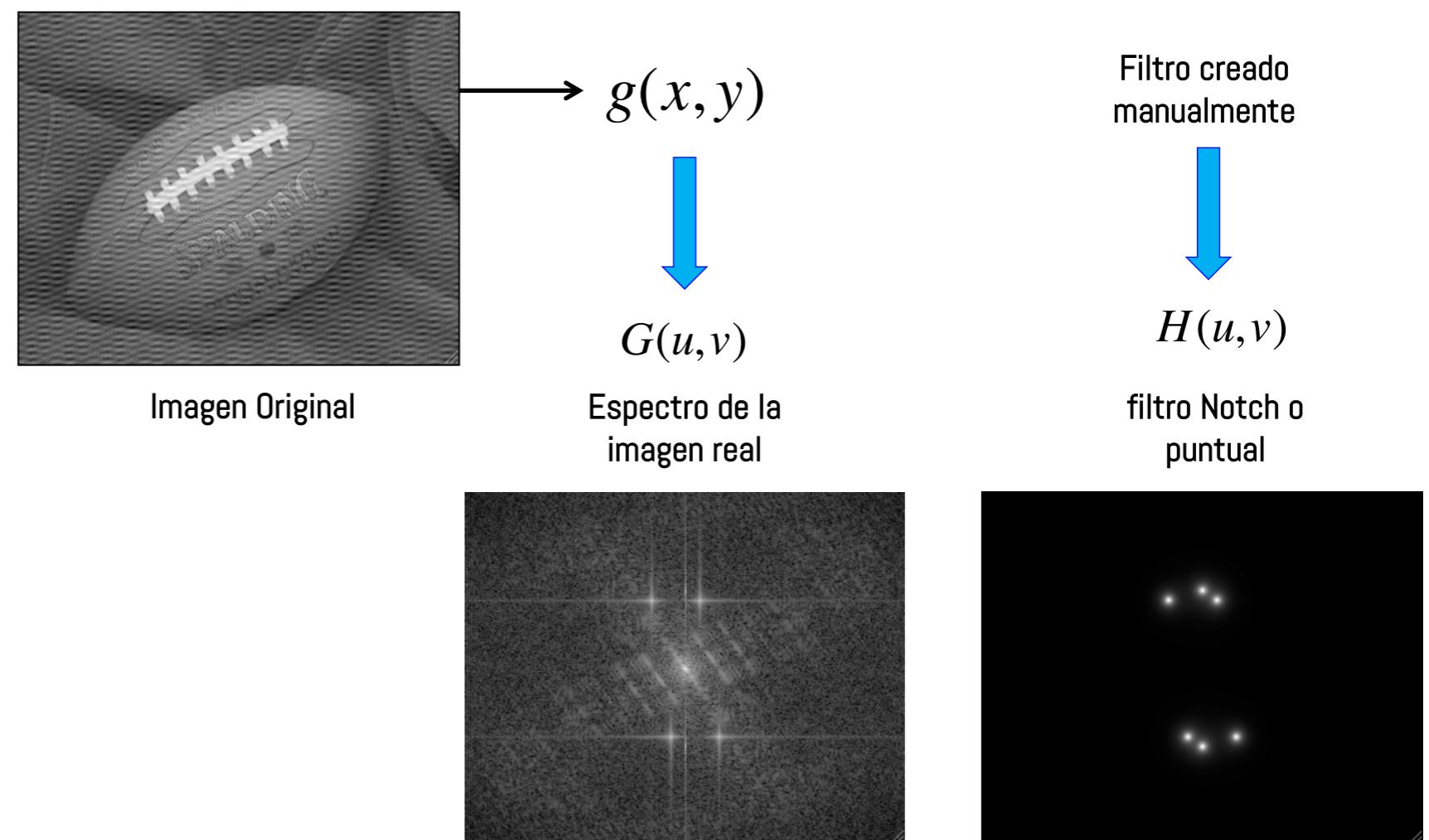
- En algunos casos el ruido puede alterar distintas frecuencias (ruido satelitales generado por los circuitos). Una forma de eliminarlo es bloquear dichas frecuencias.
- Existe otra opción que consiste en eliminar el ruido directamente empleando la señal de interferencia.



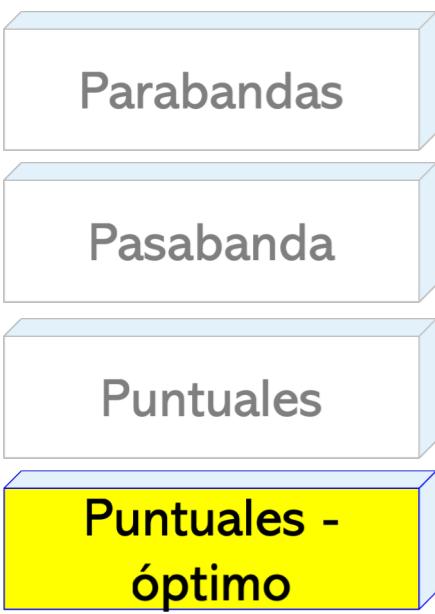
## ▶ Filtros en la frecuencia (Puntuales óptimos)



- El algoritmo requiere efectuar los siguientes pasos:
- 1ro) Construir un filtro puntual (pasabanda) que *permite el paso* únicamente de las *frecuencias de ruido*.



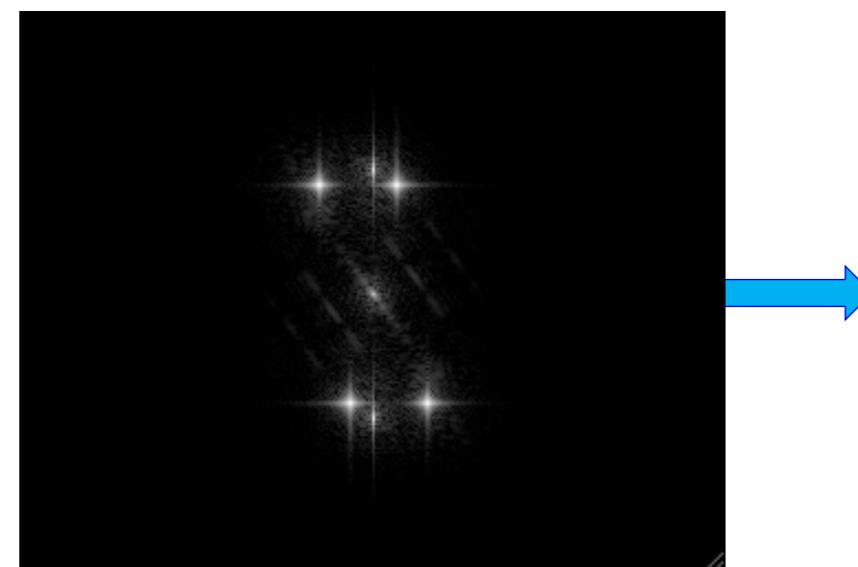
## ▶ Filtros en la frecuencia (Puntuales óptimos)



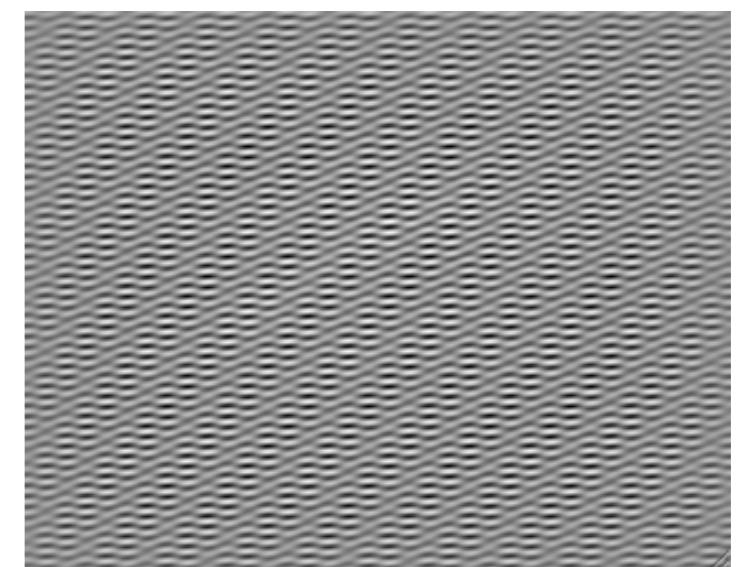
- El algoritmo requiere efectuar los siguientes pasos:
- 2do) Determinamos la inversa de Fourier de la multiplicación del *filtro* con el *espectro*.

$$H(u,v) \cdot G(u,v)$$

$$\eta(x,y) = FFT^{-1}(H(u,v) \cdot G(u,v))$$



Espectro  
Filtrado



Patrón de ruido  
(estimado)

## ▶ Filtros en la frecuencia (Puntuales óptimos)

Parabandas

Pasabanda

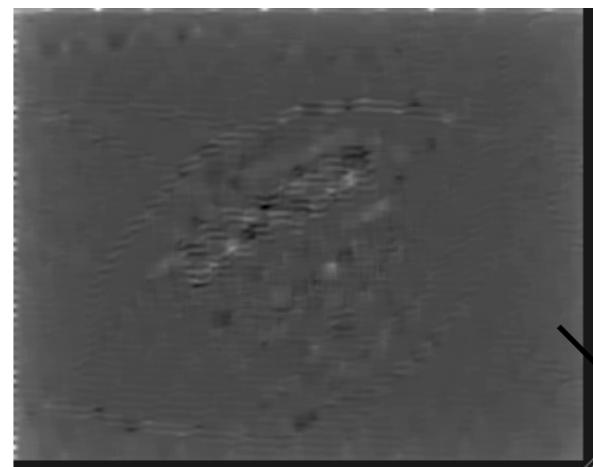
Puntuales

Puntuales -  
óptimo

- El algoritmo requiere efectuar los siguientes pasos:
  - 3ro) Determinar el factor de compensación del ruido el cual minimiza el ruido en la imagen original.

$$\omega(x,y) = \frac{\overline{g(x,y) \cdot \eta(x,y)} - \bar{g}(x,y) \cdot \bar{\eta}(x,y)}{\bar{\eta}^2(x,y) - \bar{\eta}^2(x,y)}$$

$$\omega(x,y) =$$



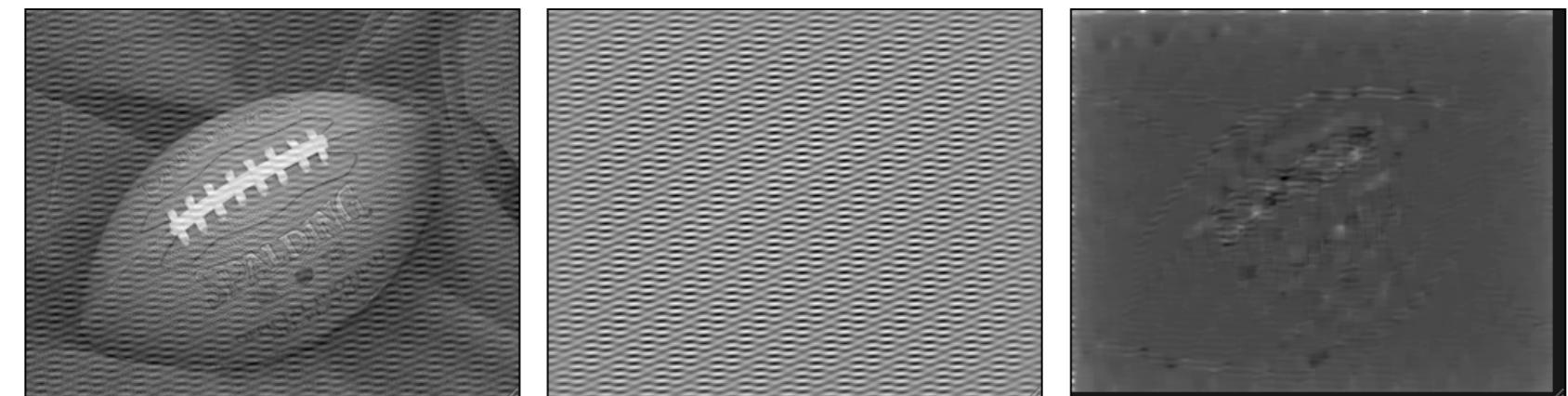
Resultado del  
proceso anterior

## ▶ Filtros en la frecuencia (Puntuales óptimos)



- El algoritmo requiere efectuar los siguientes pasos:
- 4to) Estimar la imagen libre de ruido con la siguiente fórmula:

$$\hat{f}(x, y) = g(x, y) - \omega(x, y) \cdot \eta(x, y)$$



$g(x, y)$

$\eta(x, y)$

$\omega(x, y)$

❖ En inglés este filtro se conoce como *Optimum Notch Filter*

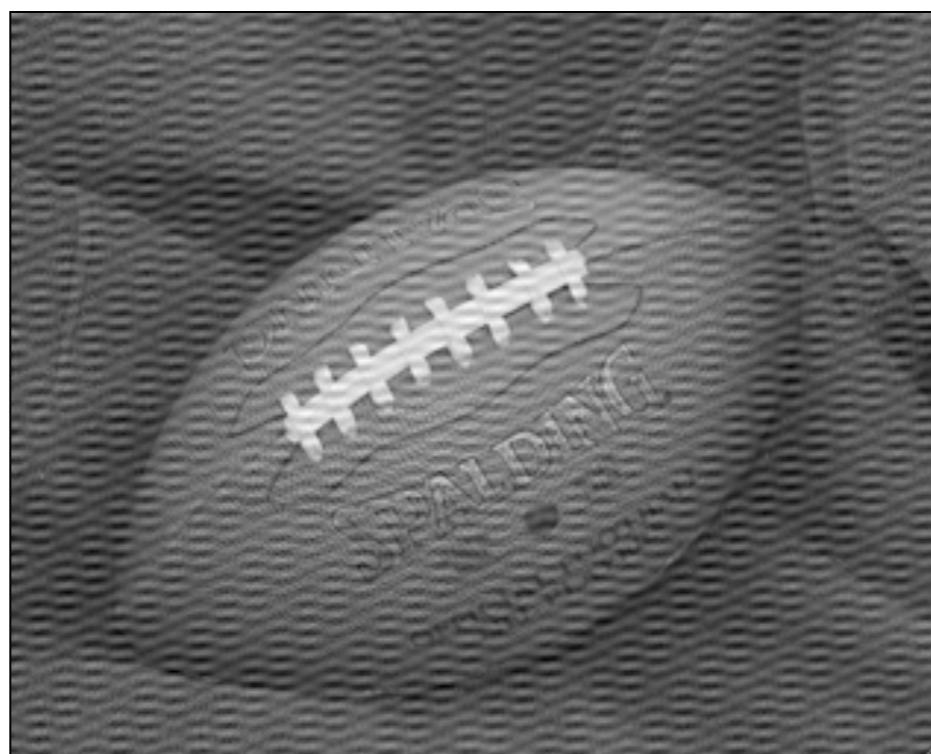
## ▶ Filtros en la frecuencia (Puntuales óptimos)

Parabandas

Pasabanda

Puntuales

Puntuales -  
óptimo



$$g(x, y)$$

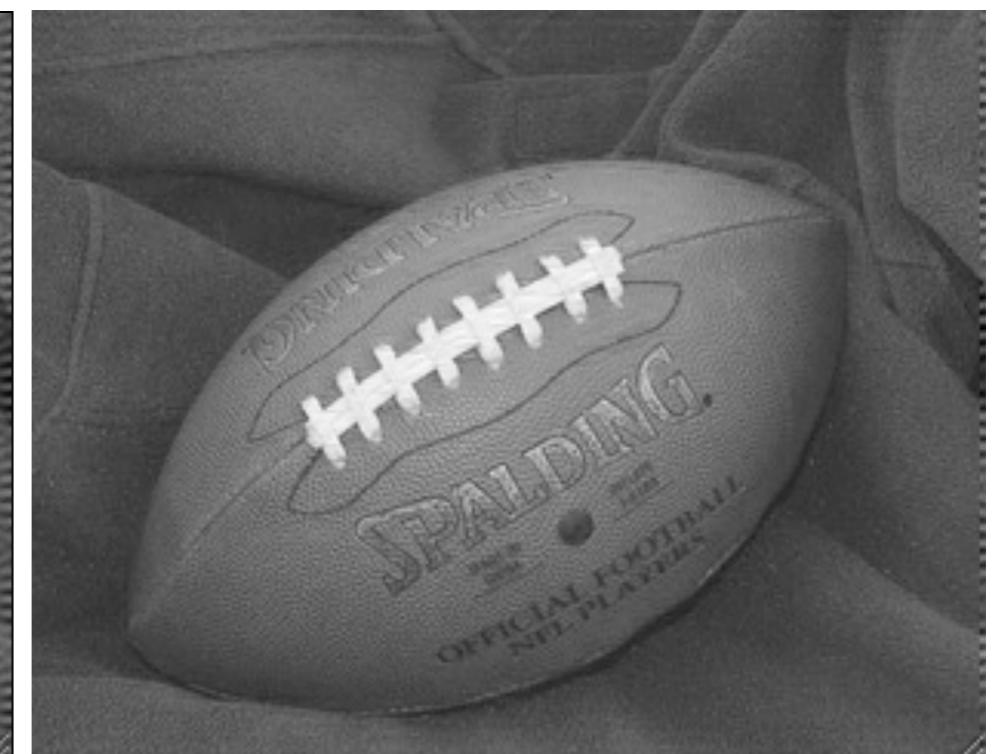
---

Imagen  
original

$$\hat{f}(x, y)$$

---

Imagen  
filtrada



## ▶ Filtros en la frecuencia (Puntuales óptimos)

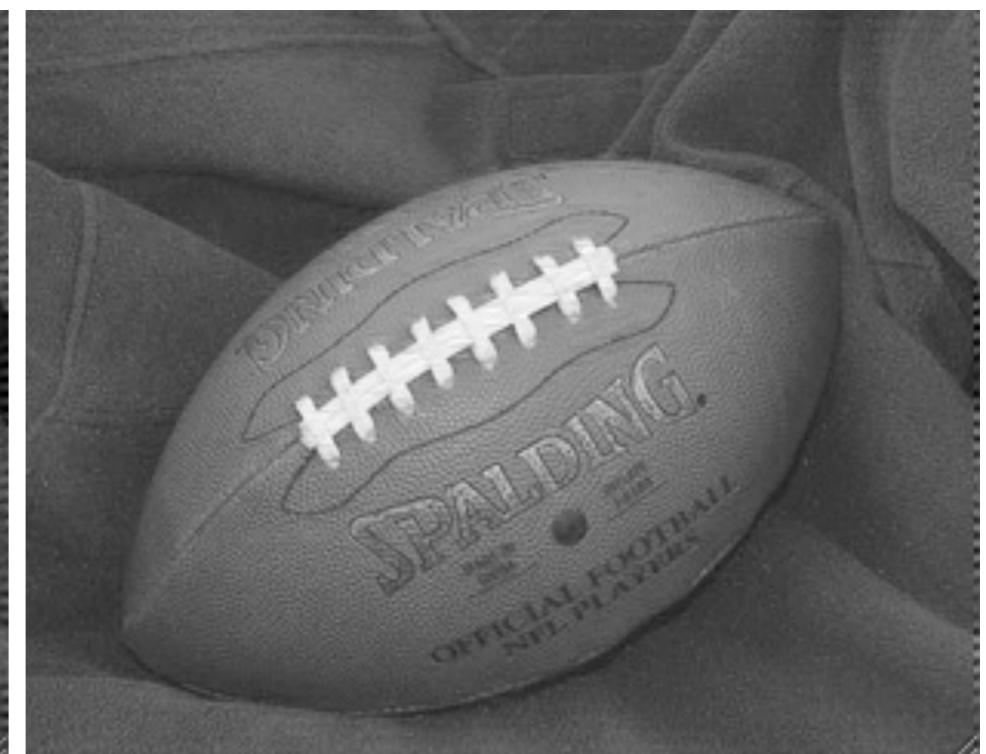
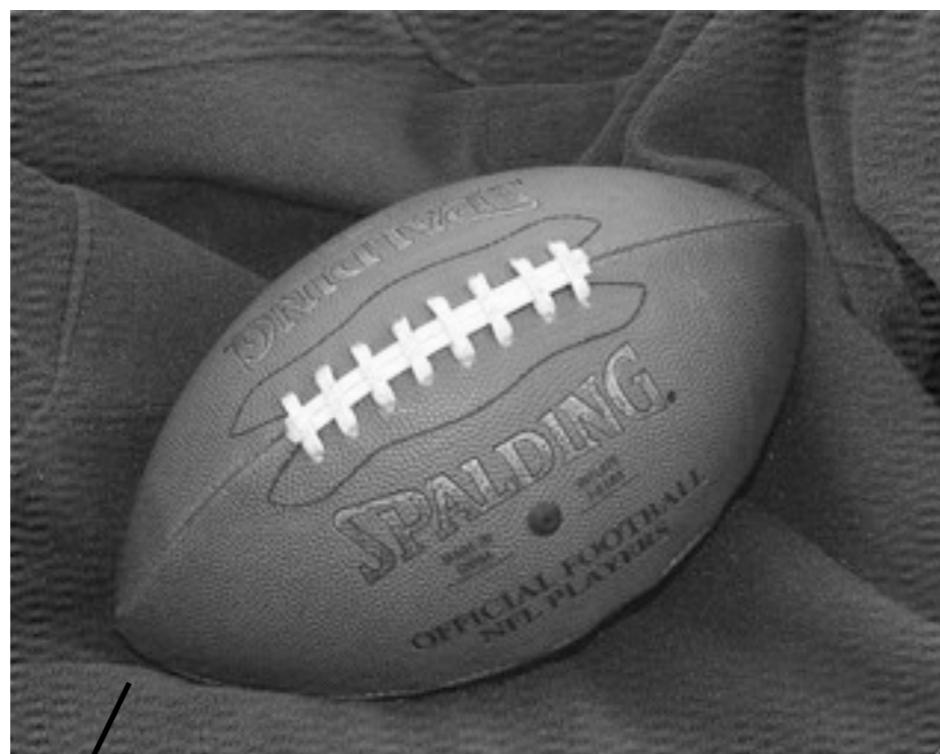
Parabandas

Pasabanda

Puntuales

Puntuales -  
óptimo

- Notar la diferencia entre aplicar un filtro puntual versus un filtro puntual óptimo.



No se eliminó completamente el ruido



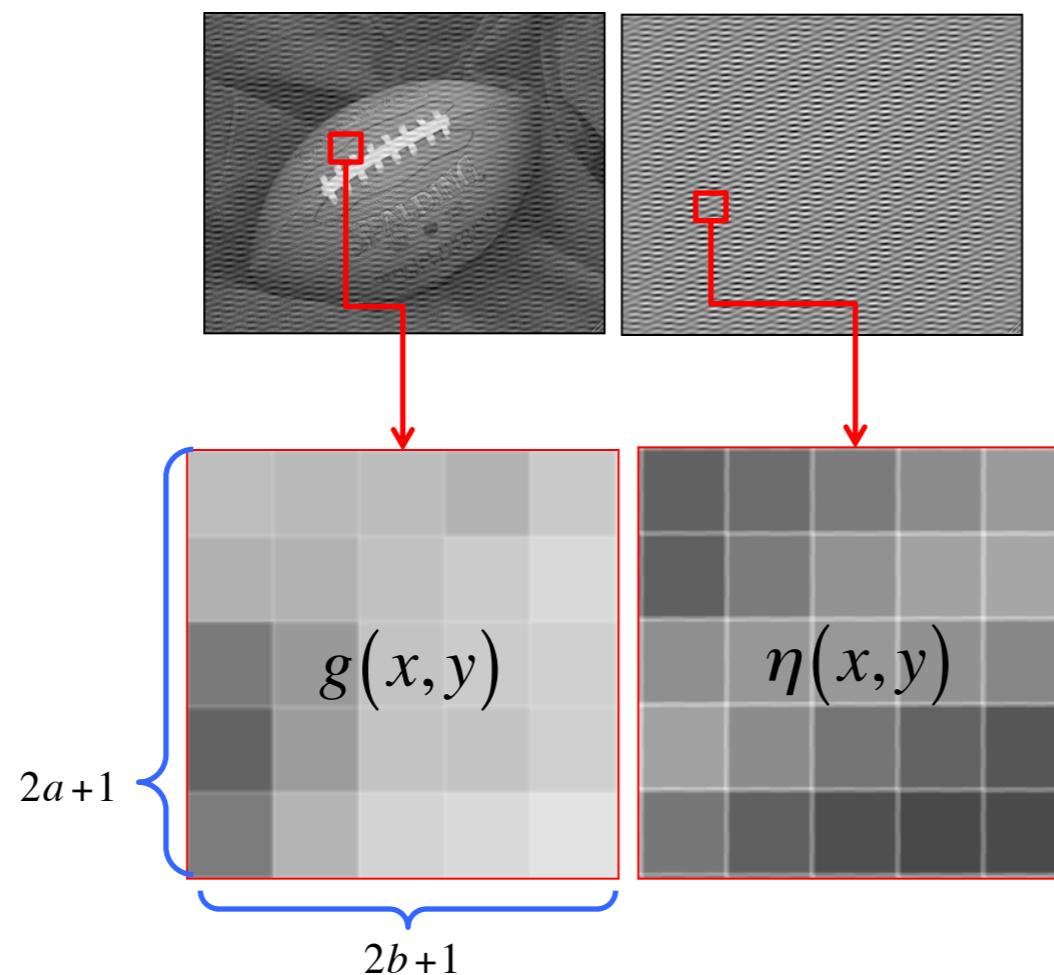
Filtro  
puntual

Filtro puntual  
óptimo

## ▶ Filtros en la frecuencia (Puntuales óptimos)



- **Importante.** El tercer paso es clave en el algoritmo. El proceso consiste en recorrer la imagen original y el patrón a mismo tiempo e ir almacenando temporalmente submatrices.



$$\omega(x,y) = \frac{\overline{g(x,y) \cdot \eta(x,y)} - \bar{g}(x,y) \cdot \bar{\eta}(x,y)}{\overline{\eta^2}(x,y) - \bar{\eta}^2(x,y)}$$

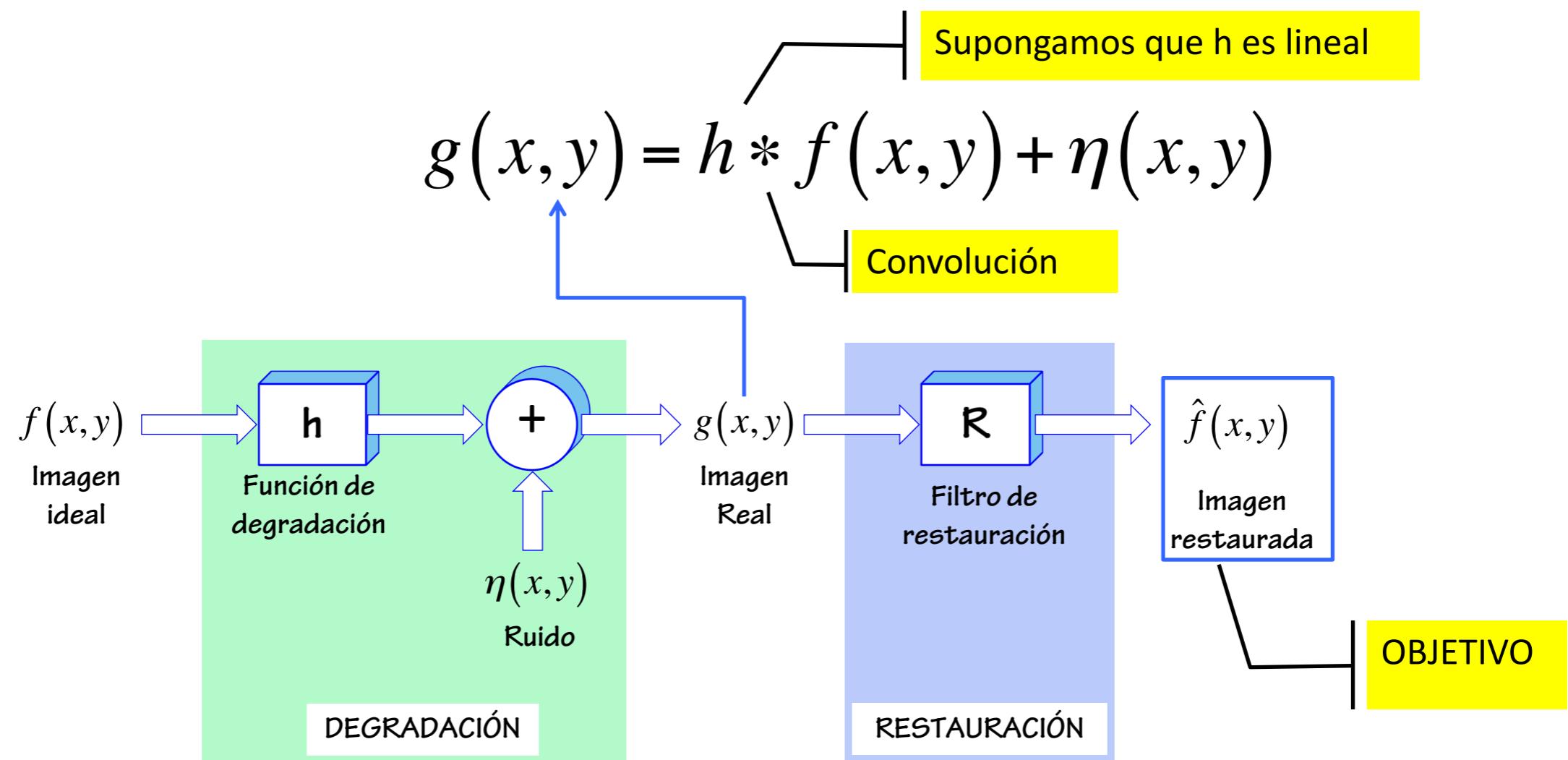
$\overline{g(x,y) \cdot \eta(x,y)}$	= np.mean(g*nu)
$\bar{g}(x,y)$	= np.mean(g)
$\bar{\eta}(x,y)$	= np.mean(nu)
$\overline{\eta^2}(x,y)$	= np.mean(nu**2)
$\bar{\eta}^2(x,y)$	= npmean(nu)**2

- Modelos de degradación
- Estimación de la función de degradación
  - Filtros Lineal
  - Mínimos cuadrados (Wiener)
  - Paramétrico



## ▶ ¿Cómo reducimos el ruido?

- En las clases anteriores reducimos el ruido únicamente en imágenes con ruido aleatorio. Suponga ahora que empleamos una matriz de degradación  $h$  la cual afecta la imagen original.



## ▶ Por modelación (ej. Turbulencia)

- Si tenemos algún conocimiento sobre la forma como se capturó la imagen, es posible estimar sus parámetros. Ejemplo: Turbulencia atmosférica



$f(x, y)$

$$g(x, y) = h * f(x, y)$$

↓

Turbulencia atmosférica

$$H(u, v) = e^{-k(u^2 + v^2)^{\frac{5}{6}}}$$

↓      ↓

Recuerde que  $u$  y  
 $v$  son posiciones  
de una matriz



$g(x, y)$

$k=0.005$

## ▶ Por modelación (turbulencia)

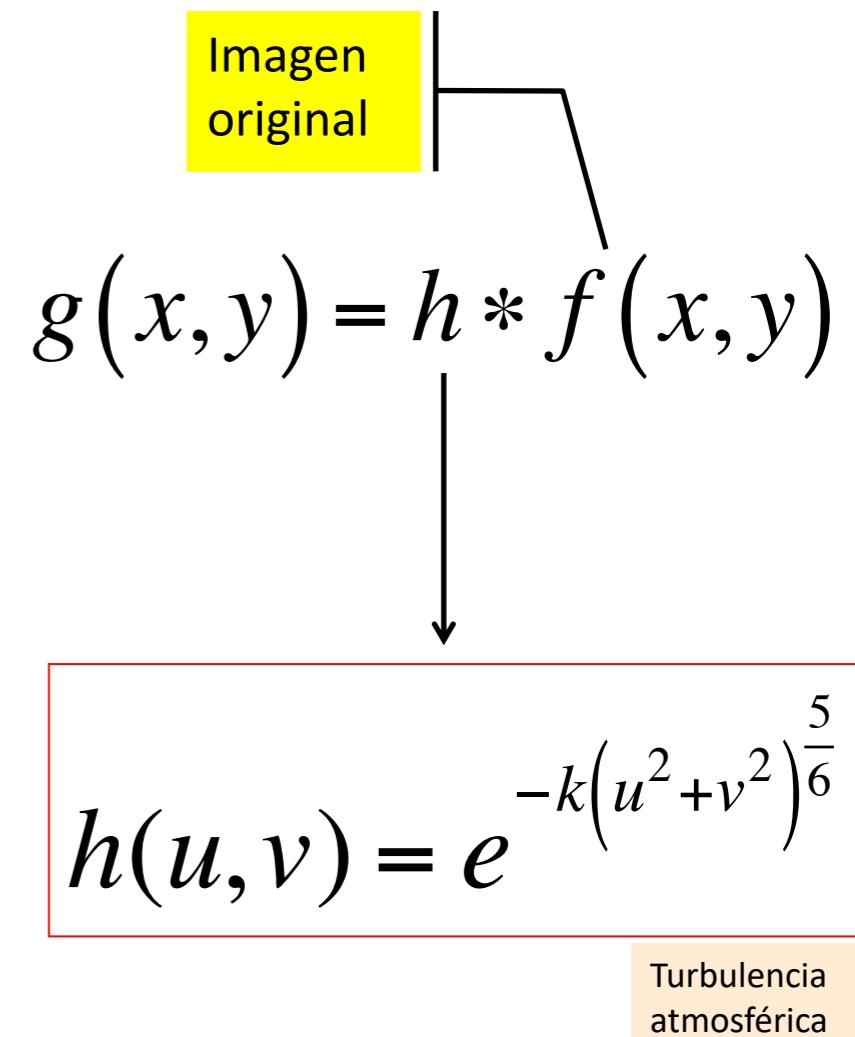
- Si tenemos algún conocimiento sobre la forma como se capturó la imagen, es posible estimar sus parámetros. Ejemplo: Turbulencia atmosférica

```
def ruido_turbulento(f,k):
    F=np.fft.fft2(f)
    m,n =F.shape
    rx = np.linspace(-m/2, m/2, m)
    ry = np.linspace(-n/2, n/2, n)
    U,V = np.meshgrid(rx, ry)

    #ecuacion de turbulencia
    H= np.exp(-k*(U**2+V**2)**(5/6))
    H= np.fft.fftshift(H)

    #aplicacion del filtro
    FT=H*f
    ft=np.real(np.fft.ifft2(FT))
    return ft
```

Recuerde que  $u$  y  $v$  son posiciones de una matriz



## ▶ Por modelación (movimiento lineal)

- Si tenemos algún conocimiento sobre la forma como se capturó la imagen, es posible estimar sus parámetros.


 $f(x, y)$ 

Imagen degradada

$$g(x, y) = h * f(x, y)$$

Imagen original


 $g(x, y)$ 

$$H(u, v) = T \cdot \text{sinc}[\pi \cdot (ua + vb)] \cdot e^{-i\pi(ua + vb)}$$

Movimiento lineal

$a, b$  y  $T$  son escalares

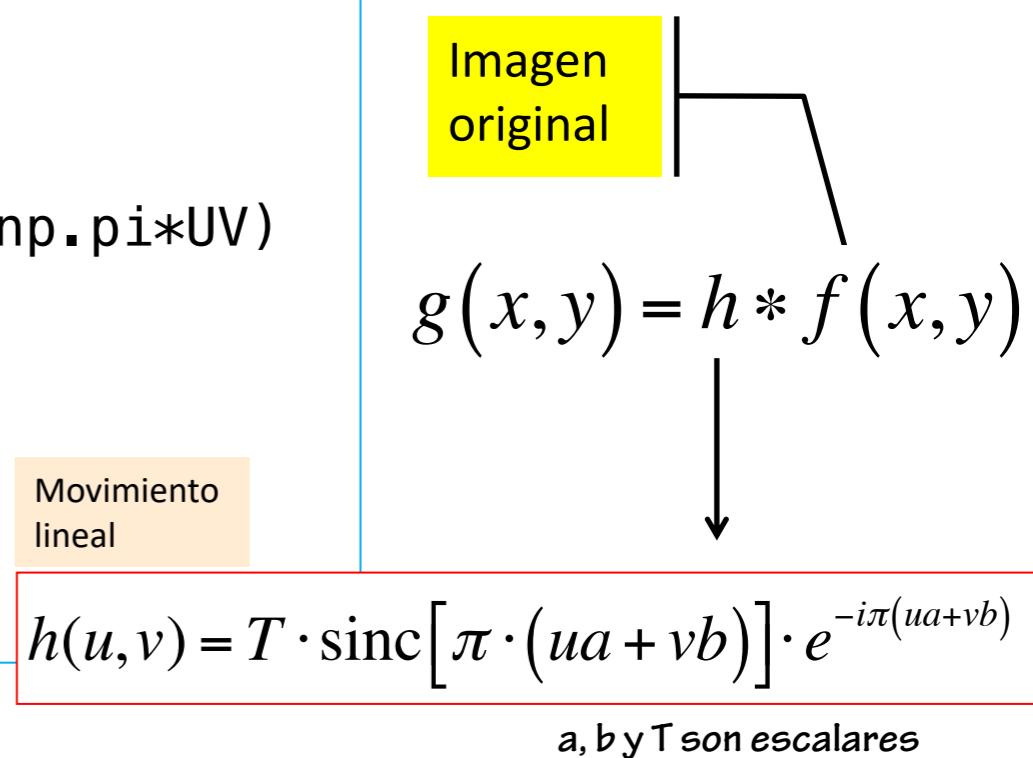
## ▶ Por modelación (movimiento lineal)

- Si tenemos algún conocimiento sobre la forma como se capturó la imagen, es posible estimar sus parámetros.

```
def ruido_movimiento(f,a,b,T):
    F = np.fft.fft2(f) # Espectro
    m,n = F.shape
    rx = np.linspace(-1, 1, m)
    ry = np.linspace(-1, 1, n)
    U,V = np.meshgrid(rx, ry)

    #ecuacion de movimiento
    UV=U*a+V*b
    H= T * np.sinc(np.pi*UV)*np.exp(-1j*np.pi*UV)
    H= np.fft.fftshift(H)

    #aplicacion del filtro
    FT = H*F
    ft = np.real(np.fft.ifft2(FT))
    return ft
```



- Modelos de degradación
- Estimación de la función de degradación
  - Filtros Lineal
  - Mínimos cuadrados (Wiener)
  - Paramétrico



## ▶ ¿Cómo reducimos el ruido?

- En las clases anteriores reducimos el ruido únicamente en imágenes con ruido aleatorio. Suponga ahora que empleamos una matriz de degradación  $h$  la cual afecta la imagen original.
- ¿Cómo podemos *restaurar la imagen*?

$$g(x, y) = h * f(x, y) + \eta(x, y)$$
$$G(u, v) = H(u, v) \cdot F(u, v) + N(u, v)$$

Note que es multiplicación

Pasemos este producto al Espectro con la ayuda de Fourier



## ▶ ¿Cómo reducimos el ruido?

- En las clases anteriores reducimos el ruido únicamente en imágenes con ruido aleatorio. Suponga ahora que empleamos una matriz de degradación  $h$  la cual afecta la imagen original.
- ¿Cómo podemos *restaurar la imagen*?

$$\begin{aligned} g(x,y) &= h * f(x,y) + \eta(x,y) \\ \\ G(u,v) &= H(u,v) \cdot F(u,v) + N(u,v) \\ \\ G(u,v) &= H(u,v) \cdot F(u,v) \end{aligned}$$

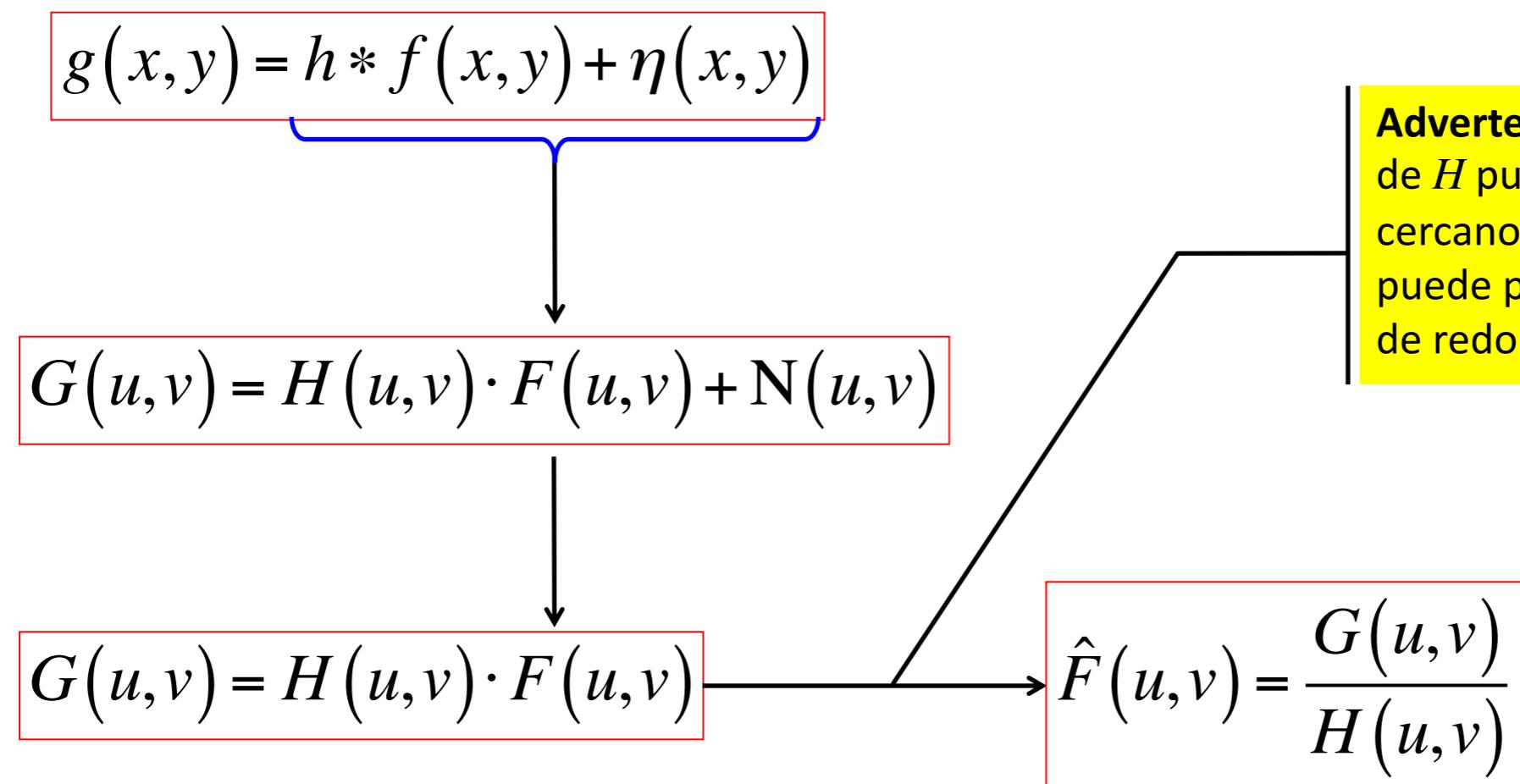
Problema: No es posible obtener el espectro del ruido ya que la función del ruido es aleatorio

El filtro lineal no puede manejar el ruido ☹

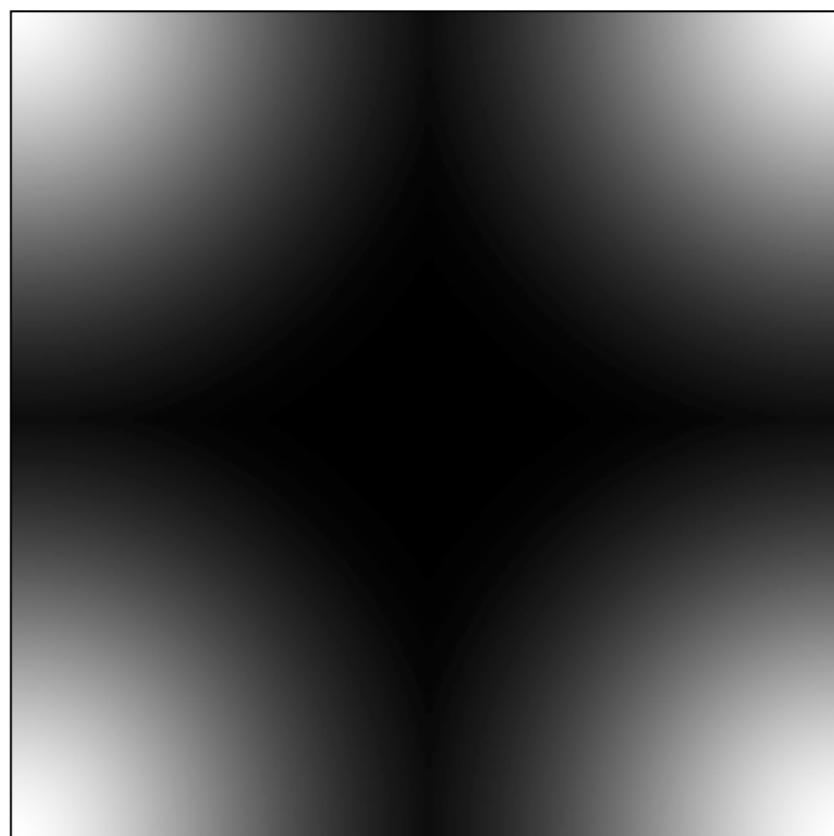


## ▶ ¿Cómo reducimos el ruido?

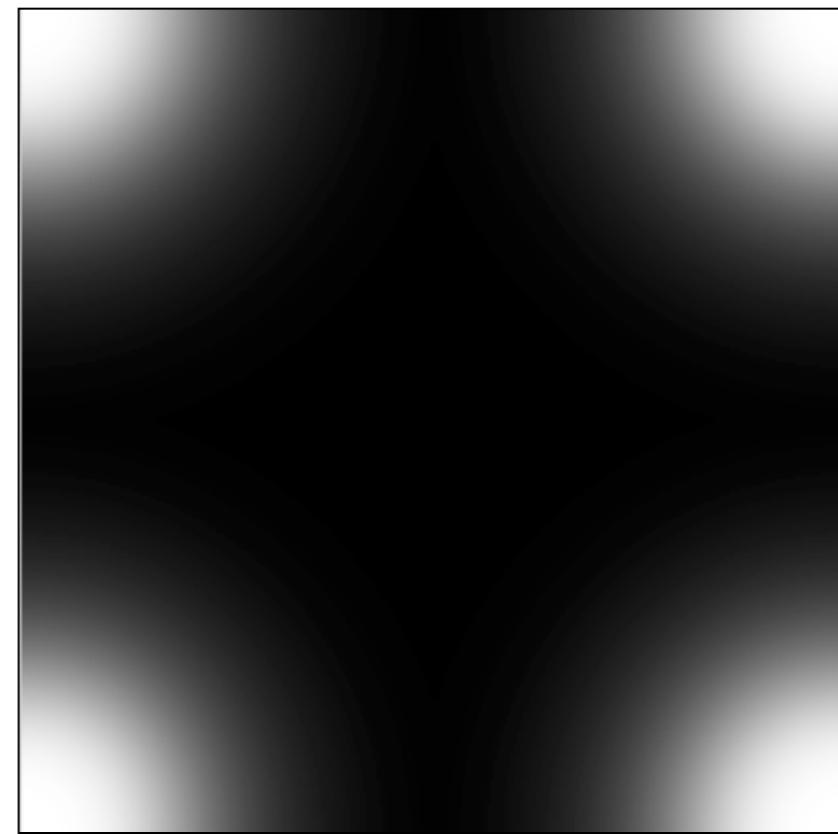
- En las clases anteriores reducimos el ruido únicamente en imágenes con ruido aleatorio. Suponga ahora que empleamos una matriz de degradación  $h$  la cual afecta la imagen original.
- ¿Cómo podemos *restaurar la imagen*?



- ▶ ¿Cómo determinamos  $H$ ?
  - Por modelación : Como vimos al comienzo de la clase.
  - Por experimentación : Si tenemos un filtro gaussiano podemos emplear uno similar u otro del tipo butterworth.



Matriz  $H$  (desconocida)  
Aplicada en la slide 23



Matriz  $H$  (Estimada)  
Aplicada en slide 32 (radio=23)

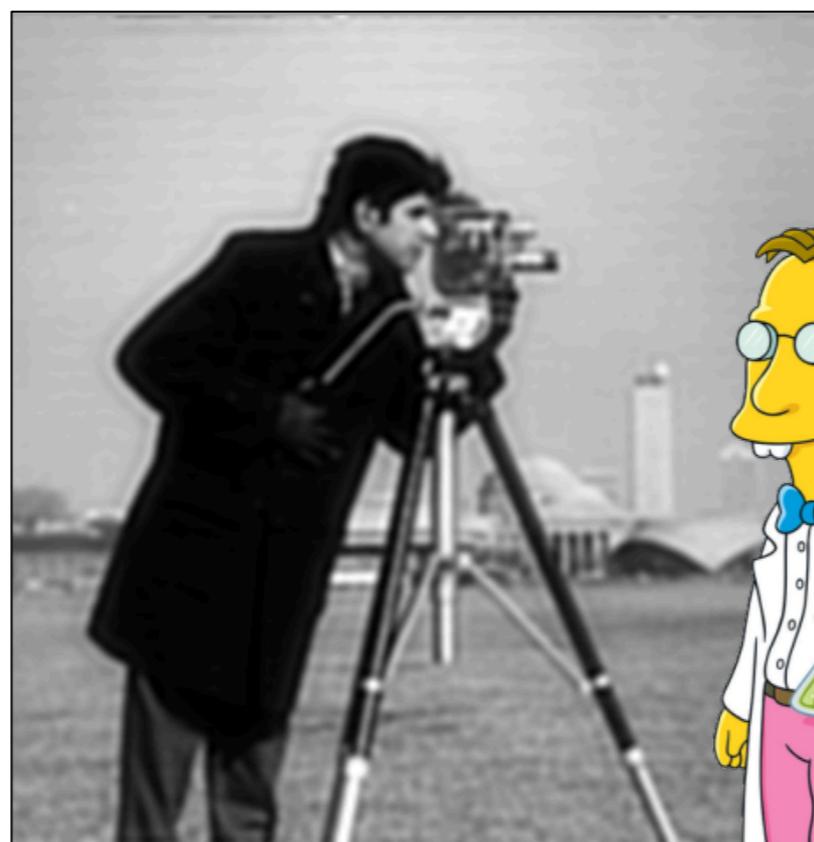
- ▶ ¿Cómo reducimos el ruido?
  - ¿Cómo podemos estimar *la imagen restaurada*?

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)}$$

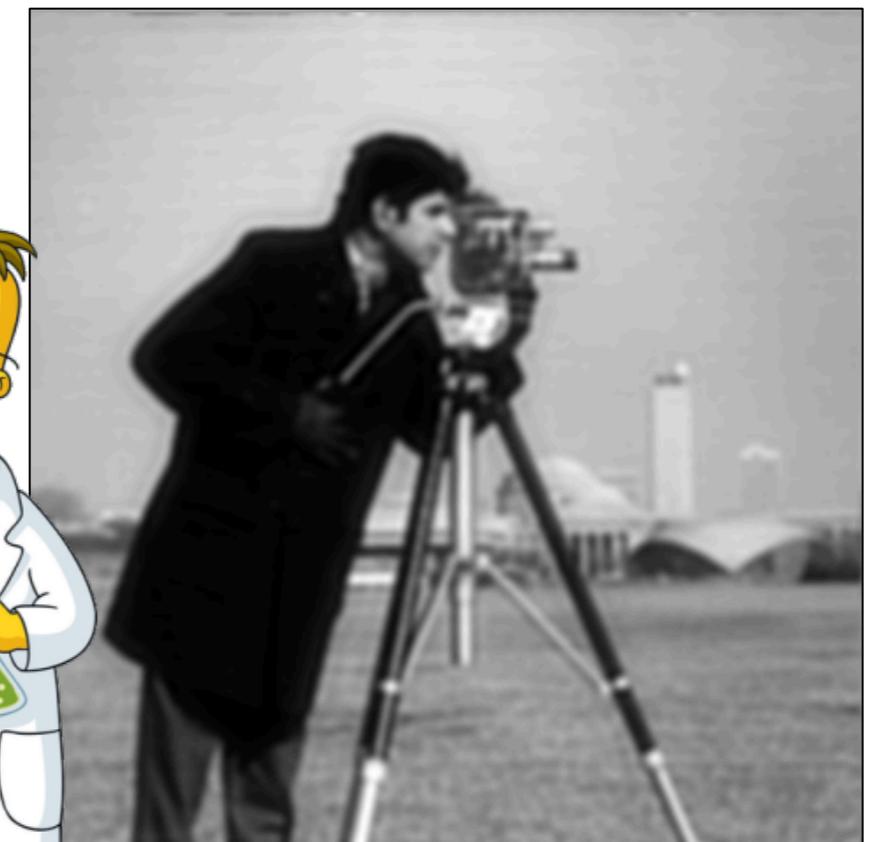
Solución: Para evitar el problema de redondeo, debemos emplear una parte del filtro H. Ejemplo, un filtro pasa alto con un radio variable



Ruido Turbulento k=0.005



Filtro H con radio 23



Filtro H con radio 26

- ¿Cómo determinamos  $H$ ?

```
def filtro_inv(g, radio):
    G=np.fft.fft2(g)
    M,N = G.shape

    H= btw(M,N,2,radio)

    Ft= G/H

    S= np.real(np.fft.ifft2(Ft))
    return S
```

$$\hat{F}(u,v) = \frac{G(u,v)}{H(u,v)}$$

Limitamos los valores menores de H

Filtro de Butterworth

```
def btw(M,N,n,fc):
    # M: filas de la imagen
    # N: columnas de la imagen
    # n: orden del filtro
    # fc: frecuencia de corte
    # Calculo de la malla

    vx = np.linspace(-M/2, M/2, M)
    vy = np.linspace(-N/2, N/2, N)
    U,V = np.meshgrid(vy,vx)
    f = np.sqrt(U**2+V**2)

    #filtro de butterworth centrado
    F = 1/(1+ (f/fc)**(2*n))
    F= np.fft.fftshift(F)

    return(F)
```

- Estimación de la función de degradación
  - Filtros Lineal
  - Mínimos cuadrados (Wiener)
  - Paramétrico
  - Comparación



## ▶ Filtro de Wiener:

- Como vimos anteriormente, el filtro lineal no es capaz de reducir el ruido. A continuación revisaremos el filtro de Wiener que es capaz de determinar la función de degradación y la estimar las características del ruido en un proceso combinado.
- Supuestos
  - Se asume que el ruido y la imagen no están correlacionados.
  - Que ya sea la imagen ideal o el ruido tiene media igual a cero.
  - Los niveles de gris de la imagen estimada son una función lineal de la imagen degradada.

## ► Filtro de Wiener:

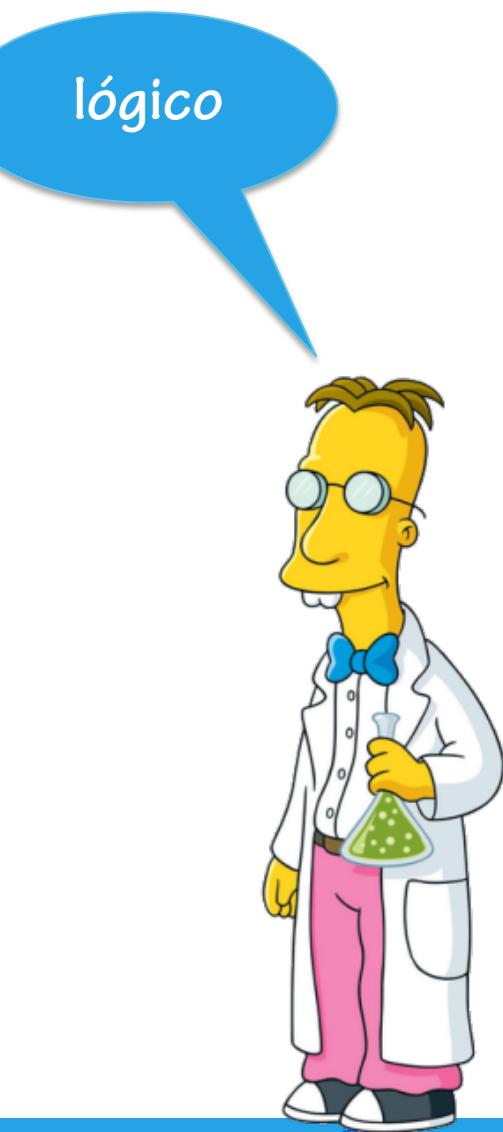
- Basado en los supuestos previos, la formulación de Wiener en el **dominio de la frecuencia** viene dada por:

Conjugado complejo

$$\hat{F}(u, v) = \left( \frac{H^*(u, v)}{|H(u, v)|^2 + \lambda \cdot \frac{S_\eta(u, v)}{S_f(u, v)}} \right) \cdot G(u, v)$$

Imagen restaurada

Imagen degradada



▶ Filtro de Wiener:

- En la ecuación anterior empleamos el conjugado complejo puede ser expresado como el producto de la cantidad compleja al cuadrado. Por ello llegamos a la siguiente expresión :

$$\hat{F}(u, v) = \left( \frac{|H(u, v)|^2}{H(u, v)} \cdot \frac{1}{\left( |H(u, v)|^2 + \lambda \cdot \frac{S_\eta(u, v)}{S_f(u, v)} \right)} \right) \cdot G(u, v)$$

$H(u, v)$  : Función de degradación

$$|H(u, v)|^2 = H^*(u, v) \cdot H(u, v)$$

$S_\eta(u, v)$  : Espectro de potencia del ruido

$S_f(u, v)$  : Espectro de potencia de la imagen sin degradación

$\lambda$  : Factor de ajuste, normalmente fijo en 1

## ▶ Filtro de Wiener:

- Debido a que no conocemos el espectro de potencia del ruido ni de la imagen no degradada, usualmente se emplea **un factor de ajuste  $K$**  que se asigna experimentalmente.

$$\hat{F}(u, v) = \left( \frac{|H(u, v)|^2}{H(u, v) \cdot (|H(u, v)|^2 + K)} \right) \cdot G(u, v)$$

$$\begin{aligned} S_\eta(u, v) &\longrightarrow |N(u, v)|^2 \\ S_f(u, v) &\longrightarrow |F(u, v)|^2 \end{aligned} \quad \left. \begin{array}{l} \text{Normalmente} \\ \text{estos espectros} \\ \text{no son conocidos} \end{array} \right\} \longrightarrow \lambda \cdot \frac{S_\eta(u, v)}{S_f(u, v)} \approx K$$

## ▶ Filtro de Wiener:

- Basado en los supuestos previos, finalmente la formulación de Wiener corresponde a la siguiente ecuación :

$$\hat{F}(u, v) = \left( \frac{|H(u, v)|^2}{H(u, v) \cdot (|H(u, v)|^2 + K)} \right) \cdot G(u, v)$$

```
G = np.fft.fft2(noise_image)
W = np.conj(H)/(np.abs(H)**2 + K)
F = W*G
iRestored = np.real(np.fft.ifft2(F))
```

$$|H(u, v)|^2 = \text{abs}(H)^{\star\star 2}$$

- ▶ Filtro de Wiener:
  - Ejemplos **sin ruido aleatorio** y **movimiento lineal**



Imagen degrada con  
movimiento lineal  
 $a=1, b=1, T=1$



Imagen restaurada (sin ruido)  
 $k=0.0006393$

- ▶ Filtro de Wiener:
  - Ejemplos **con ruido aleatorio gaussiano y movimiento lineal**



Imagen degrada con movimiento  
lineal y ruido gaussiano  
 $a=1, b=1, T=1, s=0.001$



Imagen restaurada

- ▶ Filtro de Wiener:
  - Ejemplos **sin ruido aleatorio y turbulencia**



Imagen degrada con  
movimiento turbulento  
 $k=0.002$



Imagen restaurada (sin ruido)  
 $k=0.001178$

- ▶ Filtro de Wiener:
  - Ejemplos **sin ruido aleatorio y turbulencia**



Imagen degrada con  
movimiento turbulento  
 $k=0.001$



Imagen restaurada (sin ruido)  
 $k=0.001358$

- ▶ Filtro de Wiener:
  - Ejemplos **con ruido aleatorio y turbulencia**



Imagen degrada con  
movimiento turbulento  
 $k=0.001$



Imagen restaurada

- Estimación de la función de degradación
  - Filtros Lineal
  - Mínimos cuadrados (Wiener)
  - Paramétrico
  - Comparación

▶ Filtro mínimos cuadrados con restricción

- Este filtro está basado en una formulación similar a la de Wiener, sin embargo, en vez de emplear un factor de ajuste K, utiliza una matriz de filtraje. La formulación es la siguiente

$$\hat{F}(u, v) = \left( \frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right) \cdot G(u, v)$$

donde  $P(u, v)$  : es la transformada de fourier de:  $p(u, v) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$   
 $H(u, v)$  : es la función de degradación  
 $H^*(u, v)$  : es el conjugado de  $H$   
 $\gamma$  : parámetro de ajuste

## ▶ Filtro mínimos cuadrados con restricción

- Este filtro está basado en una formulación similar a la de Wiener, sin embargo, en vez de emplear un factor de ajuste K, utiliza una matriz de filtraje. La formulación es la siguiente:

$$\hat{F}(u, v) = \left( \frac{H^*(u, v)}{|H(u, v)|^2 + \gamma |P(u, v)|^2} \right) \cdot G(u, v)$$

```
m= 256
n= 256
G = np.fft.fft2(noise_image)

p=np.array([[0 ,-1 ,0 ],[ -1,  4, -1],[0 , -1,  0 ]])
Pp=np.fft.fft2(p,s=[m,n]) _____
```

gamma=0.01
vector= np.linspace(1e-8,1e-4, 100)
F= (np.conj(H)\*G)/( abs(H)\*\*2 + gamma\*abs(Pp)\*\*2)
iRestored = np.real(np.fft.ifft2(F))

Podemos fijar  
el tamaño

- ▶ Filtro mínimos cuadrados con restricción
  - Ejemplos **sin ruido aleatorio** y **movimiento lineal**



Imagen degrada con  
movimiento lineal  
 $a=1, b=1, T=1$



Imagen restaurada  
 $\text{gamma}=0.0001$

- ▶ Filtro mínimos cuadrados con restricción
  - Ejemplos **con ruido aleatorio y movimiento lineal**



Imagen degrada con movimiento  
lineal y ruido gaussiano  
 $a=1, b=1, T=1, s=0.001$



Imagen restaurada  
 $\text{gamma}=0.01$

- ▶ Filtro mínimos cuadrados con restricción
  - Ejemplos **sin ruido aleatorio** y **turbulencia**



Imagen degrada con  
movimiento turbulento  
( $k=0.002$ )



Imagen restaurada  
 $\text{gamma}=0.00001$

- ▶ Filtro mínimos cuadrados con restricción
  - Ejemplos **con ruido aleatorio y turbulencia**



Imagen degrada con  
movimiento turbulento y ruido  
gaussiano ( $k=0.001$ )



Imagen restaurada  
 $\text{gamma}=0.01$

- Estimación de la función de degradación
  - Filtros Lineal
  - Mínimos cuadrados (Wiener)
  - Paramétrico
  - Comparación



## ▶ Comparación de filtros

- Los filtros que estiman la función de degradación son efectivas herramientas para reducir la degradación en las imágenes.
- El mayor problema es que siempre debemos poseer una buena aproximación de las matriz de degradación



Imagen degradada



Filtro Inverso



Filtro de Wiener



Filtro Paramétrico