Visión por Computadora I

Ing. Maxim Dorogov

(mdorogov@fi.uba.ar)

Laboratorio de Sistemas Embebidos -FIUBA

FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

FILTROS LINEALES

(a)

(b)

(c)

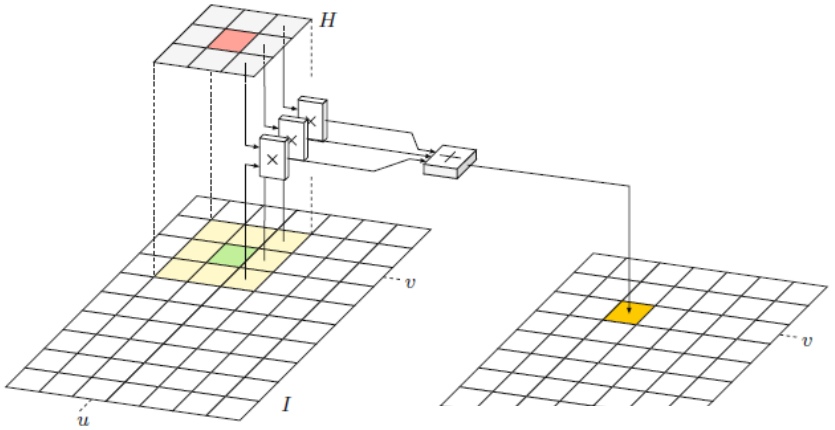
(d)

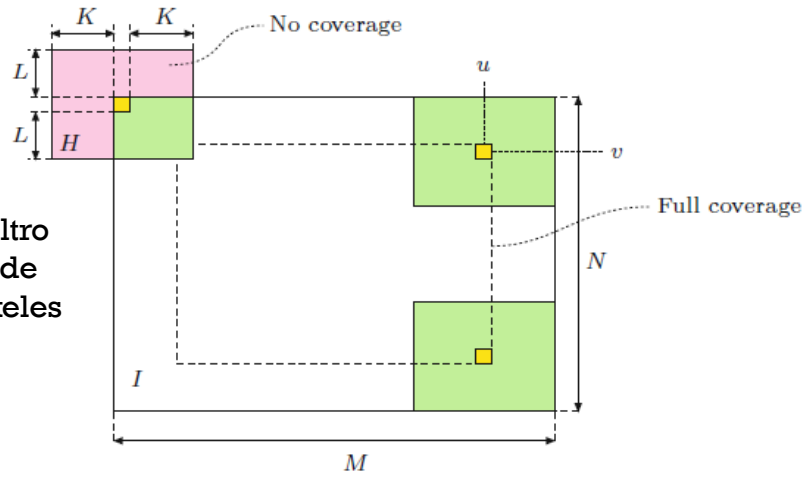
- A diferencia de los operadores de pixeles los filtros utilizan varios pixeles de la imagen original para computar 1 pixel en la imagen de salida (operadores de vecindad)
- Los ajustes adaptativos de histograma o códigos LBP son ejemplos de operadores de vecindad (se utiliza el valor de los píxeles vecinos para determinar el valor de salida del píxel en cuestión)
- Se pueden aplicar filtros a imágenes de manera de:
  1. Agregar suavizados
  2. Remarcar detalles
  3. Acentuar bordes
  4. Remover ruido
  5. Resaltar características para su posterior procesamiento.
- A continuación vamos a ver el caso de Operadores Lineales.

$$I'(u,v) \leftarrow \sum_{(i,j)\in R_H} I(u+i, v+j) \cdot H(i,j), \quad \rightarrow I' = f \otimes h$$

H(k,l): $Kernel, máscara, coef.filtro, núcleo$
$\otimes$: $Operador\ correlación.$

FILTROS LINEALES





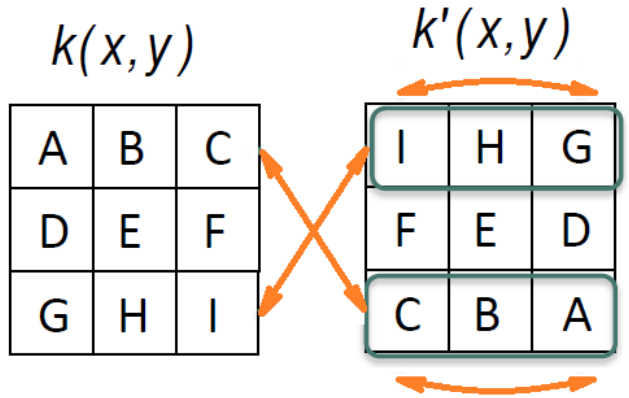
Solo es posible aplicar el filtro cuando este se superpone de forma completa con los pixeles de la imagen de entrada.

Conceptos generales:

- Consideramos al punto central del filtro como origen (0, 0). (Los filtros generalmente son impares)
- Desplazamos el filtro a lo largo y ancho de la imagen y multiplicamos los coeficientes punto a punto, sumamos y obtenemos el valor del pixel en la posición (u,v) de la imagen de salida.
- La imagen de salida resulta mas chica que la imagen original.
- Definimos:
  - **Stride:** Numero de pixeles que desplazo el filtro en X como en Y.
  - **Padding:** Agregado de pixeles en los bordes de la imagen original para preservar las dimensiones en la imagen de salida.
  - **Ganancia:** Es la suma de todos los coeficientes. Vamos a buscar siempre filtros de ganancia unitaria para evitar la saturación de luminancia.

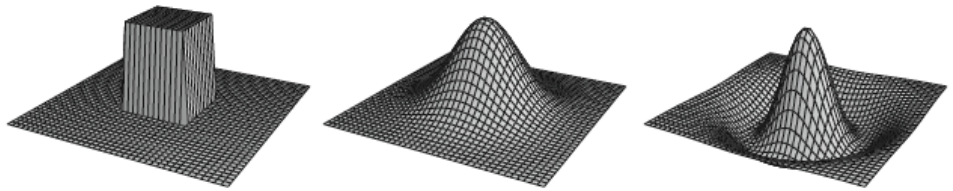
$$H = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

FILTROS LINEALES



Correlación con k(x,y)

Convolución con k'(x,y)



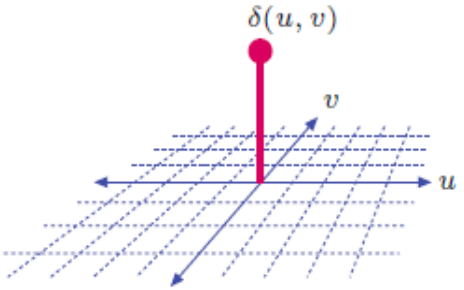
$$\frac{1}{9}\cdot\begin{bmatrix}0&0&0&0&0\\0&1&1&1&0\\0&1&1&1&0\\0&1&1&1&0\\0&0&0&0&0\end{bmatrix}\quad \frac{1}{57}\cdot\begin{bmatrix}0&1&2&1&0\\1&3&5&3&1\\2&5&9&5&2\\1&3&5&3&1\\0&1&2&1&0\end{bmatrix}\quad \frac{1}{16}\cdot\begin{bmatrix}0&0&-1&0&0\\0&-1&-2&-1&0\\-1&-2&16&-2&-1\\0&-1&-2&-1&0\\0&0&-1&0&0\end{bmatrix}$$

- Una variante de la fórmula es invertir los offsets sobre I(i,j)

$$\begin{aligned} I'(u,v) &= \sum_{(i,j)\in R_H} I(u-i,v-j)\cdot H(i,j) \\ &= \sum_{(i,j)\in R_H} I(u+i,v+j)\cdot H(-i,-j) \\ &= \sum_{(i,j)\in R_H} I(u+i,v+j)\cdot H^*(i,j). \end{aligned}$$

Esto se escribe como

$$I' = I * H$$



Donde $*$ es el operador *convolución*

- ¿Por qué rotamos el núcleo?.

Porque si lo convolucionamos con la señal impulso:

$\delta(i=0,j=0)=1$ (y cero para cualquier otro caso) resulta $h*\delta=h$

Ejemplo:

$$h(n)=[-1,0,1];\ \delta(n)=1$$

$$h'(n)=[1,0,-1]\rightarrow g(n)=h'\otimes\delta=[-1,0,1]=h=h*\delta$$

Esto parece trivial si ya conocemos los coeficientes del filtro pero tiene su aplicación en identificación de sistemas lineales.

- ¿Qué pasa si el núcleo es simétrico?

FILTROS LINEALES

Propiedades de la convolución:

1. Conmutativa:

$$h * f = f * h$$

2. Lineal:

$$s \cdot (f * h) = f * (s \cdot h) = (s \cdot f) * h$$

3. Asociativa

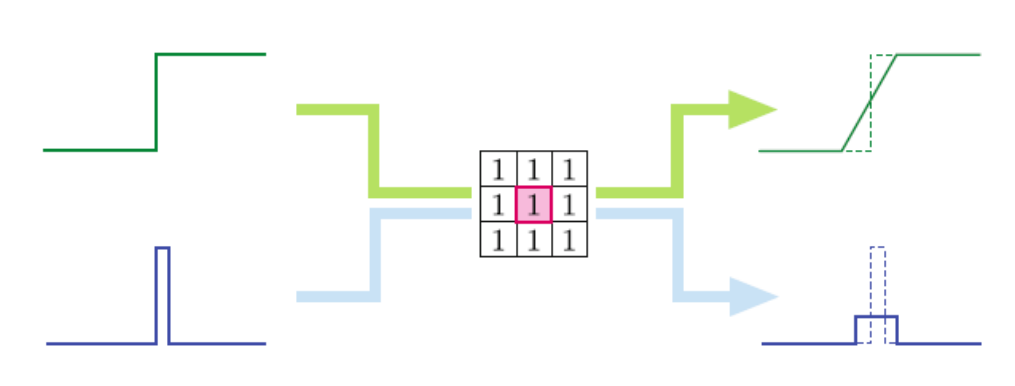
$$f * (h_1 * h_2) = (f * h_1) * h_2$$

4. Invariante en el tiempo (o a desplazamientos para el caso de imágenes):

$$f[u - d, v - d] * h = (f * h)[u - d, v - d]$$

- Una diferencia importante entre correlación y convolución es que la primera no es asociativa para operaciones con imágenes.

EJEMPLOS DE NÚCLEOS TÍPICOS



- Algunos ejemplos de núcleos (Kernels) típicos:
- $\boldsymbol{Enfoque}: \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$ $\boldsymbol{Desenfoque}: \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$ $\boldsymbol{Repujado}: \begin{bmatrix} -2 & -1 & 0 \\ -1 & 1 & 1 \\ 0 & 1 & 2 \end{bmatrix}$
- $\boldsymbol{LoG}: \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ $\boldsymbol{Sharpen}: \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ -1 & -2 & 1 \end{bmatrix}$ $\boldsymbol{Gauss}: \frac{1}{121}\begin{bmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 7 & 11 & 7 & 2 \\ 3 & 11 & 17 & 11 & 3 \\ 2 & 7 & 11 & 7 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{bmatrix}$
- $\boldsymbol{Sobel}: \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ $\boldsymbol{Prewitt\ y}: \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$ $\boldsymbol{Prewitt\ x}: \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$
- La suma de los filtros debe dar ‘1’ para que la respuesta del filtro no tenga efecto en regiones homogéneas
- Sobel y Prewitt se utilizan para la detección de bordes.
- Sobel y Prewitt son a su vez ejemplos de filtros pasa alto, direccionables.

LOG (LAPLACIANO DEL GAUSSIANO)

- Podemos definir matemáticamente el Laplaciano como

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

$$\frac{\partial^2}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \rightarrow kernel\ x\text{: } [1 \quad -2 \quad 1]$$

$$\frac{\partial^2}{\partial y^2} = f(y+1) + f(y-1) - 2f(y) \rightarrow kernel\ y\text{: } \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & -2 & 1 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

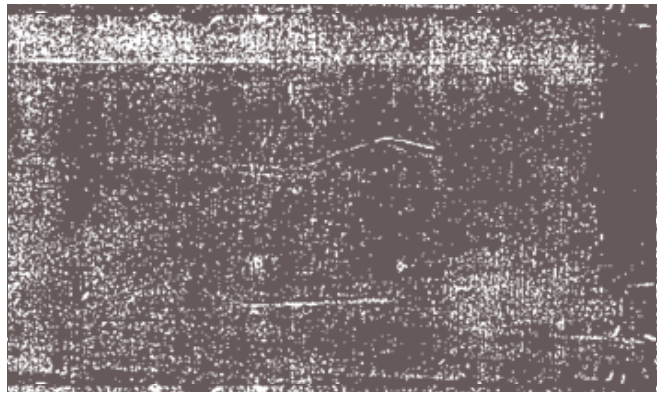
- Detecta los cruces por cero (cambio de signo)
- No usa dos núcleos con orientaciones distintas (como Prewitt, Sobel) y pierde información de orientación
- Es isotrópico→ magnitud de borde uniforme en toda dirección
- Es muy sensible al ruido

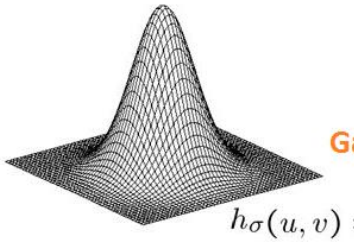
Para reducir el ruido se le puede aplicar *previamente* un filtro gaussiano a la imagen, o…hacerlo en un solo paso!

$$\nabla^2(f * g) = f * \nabla^2(g) \rightarrow Convolución\ con\ el\ Laplaciano\ de\ la\ Gaussiana$$

Matemáticamente, $LoG(x,y) = \frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right] e^{-\frac{x^2+y^2}{2\sigma^2}}$ …y podemos muestrear…

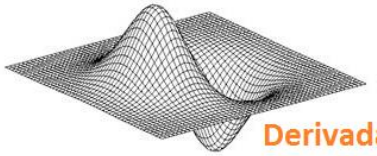
0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0





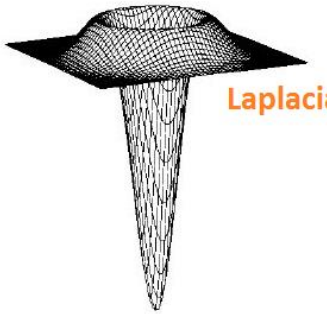
Gaussiana

$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Derivada de Gaussiana



$$\nabla^2 h_\sigma(u,v)$$

Laplaciano de Gaussiana (LoG)

FILTROS SEPARABLES

- La convolución con un núcleo implica procesar por cada píxel K^2 operaciones de multiplicación+suma donde K es el tamaño del núcleo (NxM)
- Esto puede acelerarse realizando primero una convolución unidimensional horizontal seguida de una convolución unidimensional vertical. Requiriendo así $2K$ operaciones.
- Un núcleo que permita hacer esto se dice ser *separable*
- Un núcleo K "separable", es decir corresponda a una convolución sucesiva de un núcleo horizontal h y un núcleo vertical v debe cumplir:

$$\boldsymbol{K} = vh^t$$

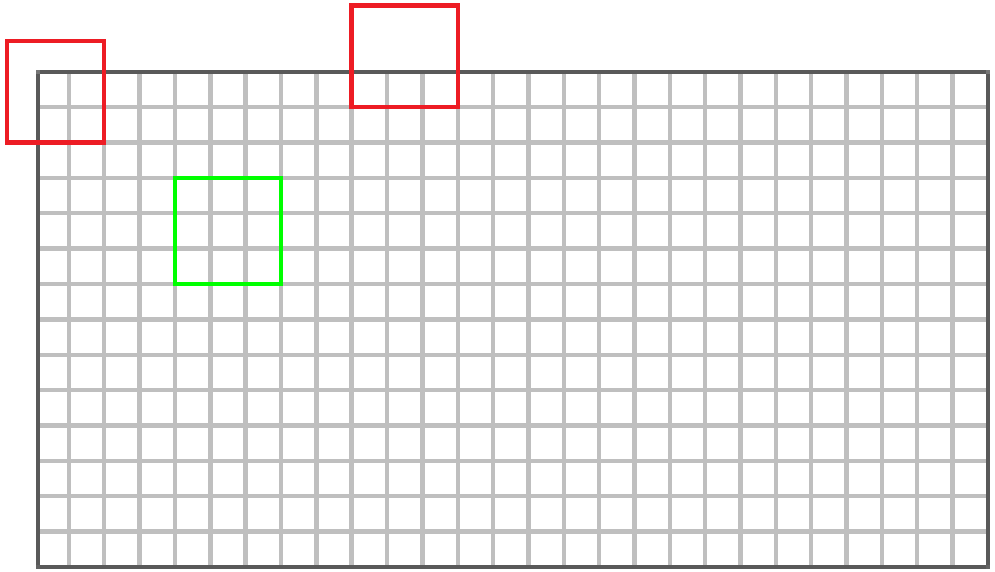
Ejemplos:

- Filtros de suavizado: $\frac{1}{3}\begin{bmatrix}1\\1\\1\end{bmatrix} * \frac{1}{3}\begin{bmatrix}1 & 1 & 1\end{bmatrix} = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\1 & 1 & 1\\1 & 1 & 1\end{bmatrix}$
- Filtro Gaussiano: $\frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix} * \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix} = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\2 & 4 & 2\\1 & 2 & 1\end{bmatrix}$
- Sobel (vale para Prewitt): $\begin{bmatrix}1\\2\\1\end{bmatrix} * \begin{bmatrix}1 & 0 & -1\end{bmatrix} = \begin{bmatrix}1 & 0 & -1\\2 & 0 & -2\\1 & 0 & -1\end{bmatrix}$
- Los filtros Laplacianos también pueden ser implementados como filtros separables.

¿Cómo sabemos si un filtro es separable?

1. Por inspección
2. Tratar al núcleo K como matriz y tomar su descomposición en valores singulares (SVD): $K = \sum_i \sigma_i \mu_i v_i^T$
   - Si solo el primer valor singular σ_0 es distinto de cero, entonces el kernel es separable y $\sqrt{\sigma_0}\,\mu_0$ y $\sqrt{\sigma_0} v_0^T$ son los kernels verticales y horizontales respectivamente.

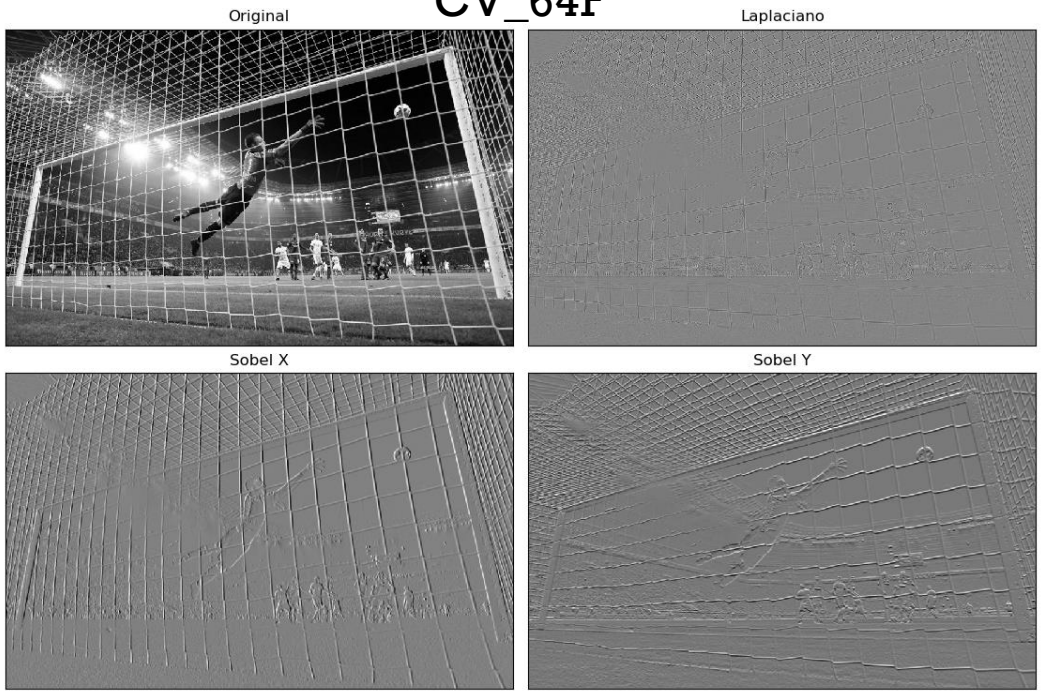
EFECTO DE BORDES (PADDING)



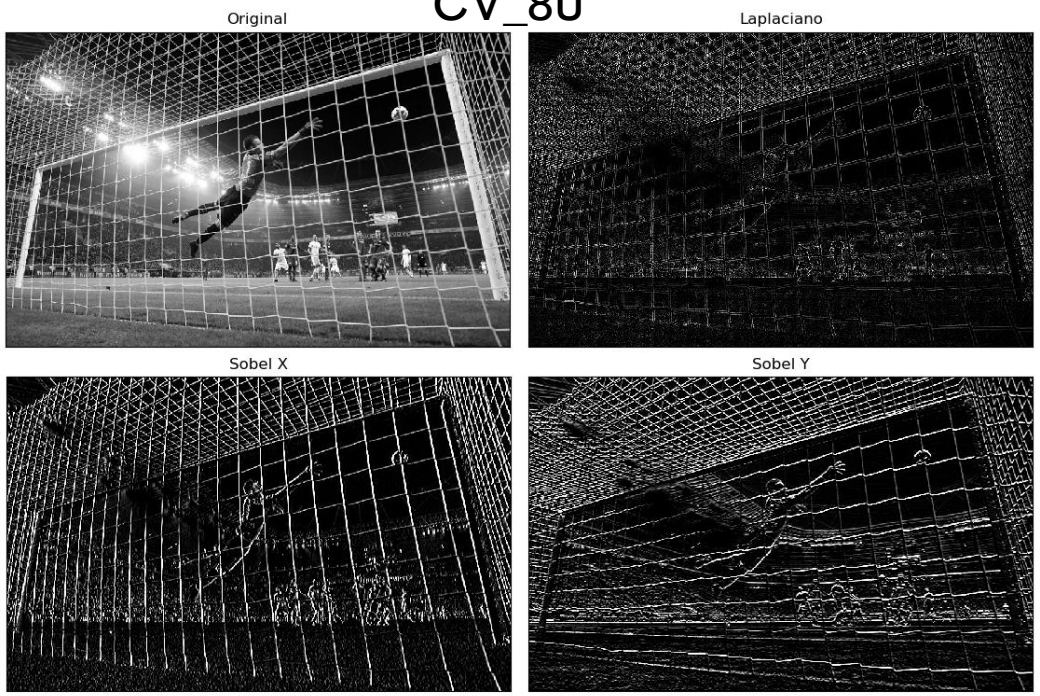
- ¿Qué pasa en los bordes de la imagen?
- Opciones:

1. **Cero:** establecer todos los píxeles fuera de la imagen de origen en 0
2. **Constante (color del borde):** establece todos los píxeles fuera de la imagen de origen en un valor de borde especificado
3. **Replicado (clamp):** repetir los píxeles del borde indefinidamente;
4. **Envolver (cíclica) (repetición o mosaico):** bucle "alrededor" de la imagen en una configuración "toroidal";
5. **Espejo:** reflejar píxeles en el borde de la imagen;
6. **Extender:** extiende la señal restando la versión reflejada de la señal del valor de píxel de borde.

CV_64F

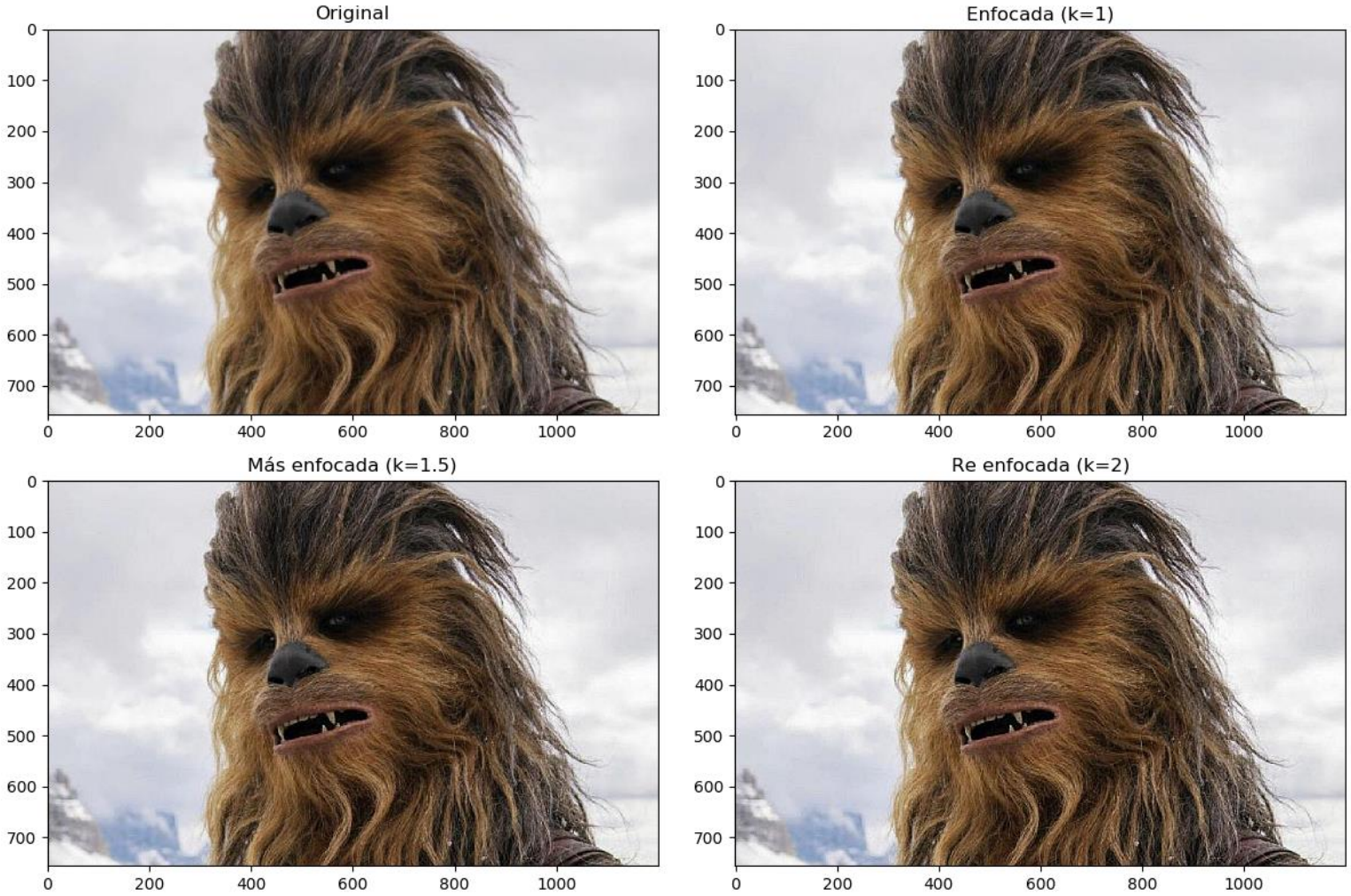


CV_8U



EJEMPLO

- En los ejemplos de la derecha el tipo de datos de salida es cv.CV_8U. Pero hay un problema:
  - La transición de negro a blanco (pendiente positiva) tiene un valor positivo.
  - Transición de blanco a negro (pendiente negativa) tiene valor negativo.
- Cuando se convierte datos a UINT8, todas las pendientes negativas se hacen cero. En palabras simples, echas de menos esa ventaja.
- Si se desea detectar ambos bordes, la mejor opción es mantener el tipo de datos de salida en algunas formas superiores, como cv.CV_16S, cv.CV_64F, etc., tomar su valor absoluto y luego volver a convertirlo en cv.CV_8U.



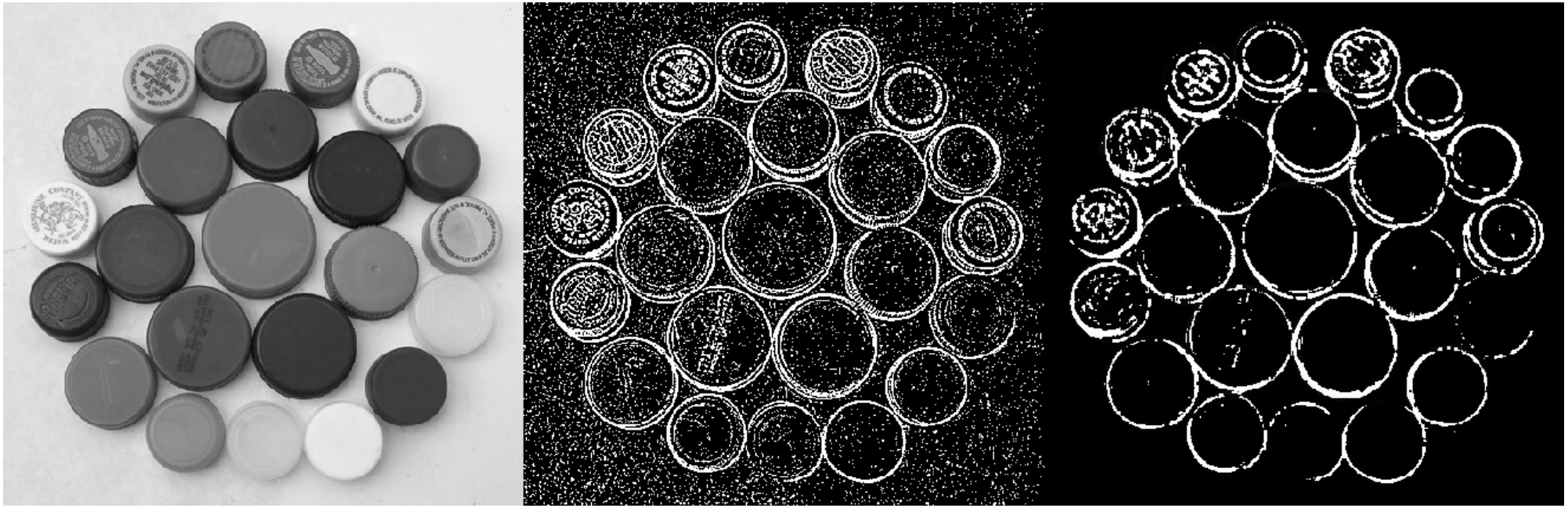
UNSHARP MASKING

- Es un algoritmo que tiene enfocar o mejorar los bordes con un filtro de suavizado
  1. Desenfoque de la imagen con un filtro suavizador (ej. Gaussiano)
     $$f(x,y) \rightarrow f_B(x,y)$$
  2. Resta de la imagen original con la suavizada (sobreviven componentes de alta frecuencia)
     $$m(x,y) = f(x,y) - f_B(x,y)$$
  3. Agregar esa máscara a la imagen original (realce de las componentes de alta frecuencia)
     $$g(x,y) = f(x,y) + k.m(x,y)$$

  De manera genérica:

  $$\boldsymbol{g(x,y) = (k+1).f(x,y) - k.f_B(x,y)}$$

  $k = 1$: *Máscara de desenfoque (unsharp masking)*
  $k > 1$: *Filtrado de alto impulso (highboost filtering)*



DOG (DIFFERENCE OF GAUSSIAN)

- De los filtros de suavizado Gaussianos sabemos que el desvío estándar incide directamente en el grado de suavizado.
  - A mayor sigma mayor nivel de desenfoque
- Por lo tanto si utilizamos dos sigma distintos y aplicamos el suavizado a una misma imagen para después restar las salidas esto actuará como un filtro pasabanda

$$DoG = Gauss_{low_{sigma}} - Gauss_{high_{sigma}}$$

FILTROS NO LINEALES

Realizan operaciones no lineales entre pixeles vecinos para obtener el pixel de salida. Dentro de esta familia los utilizados mas frecuentemente son:

- Filtros de mínima y máxima

$$I'(u,v) = \min_{(i,j)\in R} \{I(u+i, v+j)\}$$
$$I'(u,v) = \max_{(i,j)\in R} \{I(u+i, v+j)\}$$

- Filtro de Mediana

$$I'(u,v) = \text{median}_{(i,j)\in R}\{I(u+i, v+j)\}$$

Para el caso en que el num. de elementos sea par se toma el promedio de los dos centrales generando así un nivel de intensidad nuevo que no estaba presente en la imagen.

- Filtro de Mediana ponderada

Se multiplica la región por una ventana de pesos, generalmente ponderando los pixeles centrales.

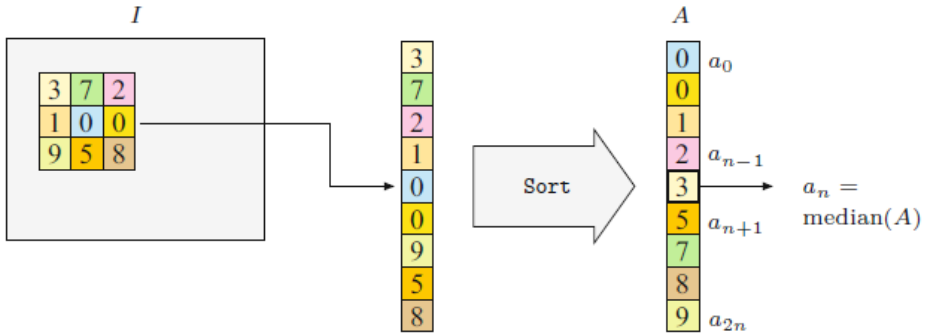


Fig. 1: Ejemplo de un filtro de mediana clásico

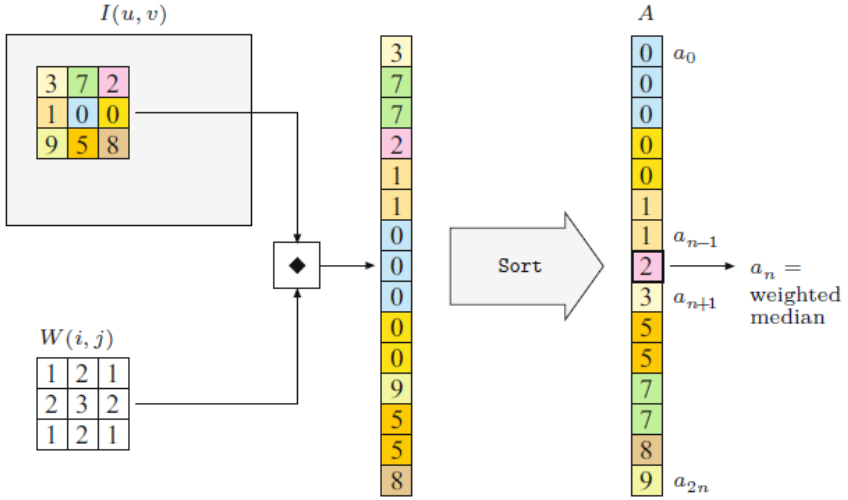


Fig. 2: Ejemplo de un filtro de mediana ponderada

FILTROS NO LINEALES

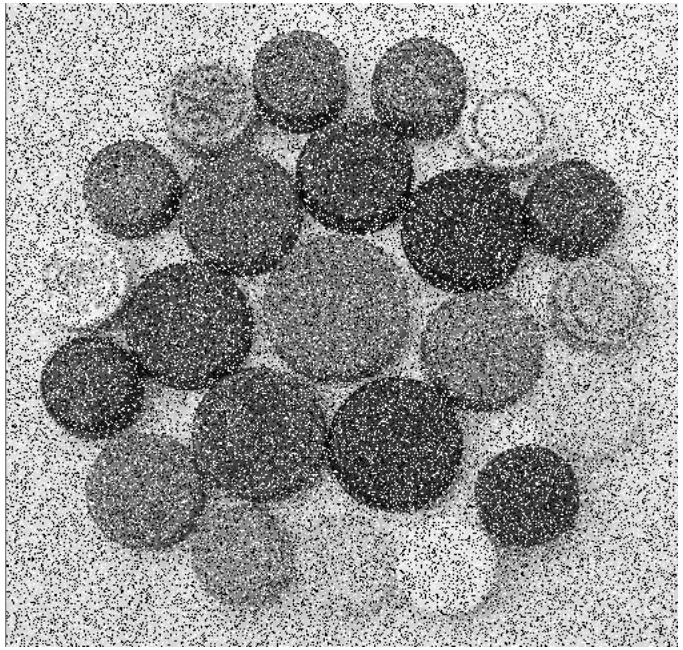
Fig.1: Imagen con ruido Salt & Pepper

Fig.2: Filtrado con desenfoque uniforme

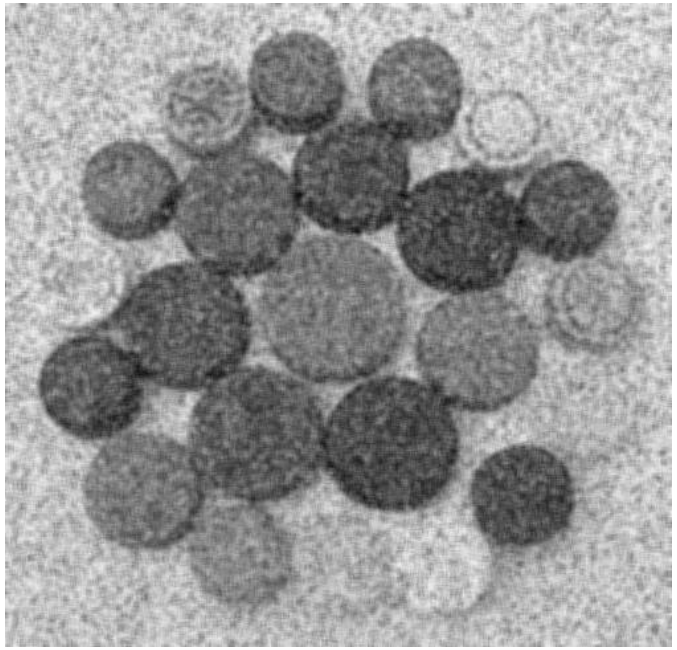
Fig.3: Filtrado con filtro de mínima (izquierda) y máxima (derecha)

Fig.4: Filtrado con filtro de mediana (izquierda) y mediana ponderada

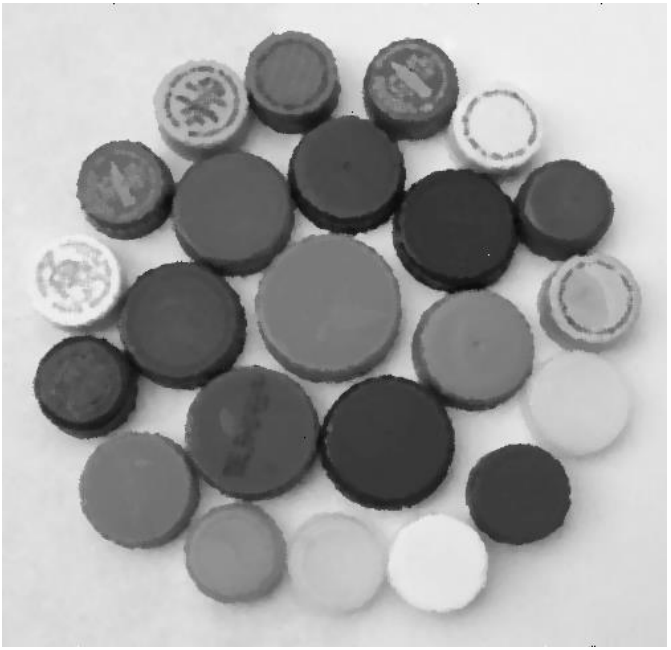
$$W = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



Original con ruido



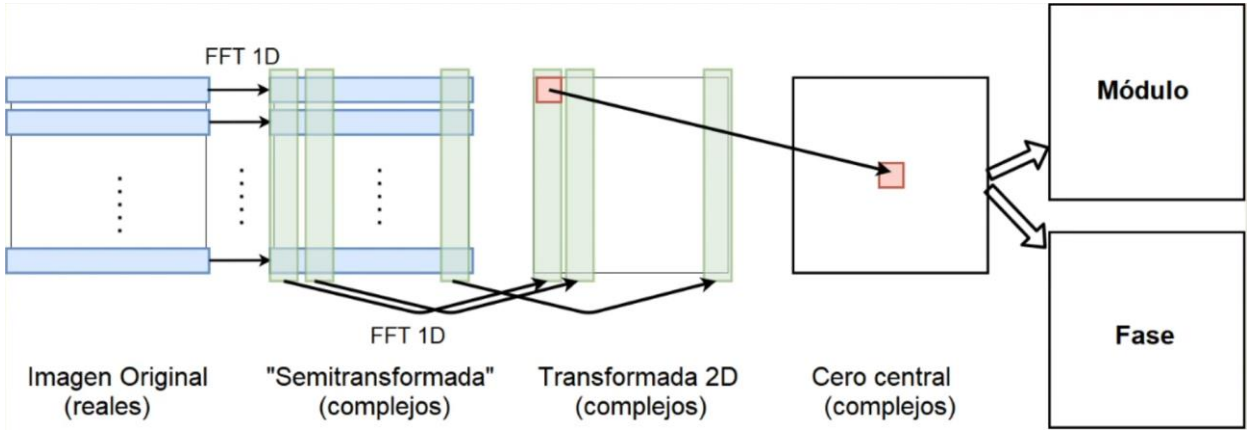
Filtro media 5x5



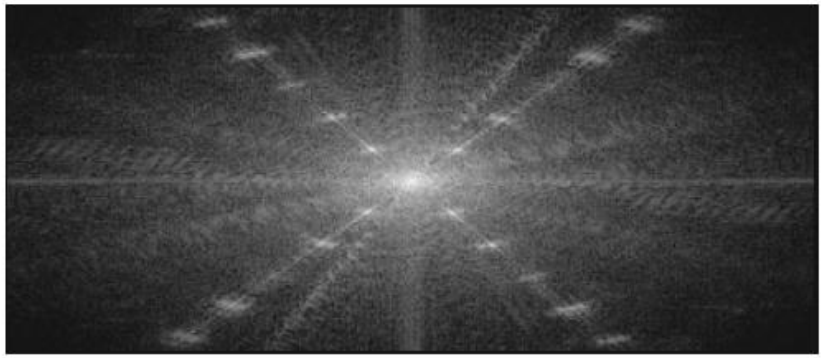
Filtro mediana 5x5

FILTROS DE RUIDO IMPULSIVO

- Ruido impulsivo: **Salt & Pepper**
  - **Media:** Valor Promedio
  - **Mediana:** Valor medio del conjunto ordenado
  - **Moda:** Valor más repetido
- Ej: $\begin{bmatrix} 1 & 9 & 5 \\ 6 & 9 & 8 \\ 4 & 10 & 3 \end{bmatrix} \rightarrow Media: \mathbf{6,1}\ / Mediana: \mathbf{6} / Moda: \mathbf{9}$
- Media: (1+9+5+6+9+8+4+10+3)/9
- Mediana: [1;3;4;5;6;8;9;9;10] / Moda: 9



(a)



(b)

FOURIER

- La transformada de Fourier se utiliza:
  1. Para analizar el comportamiento en frecuencia de sistemas y señales.
  2. En imágenes (2D DFT), para encontrar las componentes de frecuencia. El algoritmo que implementa esto de forma eficiente es la Fast Fourier Transform (FFT).

  $$H(k) = \frac{1}{\sqrt{N}} \sum_{x=0}^{N-1} h(x) e^{-j\frac{2\pi kx}{N}} ; vale\ \forall\ k, pero\ solo\ tiene\ sentido\ en\ k \in \left[-\frac{N}{2}, \frac{N}{2}\right]$$

  - $DFT \rightarrow N^2 operaciones;\ \ FFT\ \rightarrow N log_2 N$

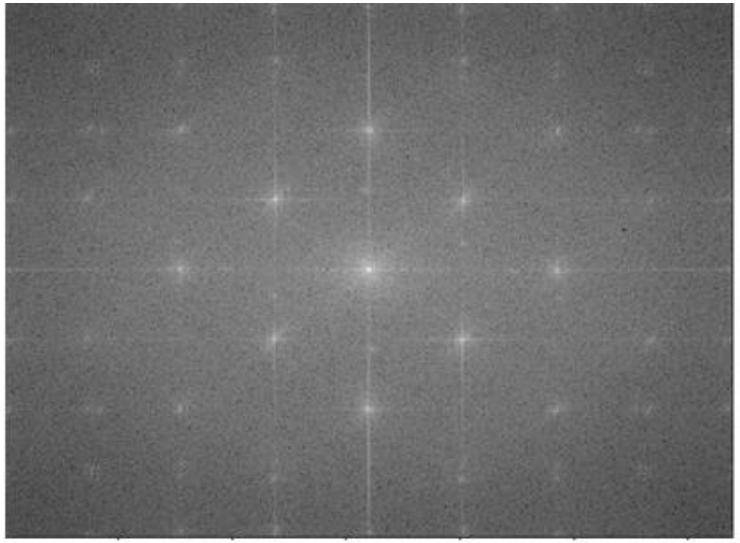
  Para 2 dimensiones nos queda:

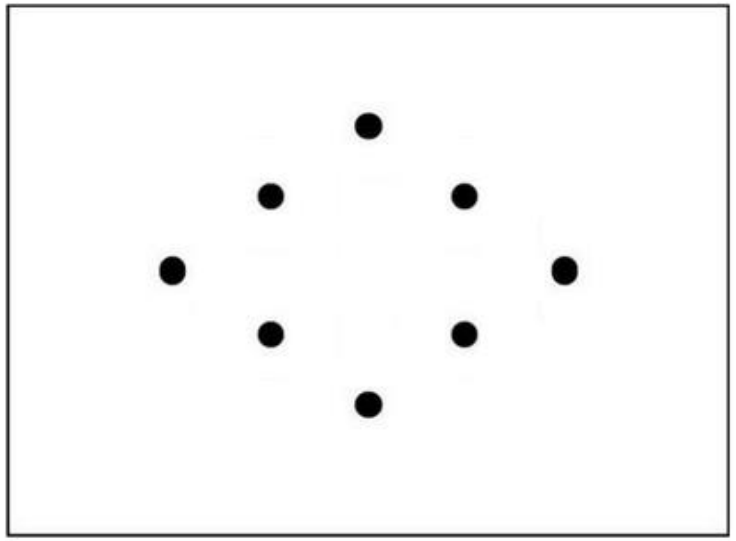
$$G(m,n) = \frac{1}{\sqrt{MN}} \cdot \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} g(u,v) \cdot e^{-\mathrm{i}2\pi \frac{mu}{M}} \cdot e^{-\mathrm{i}2\pi \frac{nv}{N}}$$

- ¿Dónde es que varía fuertemente la intensidad en imágenes?
  1. En bordes
  2. En donde hay ruido
- Por lo tanto, bordes y ruido son componentes de alta frecuencia.
- Algunas propiedades:
  - Superposición: $f_1(x) + f_2(x) \rightarrow F_1(w) + F_2(w)$
  - Inversa: $f(-x) \rightarrow F^*(w)$
  - Convolución: $f(x) * h(x) \rightarrow F(w)H(w)$
  - Correlación: $f(x) \otimes h(x) \rightarrow F(w)H^*(w)$
  - Multiplicación: $f(x)h(x) \rightarrow F(w) * H(w)$
  - Escalamiento: $f(ax) \rightarrow \frac{1}{a} F\left(\frac{w}{a}\right)$
  - Diferenciación: $f'(x) \rightarrow jwF(w)$

Original

Magnitud del espectro





Máscara

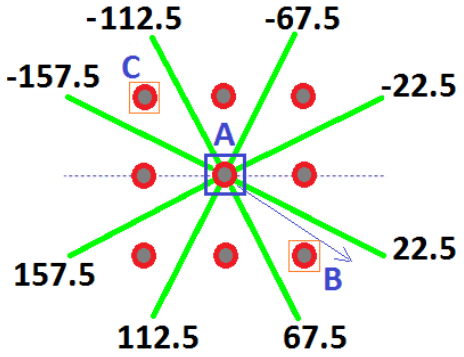
Transformada inversa

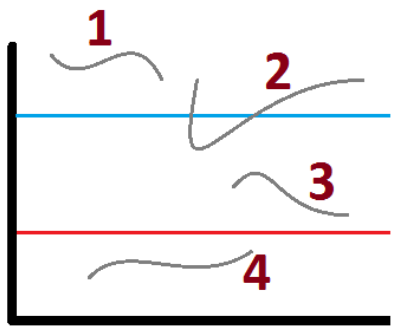
FOURIER: APLICACIONES

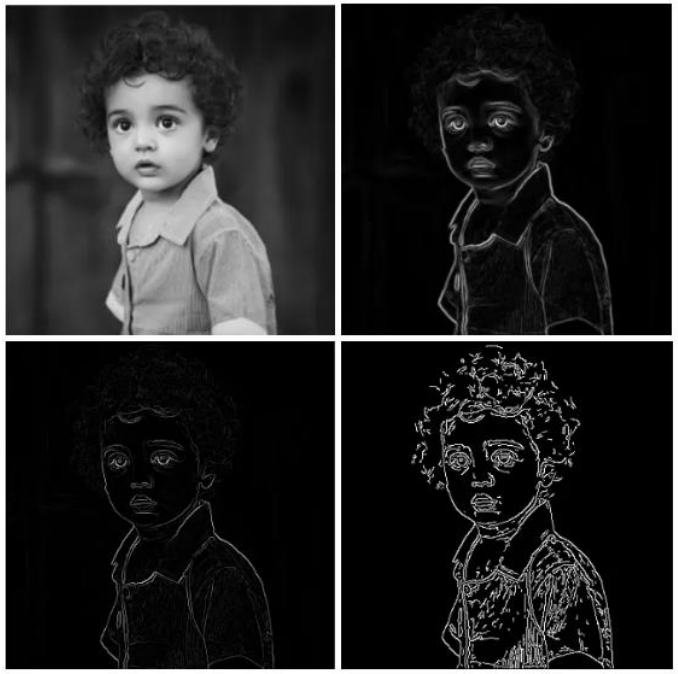
Trabajar en el dominio de la frecuencia espacial permite resolver problemas de forma sencilla que serian muy complejos si solo usamos la imagen original.

1. Transformamos la imagen al dominio de frecuencia espacial.
2. Detectamos las componentes que corresponden con el patrón de impresión.
3. Aplicamos una mascara binaria para eliminar esas componentes de la escena
4. Hacemos la transformada inversa para volver al dominio original

DETECTOR DE BORDES DE CANNY







- Algoritmo desarrollado por John F. Canny en 1986
- Consta de varias etapas:
  1. **Reducción de ruido**
     - Canny usa un filtro Gaussiano, en general de 5x5

$$H_{i,j} = \frac{1}{2\pi\sigma^2} e^{-\frac{(i-(k+1))^2+(j-(k+1))^2}{2\sigma^2}} \quad \forall\ 1 \le i,j \le (2k+1)$$

  2. **Encontrar el gradiente de intensidad de la imagen**
     - Se puede utilizar un operador de Sobel (filtro direccionable) y obtener Gx, Gy. Luego,

$$G = \sqrt{G_x^2 + G_y^2} : Magnitud \qquad \theta = \operatorname{atan}\left(\frac{G_y}{G_x}\right) : Dirección$$

  3. **Supresión de no-máximos**
     - Técnica de adelgazamiento de bordes. Se verifica para cada píxel si hay un máximo local vecino en la dirección del gradiente o no. Si es un máximo local se retiene, sino se elimina.
       1. Los vecinos estarán a 0°,45°,90°,135° → los valores del gradiente tienen que redondearse a esos rangos.
       2. Una vez definida la dirección:

$$Si\ A > C \wedge A > B \Rightarrow Se\ retiene\ A, caso\ contrario\ se\ suprime$$

  4. **Umbral con histéresis**
     - Del paso anterior se obtendrán distintos bordes, algunos más brillantes que otros. Para saber cuáles son bordes reales y cuales no, Canny usa histéresis. Se definen dos umbrales: Alto y Bajo
       1. Cualquier borde con intensidad mayor que 'Alto' es borde seguro, se retiene
       2. Cualquier borde con intensidad menor que 'Bajo' no es borde, se descarta
       3. Los bordes entre los umbrales 'Alto' y 'Bajo' se clasifican como bordes solo si están conectados a un borde seguro, de otro modo se descarta.

TP2 - CLASE 3

- Implementar la función `create_gauss_filter(h, w, k_size, sigma)` para crear filtros gaussianos para filtrado espectral. Debe retornar un filtro gaussiano de tamaño HxW en dominio espacial y su transformada de Fourier.
  1. Graficar ambas representaciones para diferentes tamaños de kernel y sigma. Aplicar el filtro una imagen para validar el funcionamiento en el dominio espectral.
  2. Usando el método descripto en el paper "Image Sharpness Measure for Blurred Images in Frequency Domain" comparar el resultado de un filtrado por convolución con el filtrado espectral.
  3. Repetir la comparación usando uno de los métodos descriptos en el apéndice del paper "Analysis of focus measure operators in shape-from-focus"