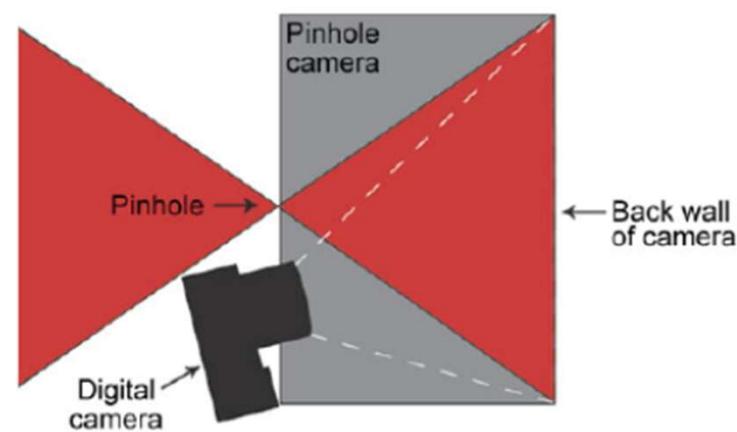


Visión Computacional

Ivan Sipiran

Cámara Pinhole

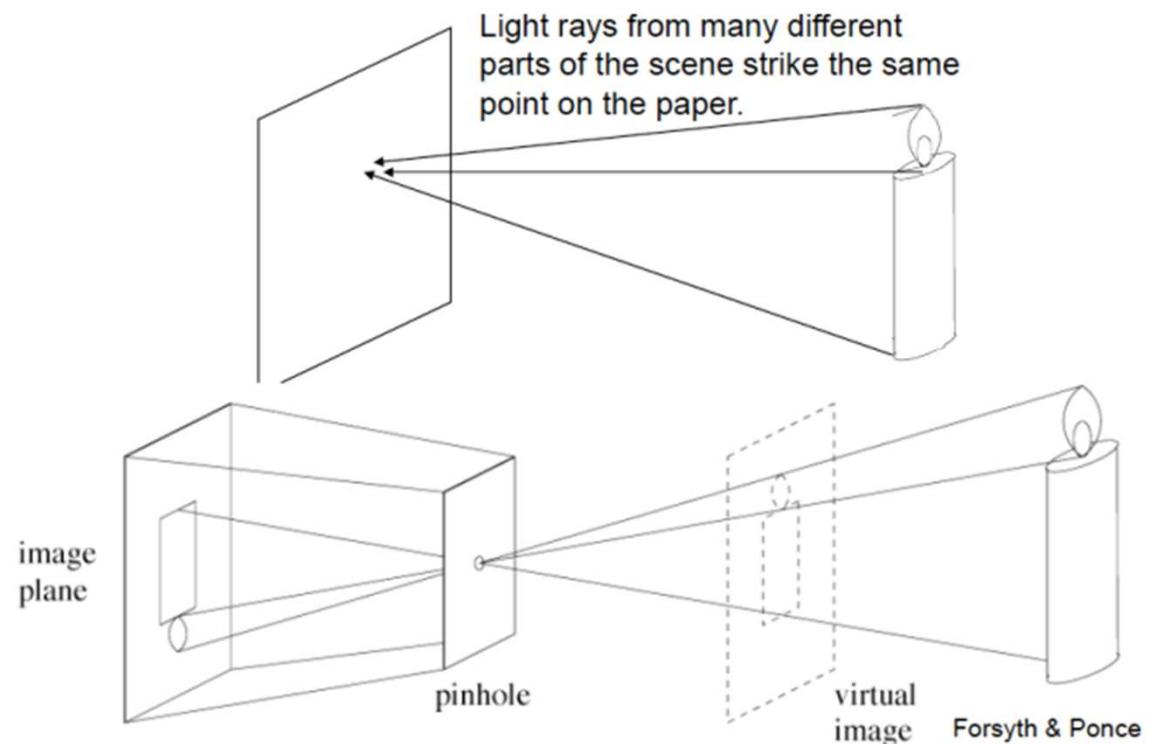
Camara DIY



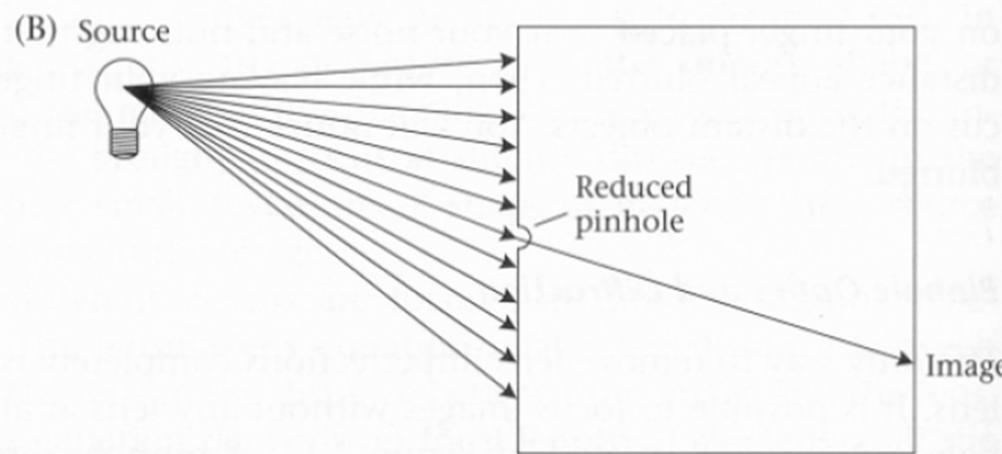
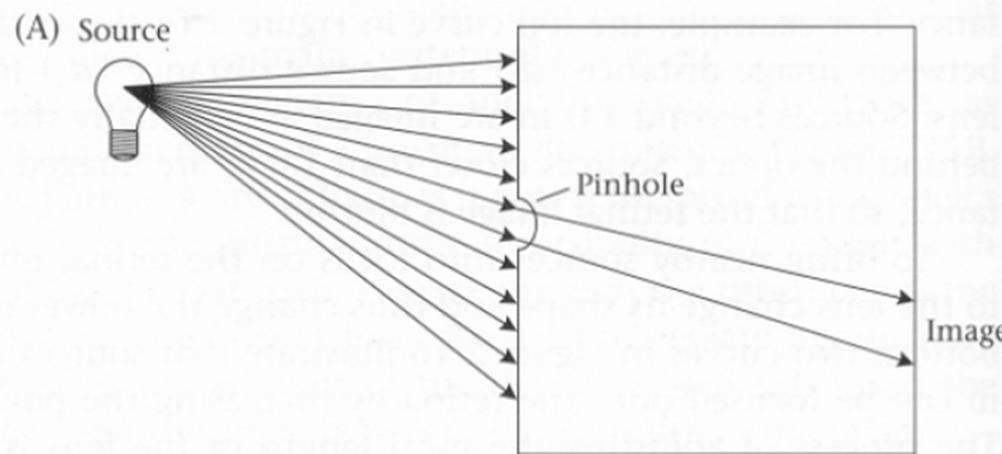
Cámara Pinhole

Cómo funciona?

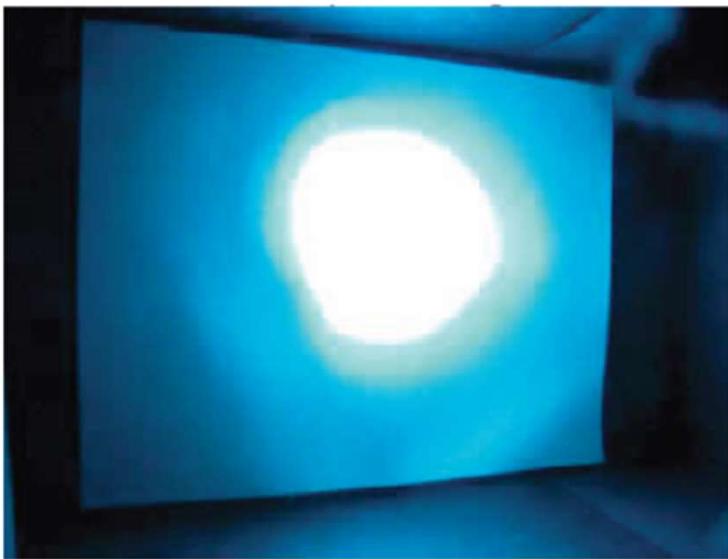
Solo permite rayos desde
Un punto de la escena a un
Punto en el papel



Cámara Pinhole

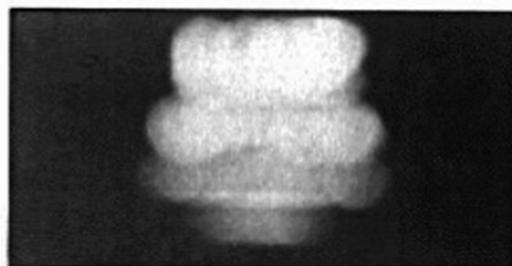


Cámara Pinhole



Cámara Pinhole

Mientras más pequeña la apertura, la imagen es más nítida



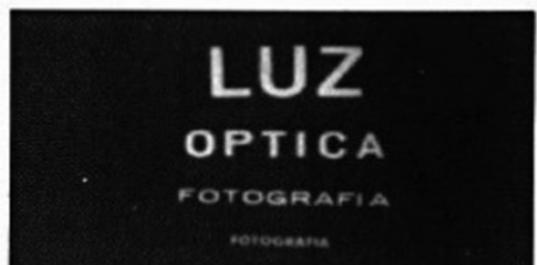
2 mm



1 mm



0.6mm



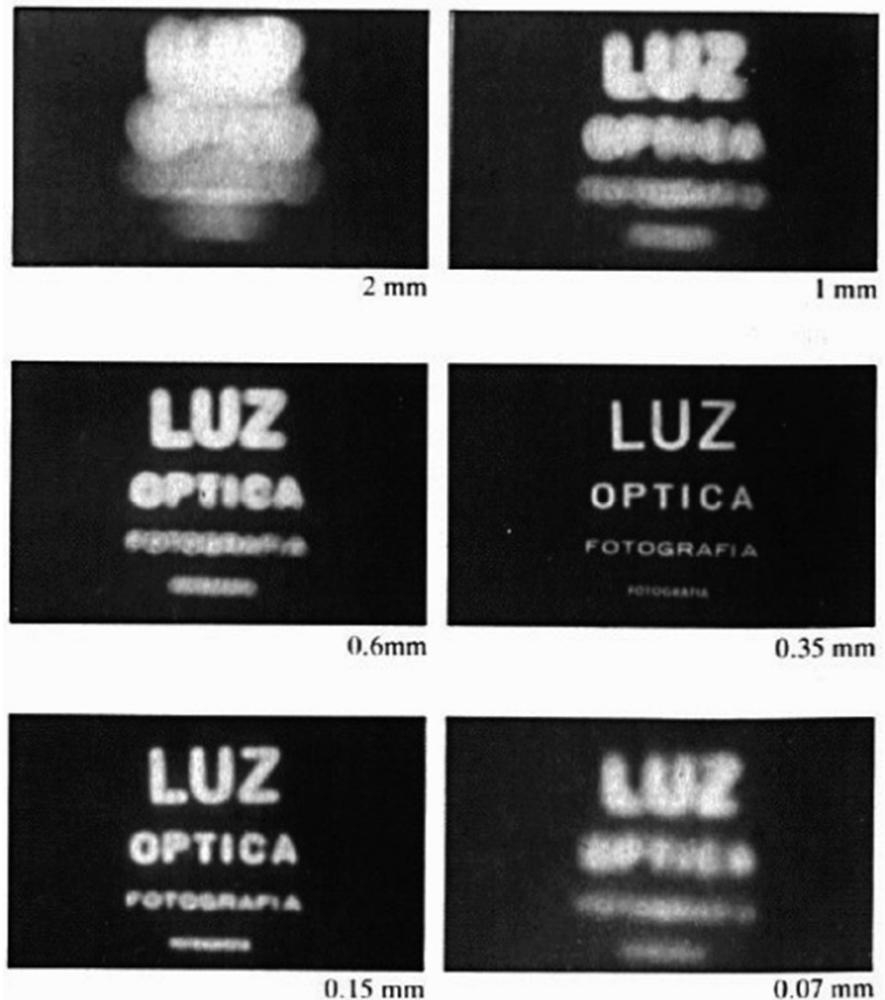
0.35 mm

Cámara Pinhole

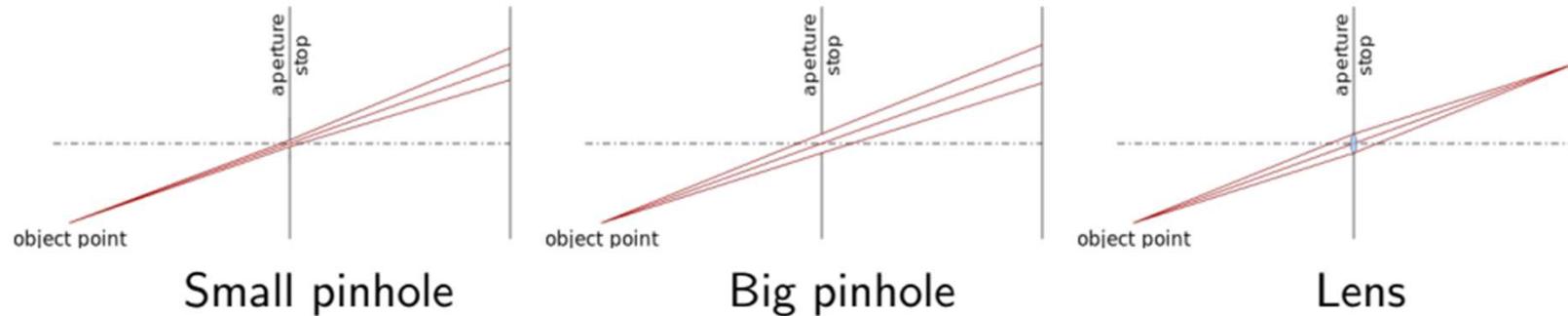
Mientras más pequeña la apertura, la imagen es más nítida

Hasta un cierto punto (tiene que entrar luz)

Efectos de difracción



Cámara Pinhole - Lentes

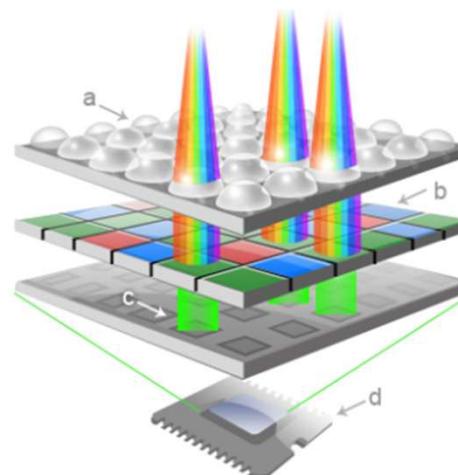


Lente ayuda a enfocar la luz

Existe una distancia a la cual los objetos están bien enfocados

Se puede cambiar con forma de lente

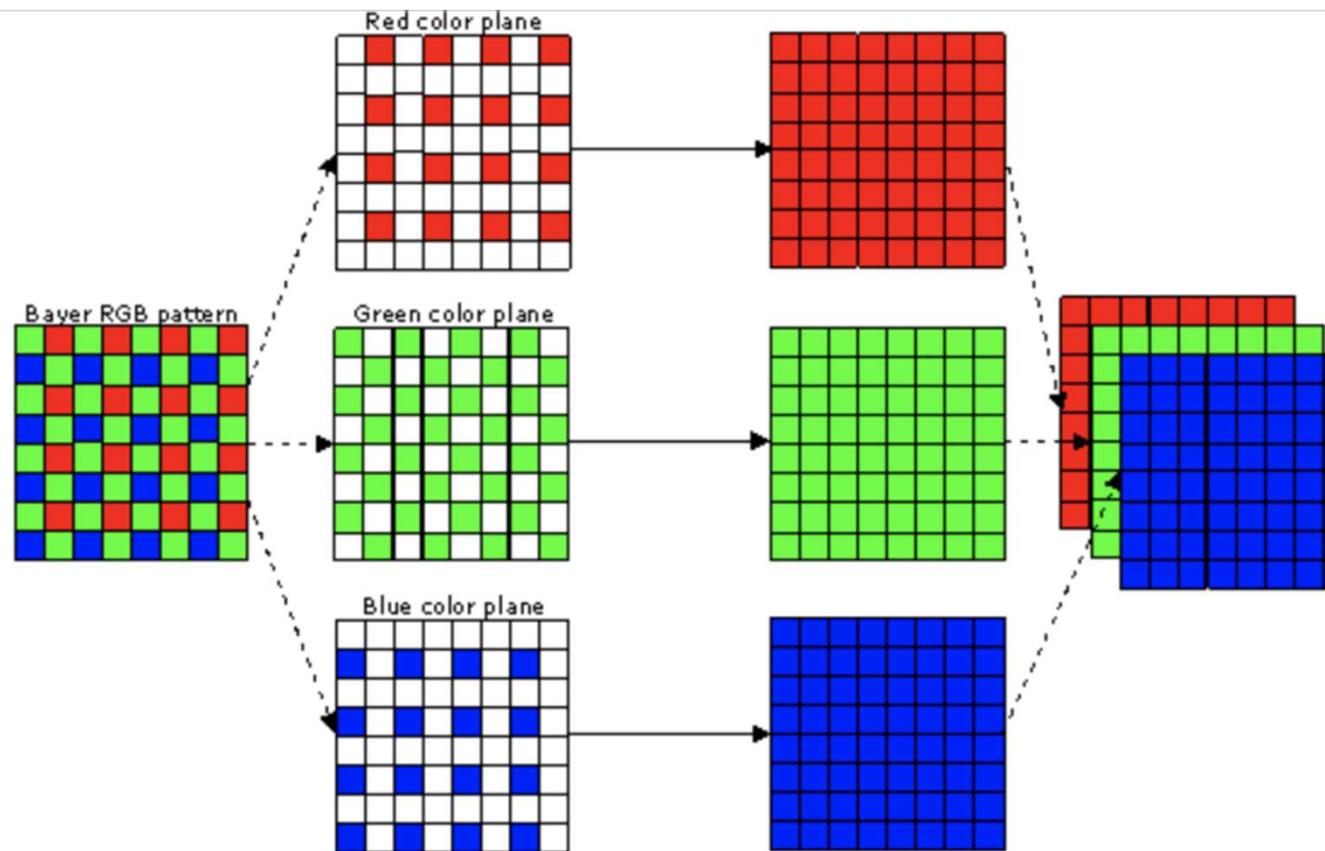
Cámara Digital



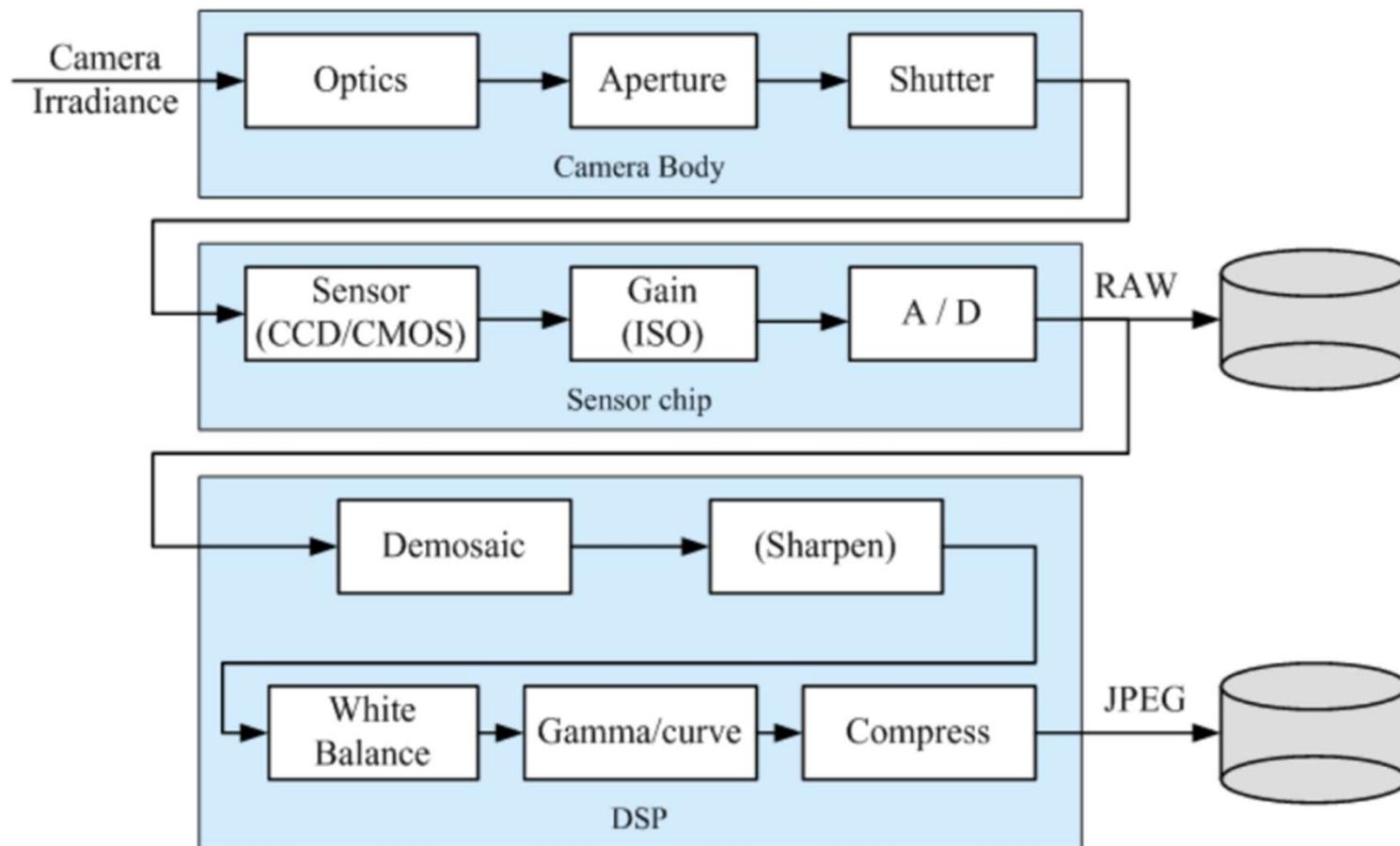
La cámara digital reemplaza el film con un sensor de array

Cada celda en el array es un diodo sensible a la luz que convierte fotones en electrones

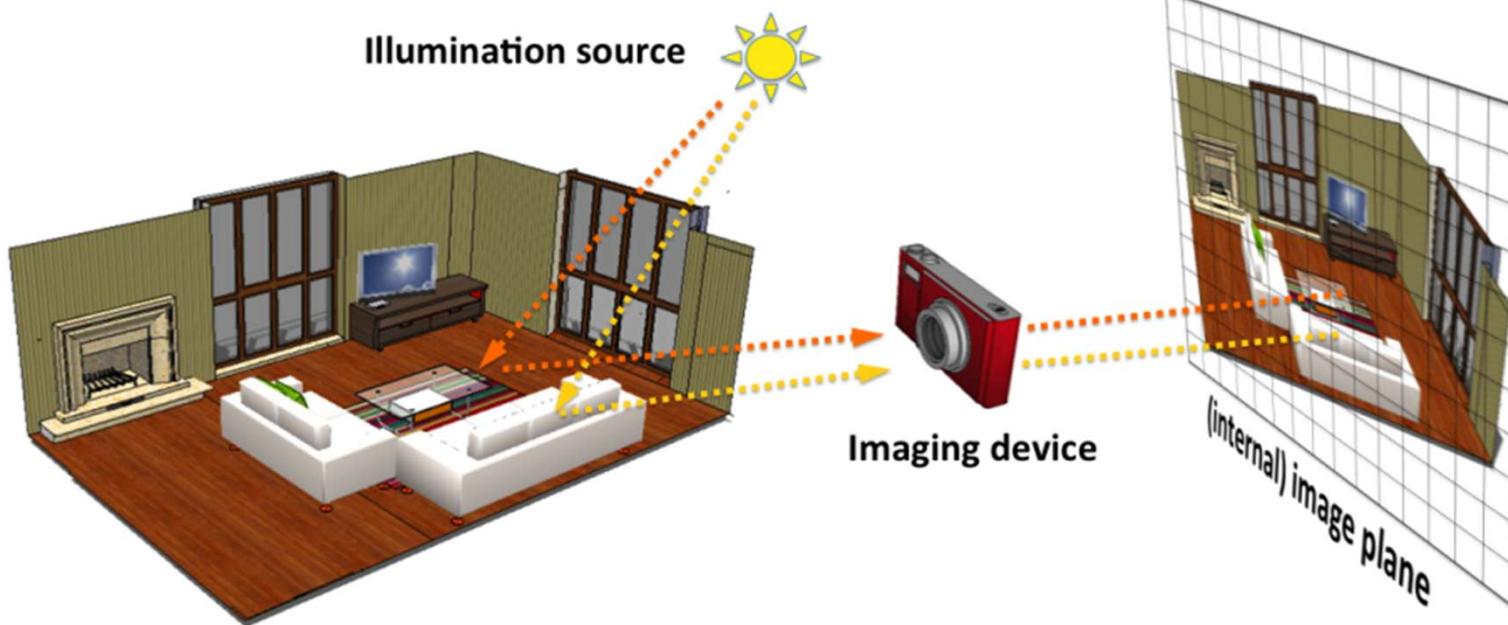
Cámara Digital



Cámara Digital



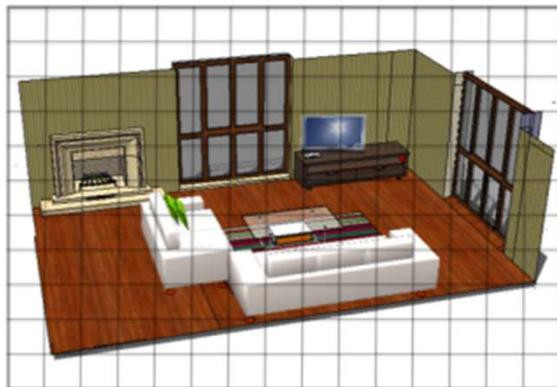
Formación de la imagen



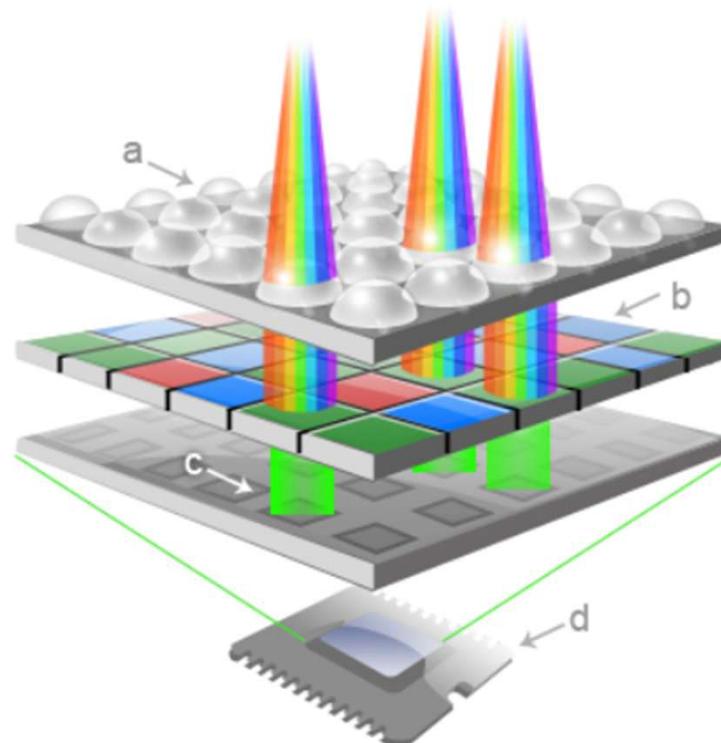
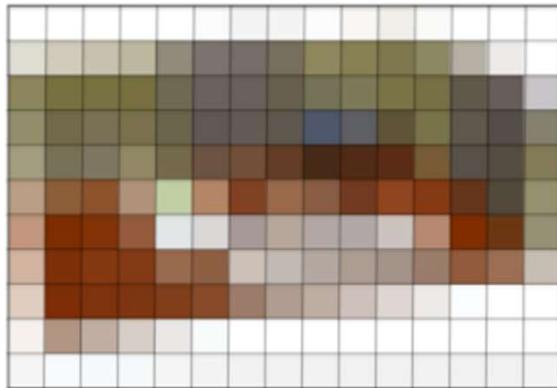
Proceso depende de:

- Condiciones de luz
- Geometría de la escena
- Propiedades de superficie
- Óptica de la cámara

Formación de la imagen



Sampling and quantization



<http://pho.to/media/images/digital/digital-sensors.jpg>

Imagen digital

Imagen es una matriz de valores enteros

Típicamente lo denotamos con I



Imagen digital

Imagen es una matriz de valores enteros

Típicamente lo denotamos con I

$I(i, j)$ se llama intensidad



Imagen digital

Imagen es una matriz de valores enteros

Típicamente lo denotamos con I

$I(i,j)$ se llama intensidad



Imagen digital

Imagen es una matriz de valores enteros

Típicamente lo denotamos con I

$I(i,j)$ se llama intensidad



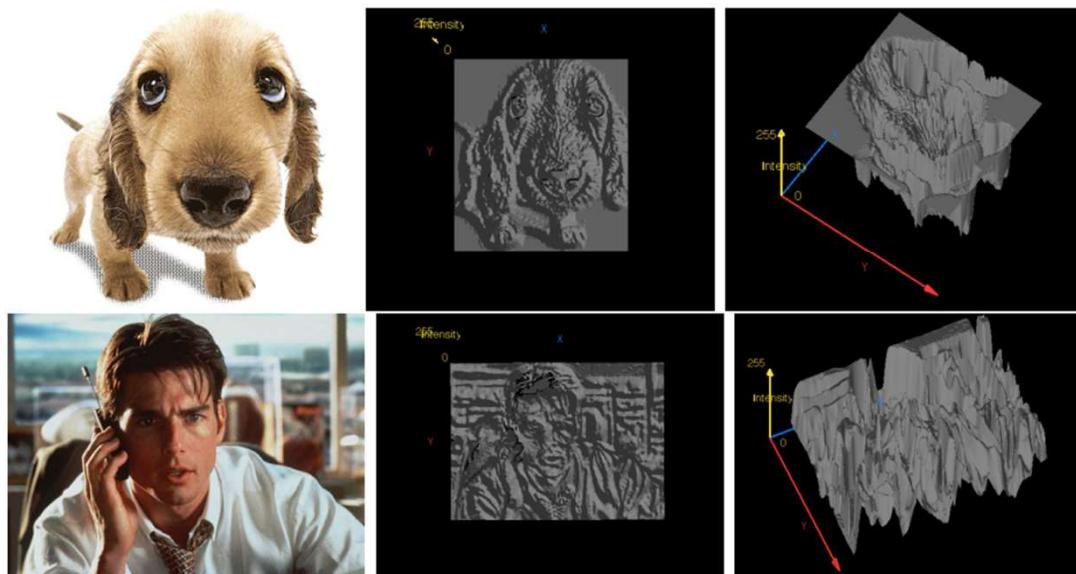
Imagen digital

- Imagen es una matriz de valores enteros
 - Típicamente lo denotamos con I
 - $I(i, j)$ se llama intensidad
 - Matriz puede ser de tamaño $m \times n$
(escala de grises)
 - O $m \times n \times 3$ (color)



Intensidad

- Una imagen como una función $f: \mathbb{R}^2 \rightarrow \mathbb{R}$ dando la intensidad para una posición dada
- Intensidad 0 es negro y 255 blanco



Transformaciones

- Como en cualquier función, podemos aplicar operaciones

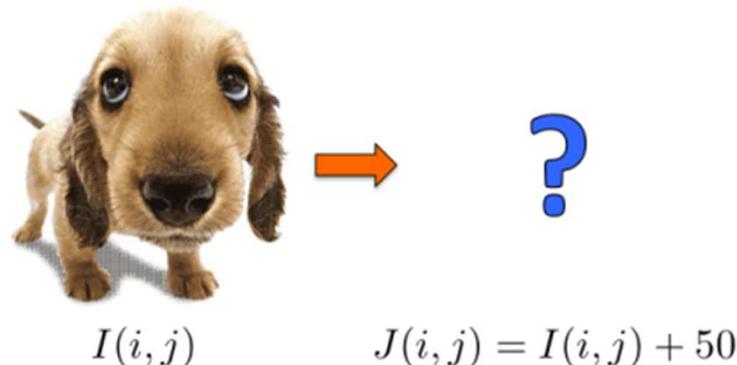


$$I(i, j)$$

$$J(i, j) = I(i, j) + 50$$

Transformaciones

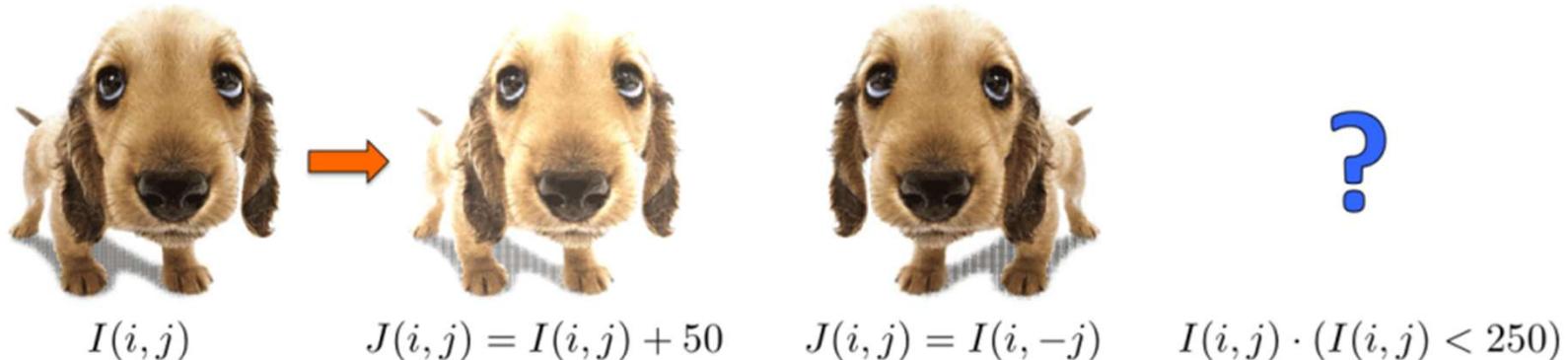
- Como en cualquier función, podemos aplicar operaciones



- Hay operaciones especiales que nos interesan
 - Correlación y convolución

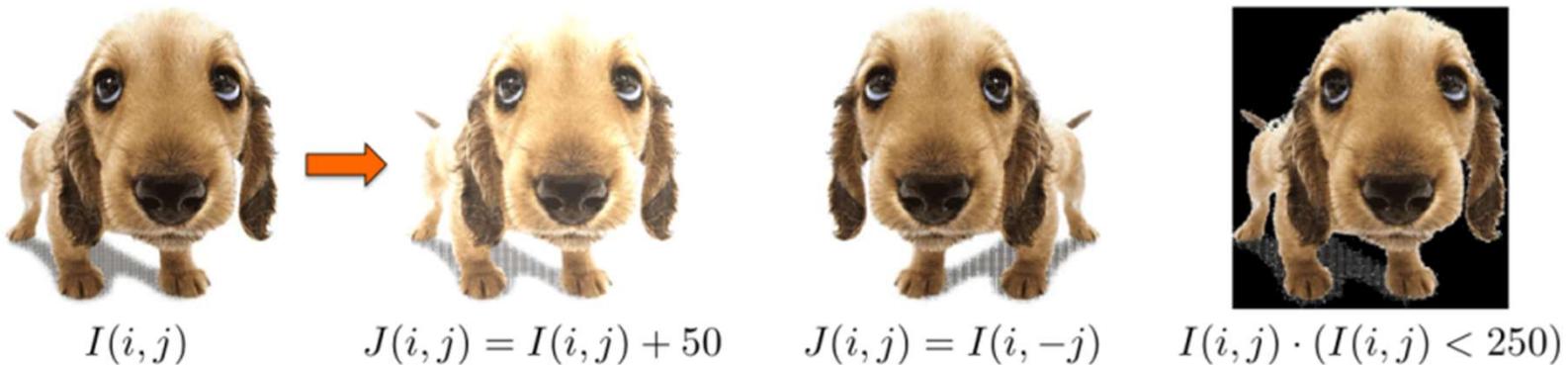
Transformaciones

- Como en cualquier función, podemos aplicar operaciones



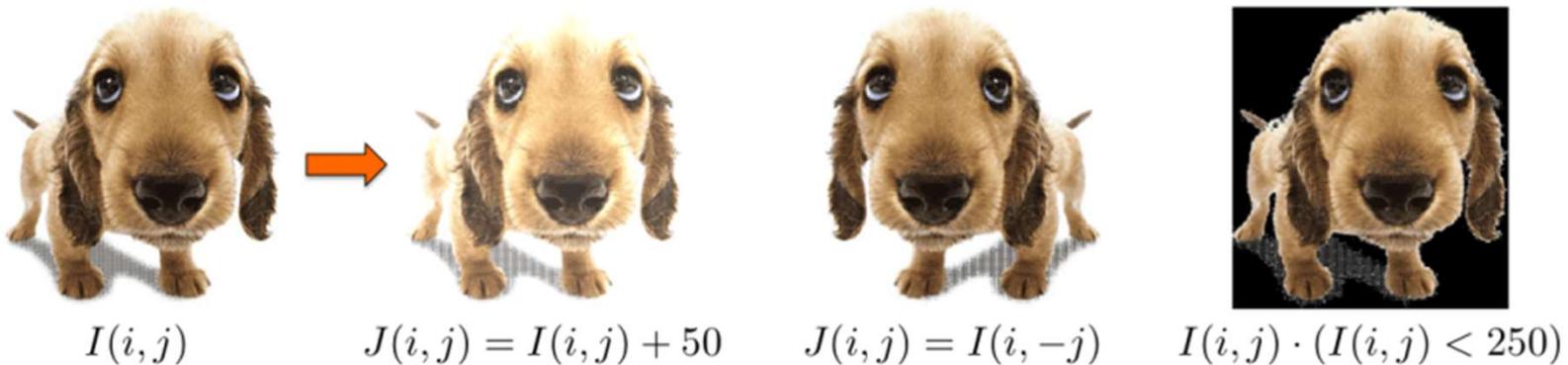
Transformaciones

- Como en cualquier función, podemos aplicar operaciones



Transformaciones

- Como en cualquier función, podemos aplicar operaciones



Filtros Lineales

Motivación

- Cómo podemos encontrar a Waldo?



Motivación

- Deslizar y comparar!
- En lenguaje formal: **filtering**

Motivación

- Dada una cámara y una escena fija, cómo se puede reducir el ruido?



Filtrado de imágenes

- Modificar los pixels en una imagen basado en alguna función de una vecindad local de cada píxel

10	5	3
4	5	1
1	1	7

Local image data

Some function



	7	

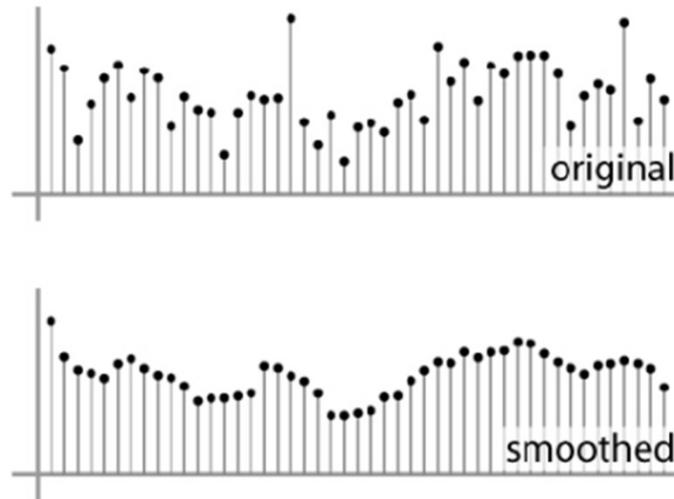
Modified image data

Aplicaciones de filtrado

- Realce de imágenes: eliminación de ruido
- Detectar patrones: matching de plantillas
- Extraer información: texturas, bordes, etc
- Redes neuronales convolucionales

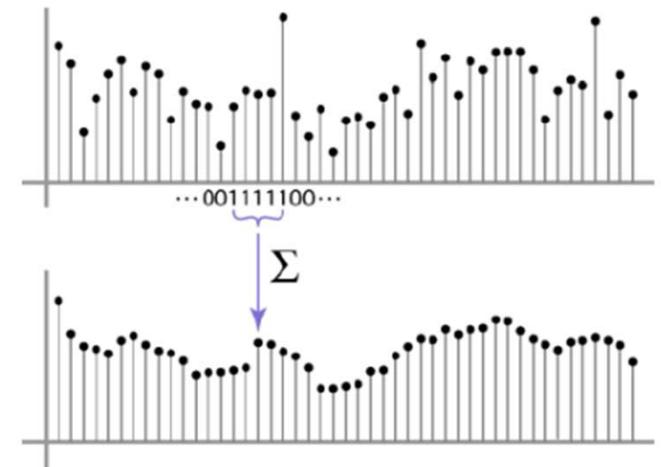
Ruido

- Lo más simple
 - Reemplazas cada píxeles por el promedio de sus vecinos
 - Se asume que píxeles vecinos son similares, y que el ruido es independiente



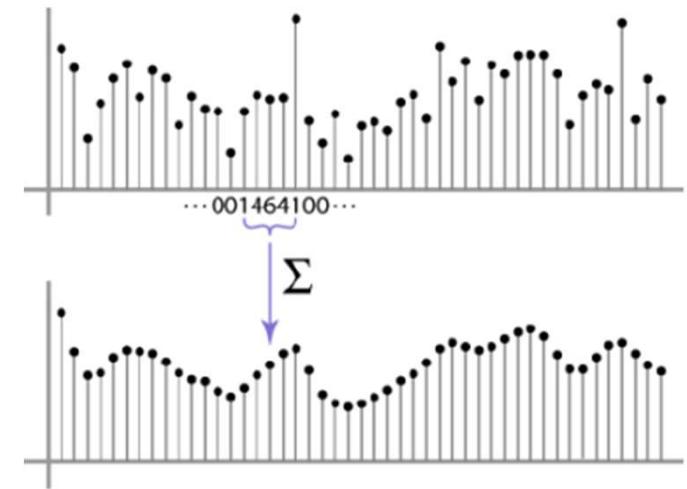
Ruido

- Lo más simple
 - Reemplazas cada píxels por el promedio de sus vecinos
 - Se asume que píxeles vecinos son similares, y que el ruido es independiente
 - Promedio móvil en 1D: $[1, 1, 1, 1, 1]/5$



Ruido

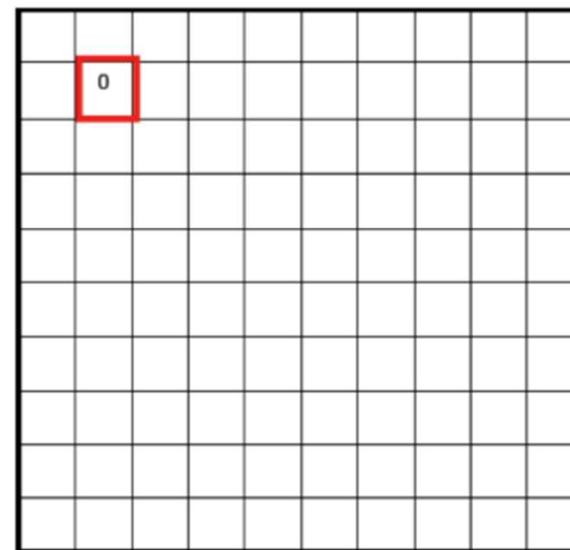
- Lo más simple
 - Reemplazas cada píxeles por el promedio de sus vecinos
 - Se asume que píxeles vecinos son similares, y que el ruido es independiente
 - Pesos no uniformes: $[1, 4, 6, 4, 1] / 16$



Promedio móvil en 2D

$$I(i,j)$$

$$G(i, j)$$



Promedio móvil en 2D

$$I(i,j)$$

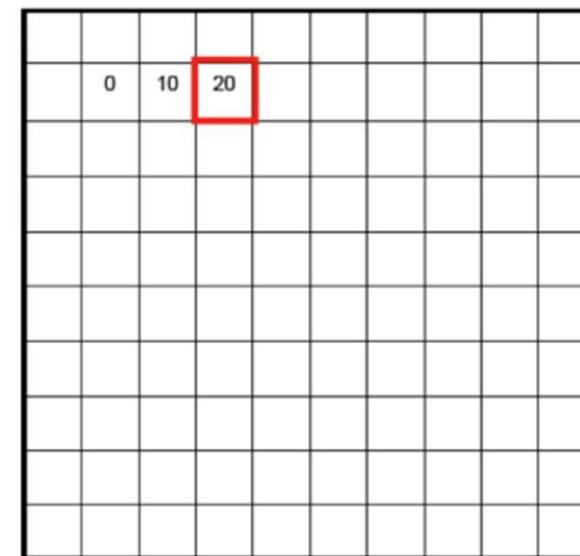
$$G(i, j)$$

0	10
---	----

Promedio móvil en 2D

$$I(i,j)$$

$$G(i, j)$$

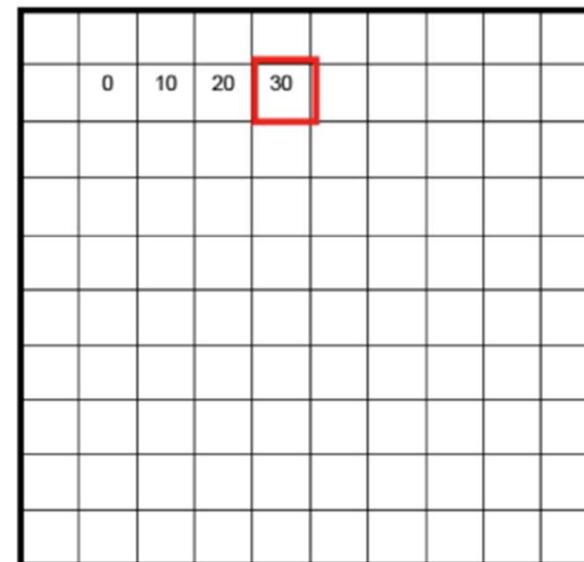


Promedio móvil en 2D

$I(i, j)$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$G(i, j)$



Promedio móvil en 2D

$$I(i,j)$$

$$G(i,j)$$

0	10	20	30	30
---	----	----	----	----

Promedio móvil en 2D

$$I(i,j)$$

$$G(i,j)$$

Filtrado lineal: correlación

- Combinaciones ponderadas de píxeles en pequeñas vecindades

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^k \sum_{v=-k}^k I(i+u, j+v)$$

Filtrado lineal: correlación

- Combinaciones ponderadas de píxeles en pequeñas vecindades

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^k \sum_{v=-k}^k I(i+u, j+v)$$

- El valor de salida es determinado como la suma ponderada de píxeles de entrada

$$G(i,j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u,v) \cdot I(i+u, j+v)$$

Filtrado lineal: correlación

- Combinaciones ponderadas de píxeles en pequeñas vecindades

$$G(i, j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^k \sum_{v=-k}^k I(i+u, j+v)$$

- El valor de salida es determinado como la suma ponderada de píxeles de entrada

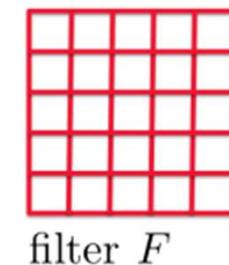
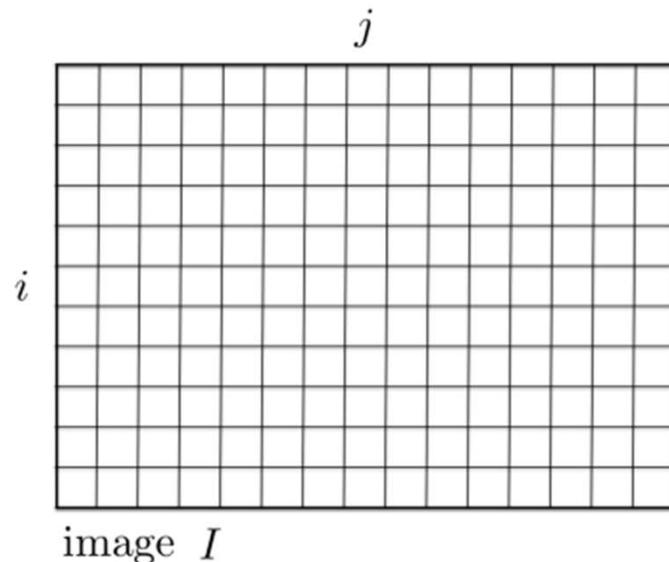
$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i+u, j+v)$$

- Los valores del **kernel** de pesos $F(u, v)$ se llaman coeficientes del filtro.
- Este operador se llama correlación

$$G = F \otimes I$$

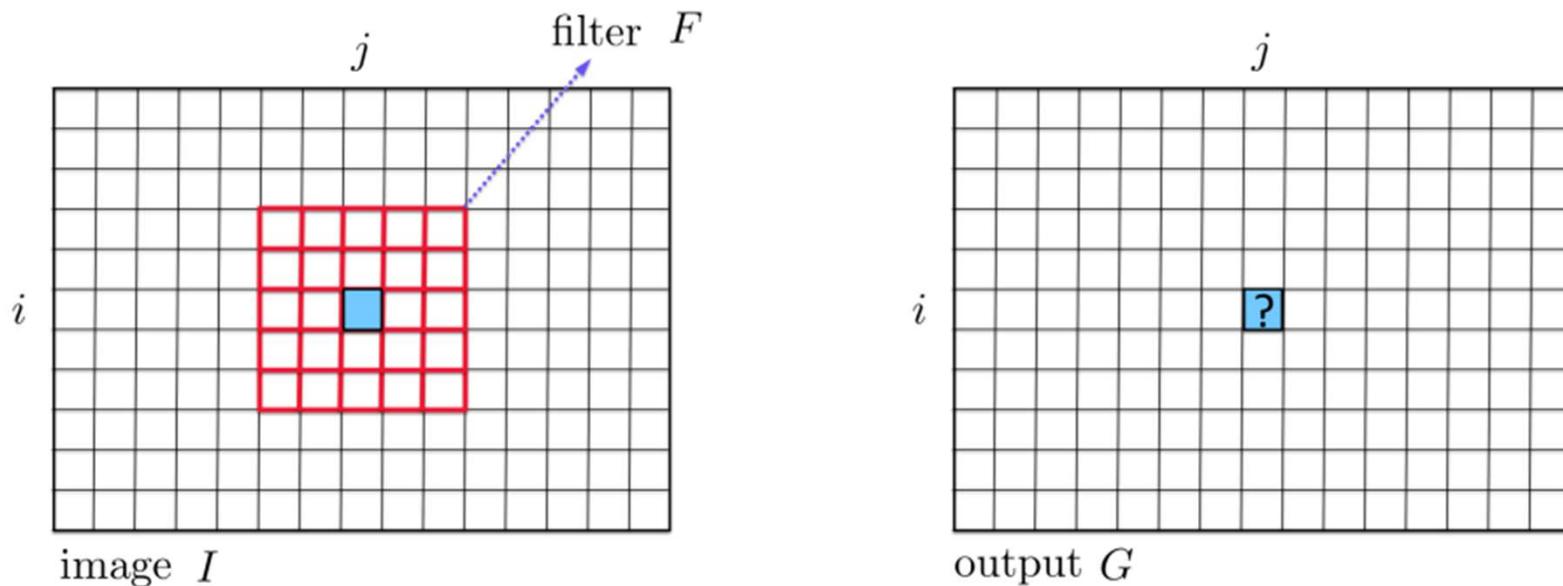
Filtrado lineal: correlación

- En la práctica es muy simple!



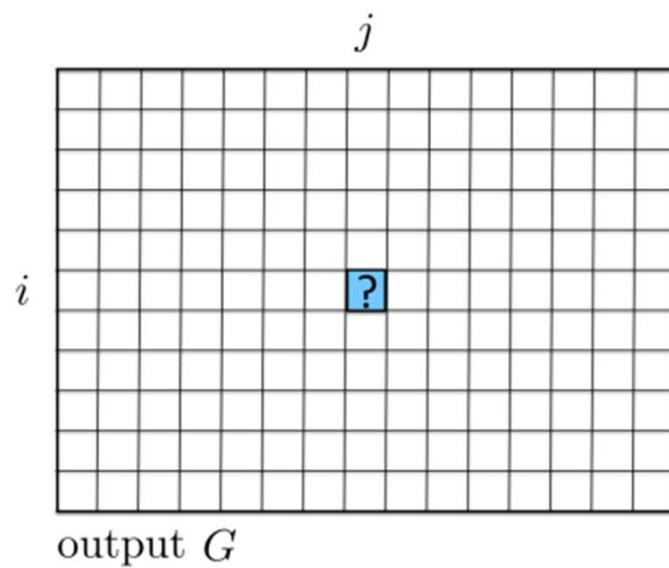
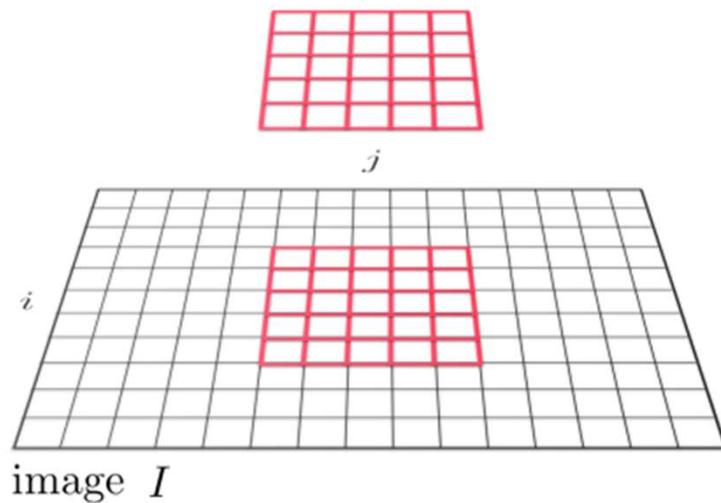
Filtrado lineal: correlación

- En la práctica es muy simple!



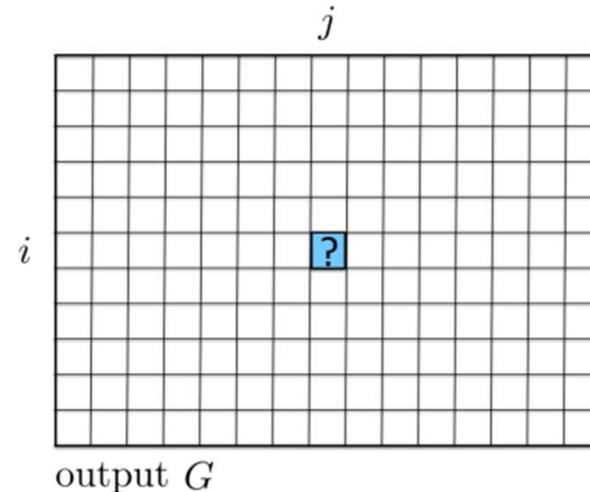
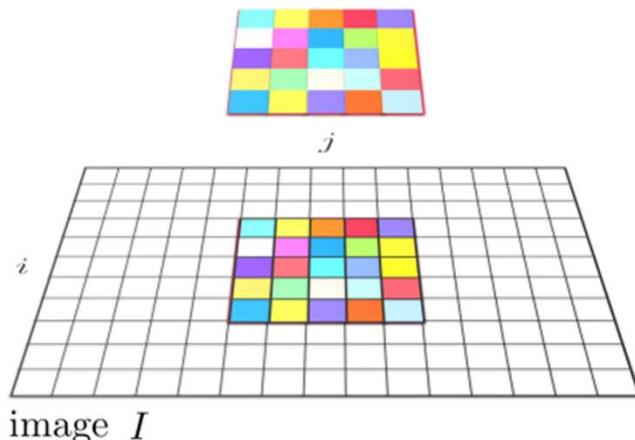
Filtrado lineal: correlación

- En la práctica es muy simple!



Filtrado lineal: correlación

- En la práctica es muy simple!

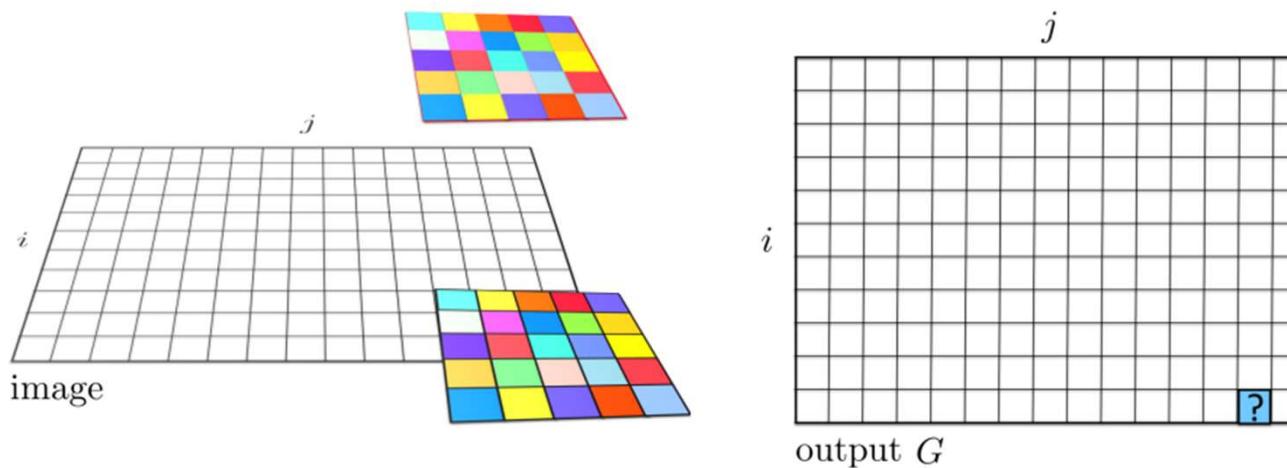


$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

$$G(i, j) = F(\square) \cdot I(\square) + F(\square) \cdot I(\square) + F(\square) \cdot I(\square) + \dots + F(\square) \cdot I(\square)$$

Filtrado lineal: correlación

- Qué pasa en los bordes de una imagen?



$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

$$G(i, j) = F(\square) \cdot I(\square) + F(\square) \cdot I(\square) + F(\square) \cdot I(\square) + \dots + F(\square) \cdot I(\square)$$

Filtrado lineal: correlación

- Cuál es el resultado?



Original

0	0	0
0	1	0
0	0	0

?

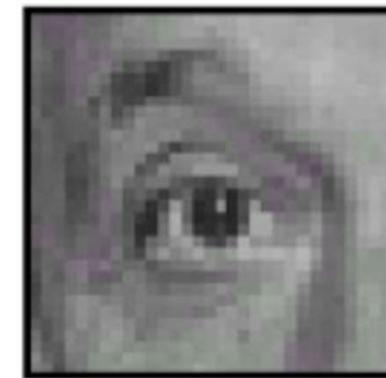
Filtrado lineal: correlación

- Cuál es el resultado?



Original

0	0	0
0	1	0
0	0	0



**Filtered
(no change)**

Filtrado lineal: correlación

- Cuál es el resultado?



0	0	0
0	0	1
0	0	0

?

Filtrado lineal: correlación

- Cuál es el resultado?

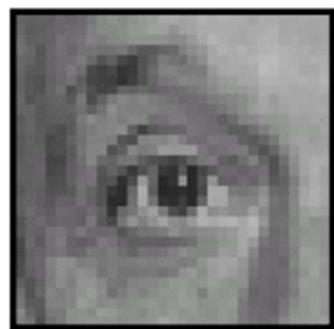


0	0	0
0	0	1
0	0	0



Filtrado lineal: correlación

- Cuál es el resultado?

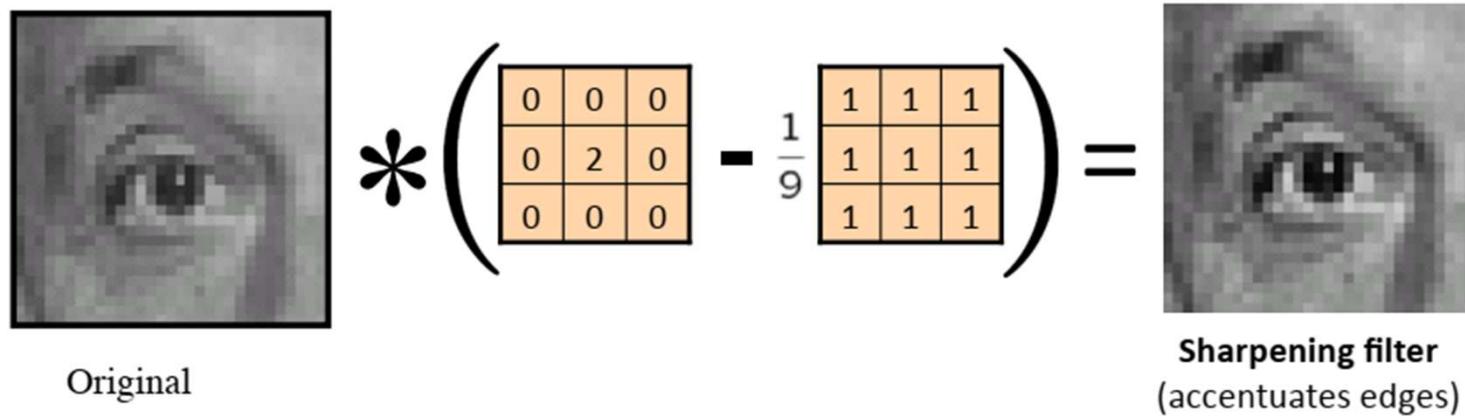


Original

$$\text{Original} * \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{array} - \frac{1}{9} \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right) =$$

Filtrado lineal: correlación

- Cuál es el resultado?

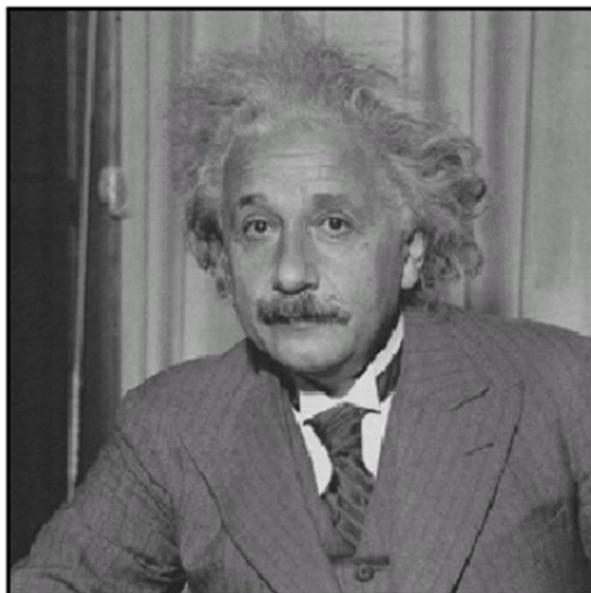


The diagram shows the mathematical operation of linear filtering (correlation) on a grayscale image of an eye. On the left is the **Original** image. In the center is the filtering equation: $\text{Original} * \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{array} - \frac{1}{9} \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right) = \text{Result}$. The result is a **Sharpening filter** output, which accentuates the edges of the eye.

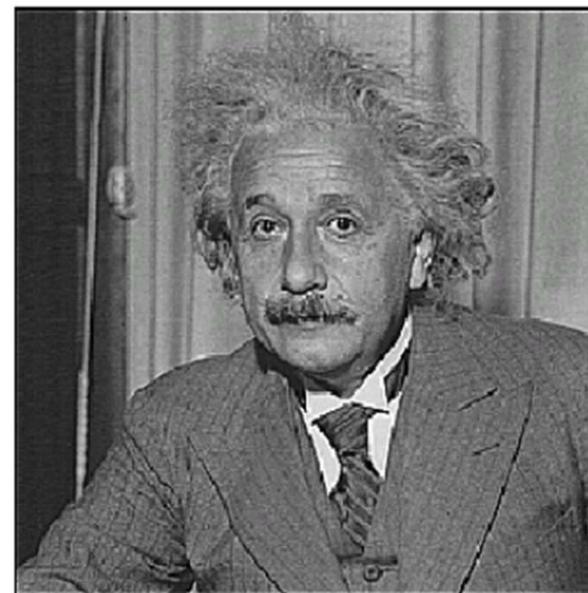
$$\text{Original} * \left(\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{array} - \frac{1}{9} \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array} \right) = \text{Result}$$

Sharpening filter
(accentuates edges)

Sharpening



before



after

Sharpening



Ejemplo de correlación

- Cuál es el resultado de filtrar la señal de impulso (imagen) I con un filtro arbitrario F ?

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

$I(i, j)$



a	b	c
d	e	f
g	h	i

$F(i, j)$

$G(i, j)$

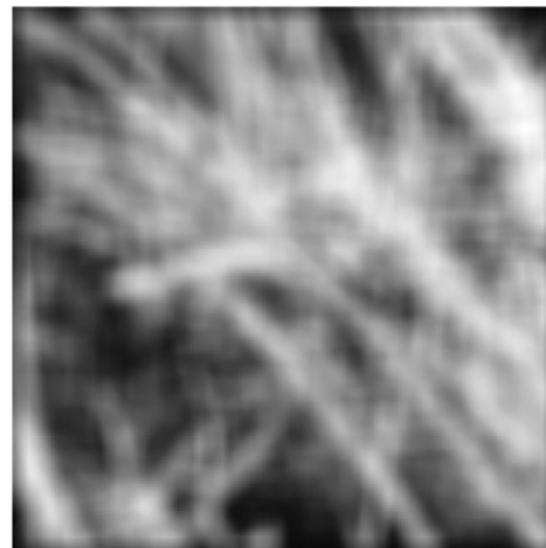
Smoothing by averaging



depicts box filter:
white = high value, black = low value



original



filtered

- Cuál es el efecto del tamaño del filtro?

Filtrado Gaussiano

- Que la influencia dependa de píxels cercanos
- Remueve altas frecuencias de la imagen

0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	90	0	90	90	90	0	0	0
0	0	0	90	90	90	90	90	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

$$I(i, j)$$

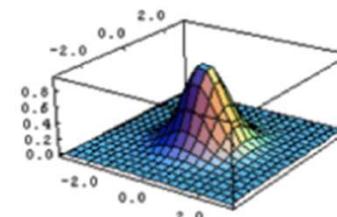
$$\frac{1}{16}$$

1	2	1
2	4	2
1	2	1

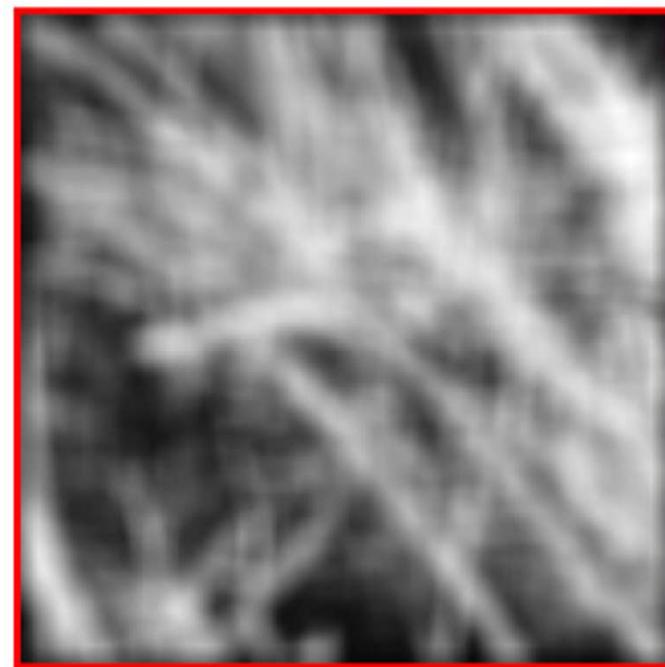
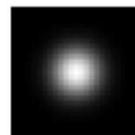
$$F(i, j)$$

This kernel is an approximation of a 2d Gaussian function:

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

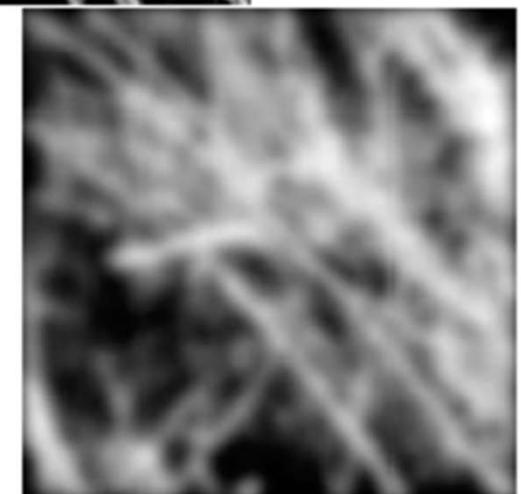


Filtrado Gaussiano



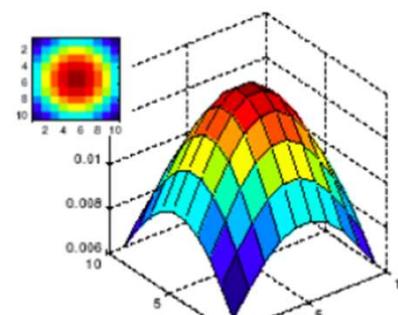
Filtrado Gaussiano

- Promedio vs Gaussiano

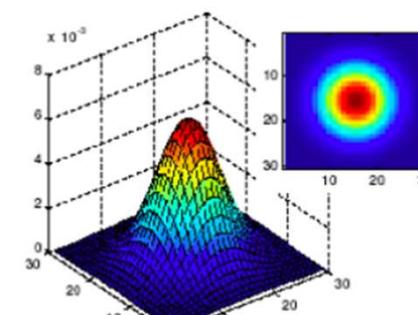


Filtrado Gaussiano

- Tamaño del filtro
 - Función Gaussiana tiene soporte infinito, pero filtros discretos usan kernels finitos



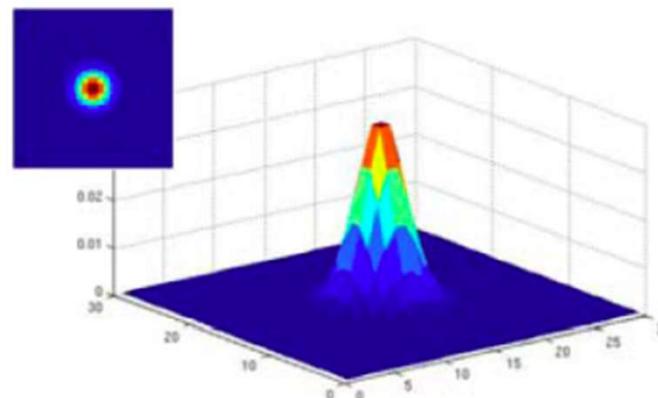
$\sigma = 5$ with
10 x 10
kernel



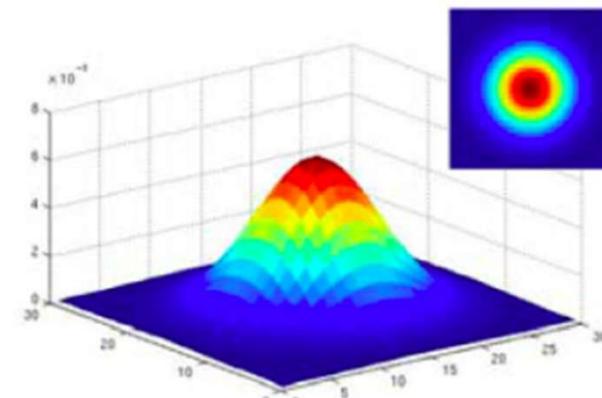
$\sigma = 5$ with
30 x 30
kernel

Filtrado Gaussiano

- Varianza del Gaussiano
 - Determina la cantidad de suavamiento

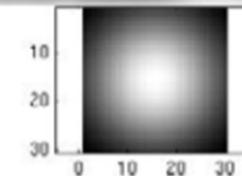
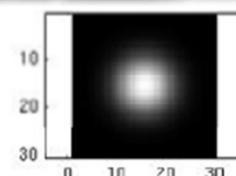
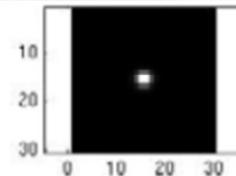


$\sigma = 2$ with
30 x 30
kernel



$\sigma = 5$ with
30 x 30
kernel

Filtrado Gaussiano



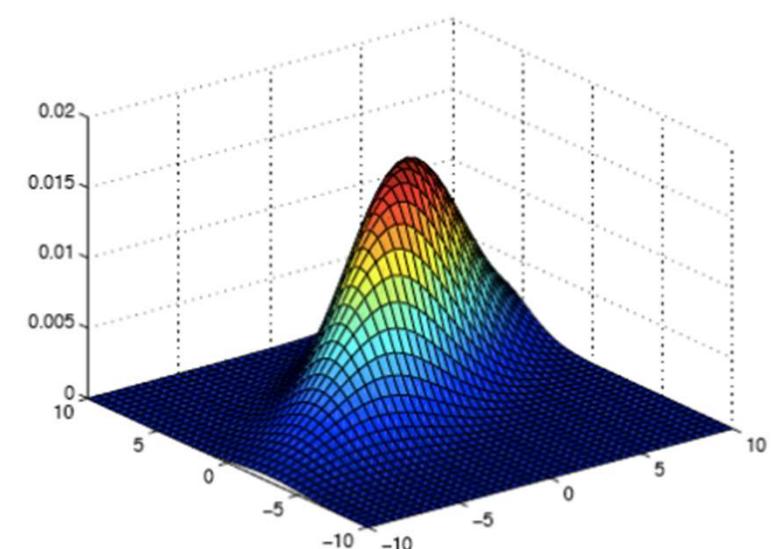
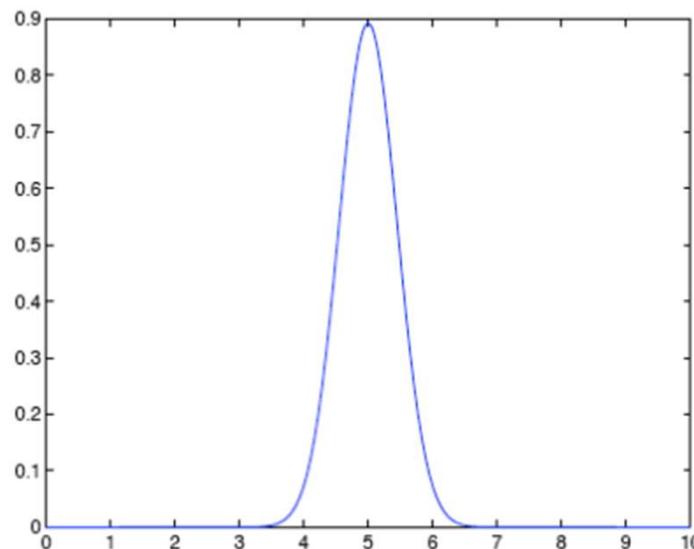
• • •

```
for sigma=1:3:10
    h = fspecial('gaussian', fsize, sigma);
    out = imfilter(im, h);
    imshow(out);
    pause;
end
```

Filtrado Gaussiano

- Se puede generalizar el Gaussiano

$$\mathcal{N}(\mathbf{x}; \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)\right)$$



Propiedades de filtros de suavizado

- Todos los valores son positivos
- Todo el kernel suma uno
- Cantidad de suavizado proporcional al tamaño
- Remueven altas frecuencias. Se conocen como “low-pass” filters

Template matching

- Cómo podemos usar filtering para encontrar a Waldo?



image I



filter F

Template matching

- Recordemos correlación

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

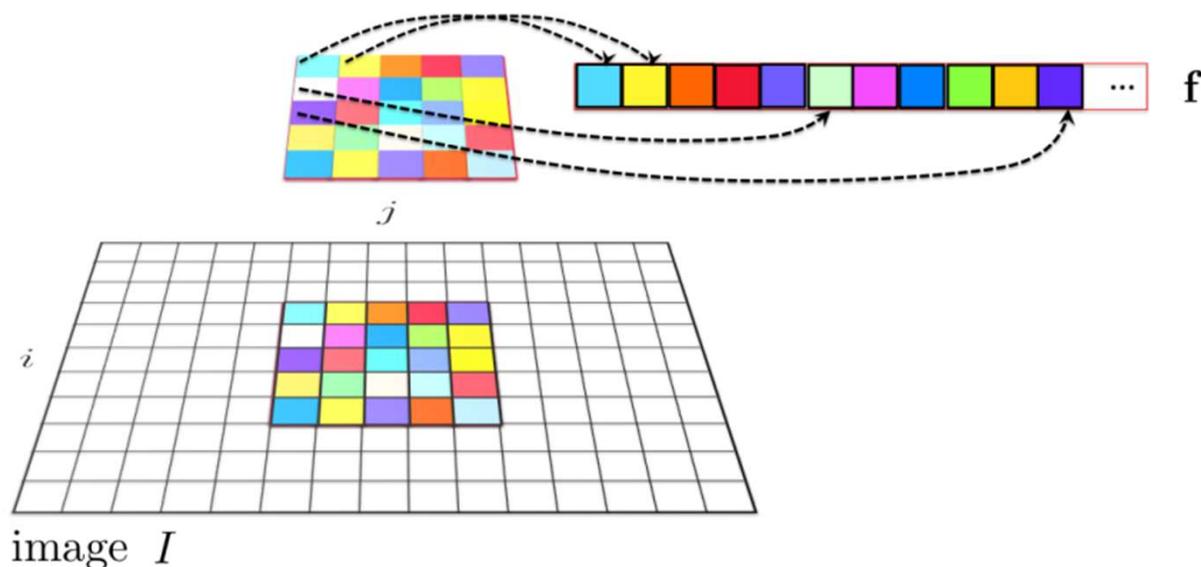
- Veamos como operación de vectores

Template matching

- Recordemos correlación

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

- Veamos como operación de vectores

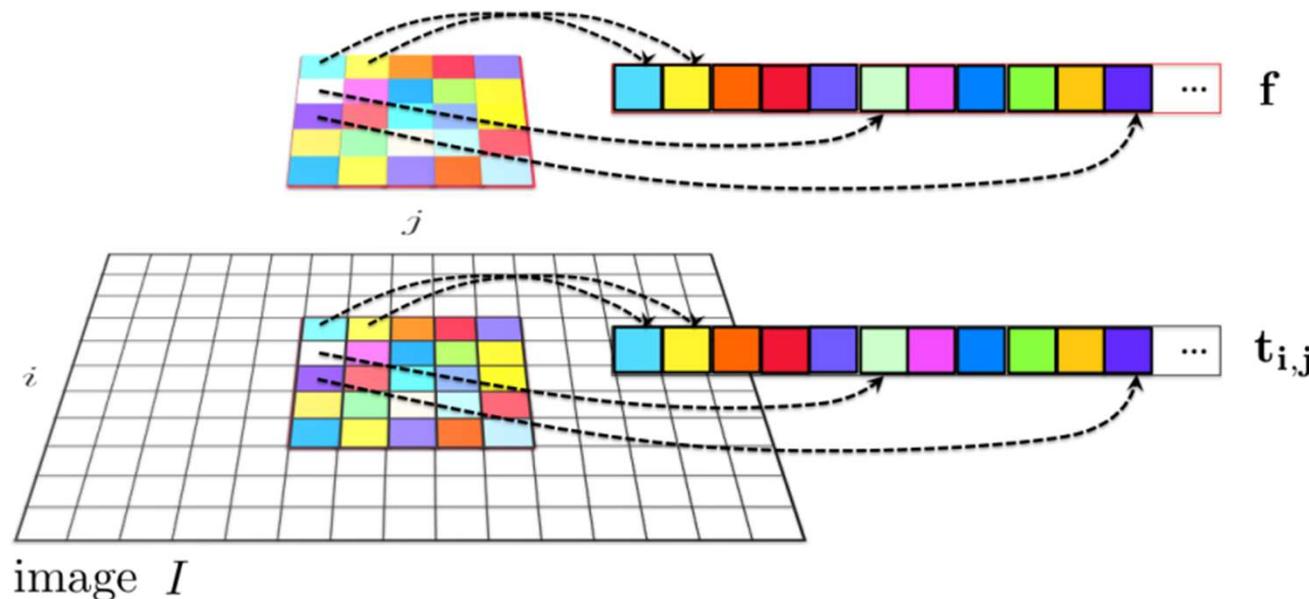


Template matching

- Recordemos correlación

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

- Veamos como operación de vectores

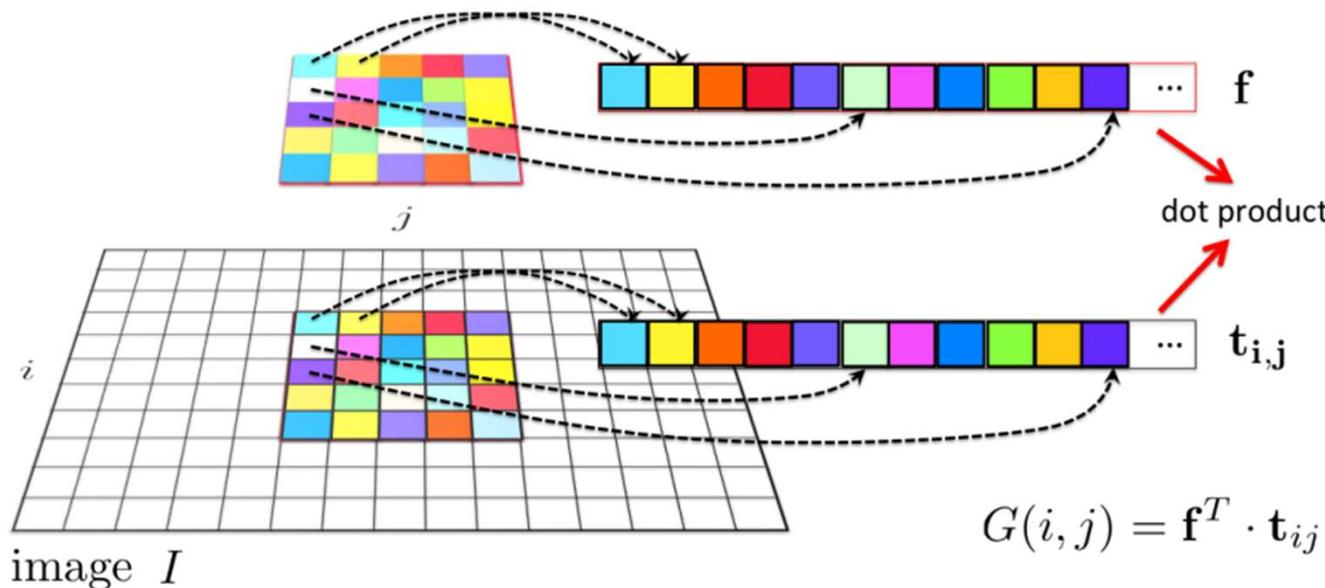


Template matching

- Recordemos correlación

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

- Veamos como operación de vectores



Template matching

- Recordemos correlación

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

- Correlación es el producto punto entre el filtro y la vecindad

$$G(i, j) = \mathbf{f}^T \cdot \mathbf{t}_{ij}$$

Template matching

- Recordemos correlación

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i + u, j + v)$$

- Correlación es el producto punto entre el filtro y la vecindad

$$G(i, j) = \mathbf{f}^T \cdot \mathbf{t}_{ij}$$

- Para encontrar a Waldo, queremos encontrar el mejor score (por ejemplo,
 - 1) para aquella porción de la imagen que se parece al filtro
- Cross-correlation normalizada

$$G(i, j) = \frac{\mathbf{f}^T \cdot \mathbf{t}_{ij}}{\|\mathbf{f}\| \cdot \|\mathbf{t}_{ij}\|}$$

Template matching



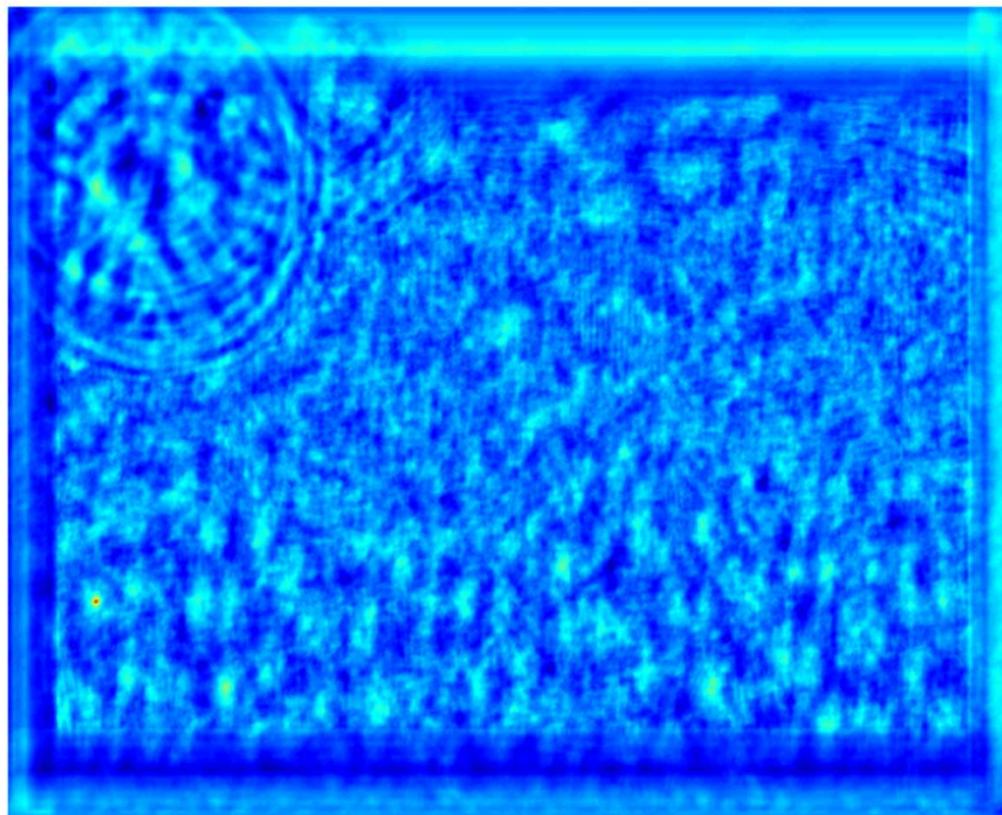
image I



filter F

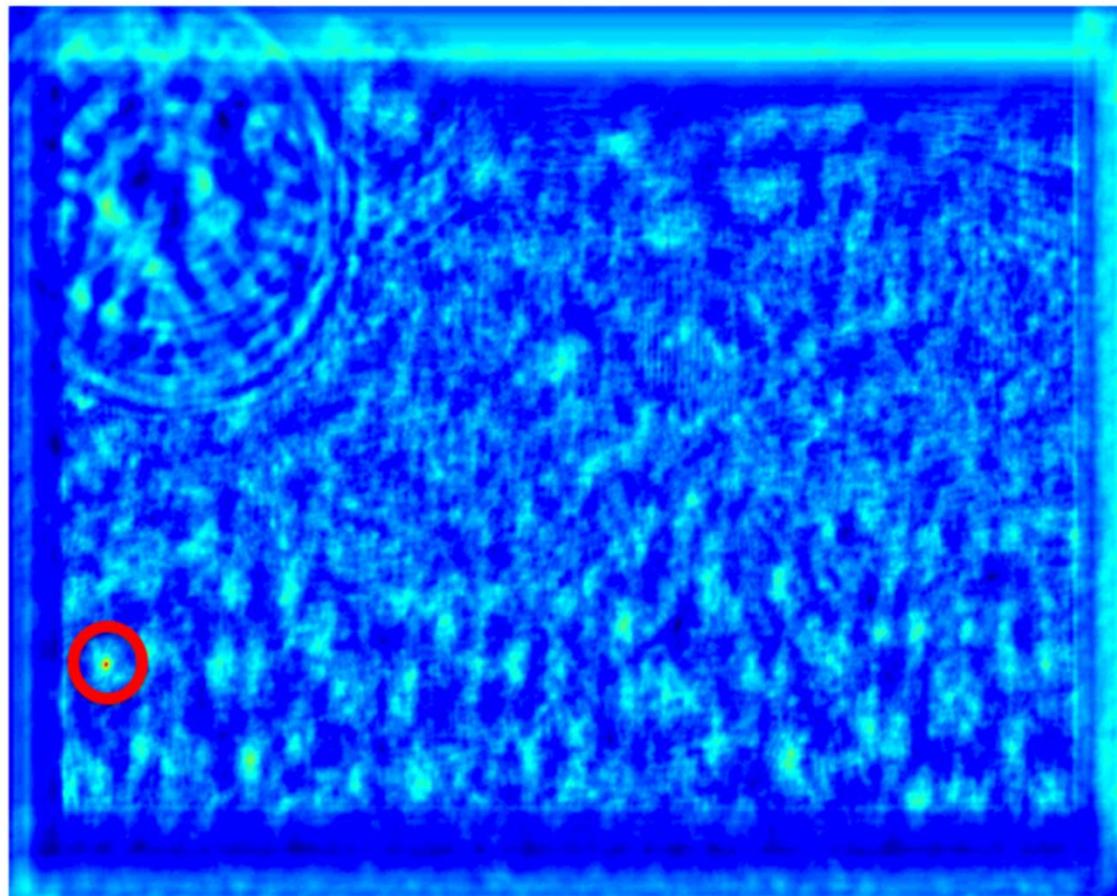
Template matching

Resultado de cross-correlation



Template matching

Encontramos el valor más alto



Template matching

Voilá!

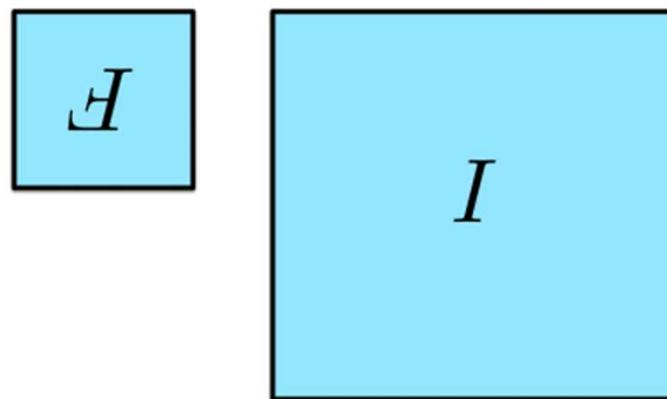


Convolución

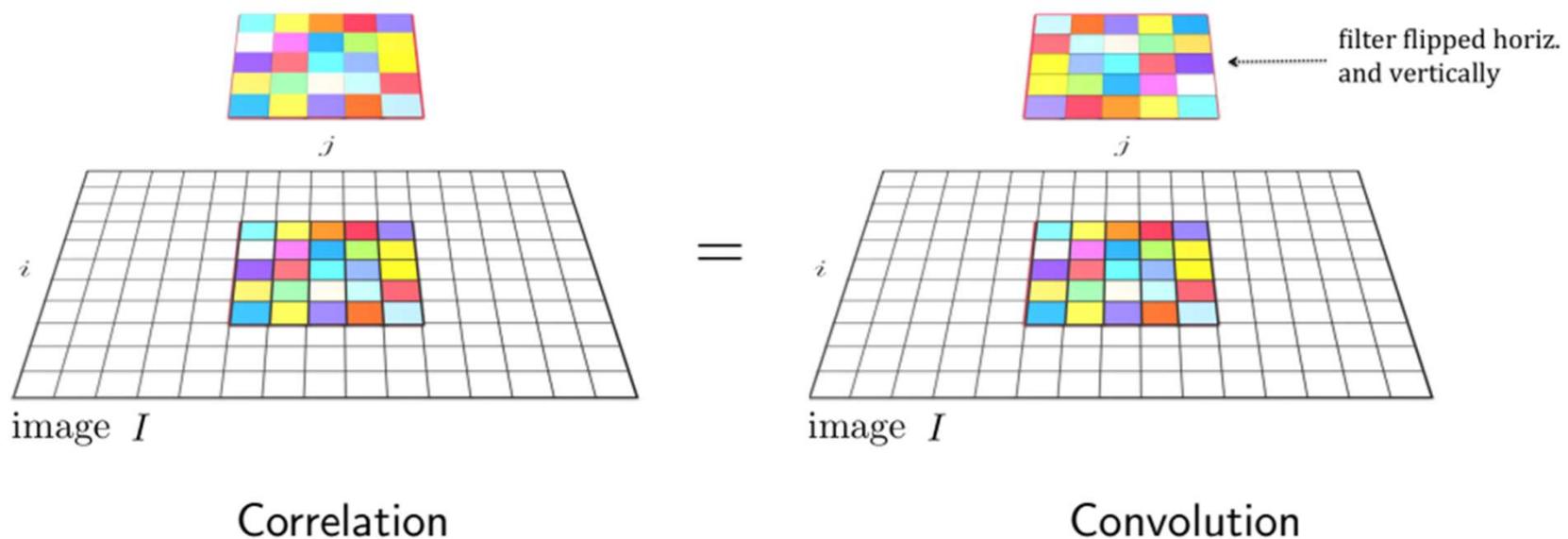
- Operador de convolución

$$G(i, j) = \sum_{u=-k}^k \sum_{v=-k}^k F(u, v) \cdot I(i - u, j - v)$$

- Equivalente a voltear el filtro en ambas dimensiones y aplicar correlación



Convolución vs Correlación



Convolución vs Correlación

- Para un filtro Gaussiano, hay diferencia?

Convolución vs Correlación

- Para un filtro Gaussiano, hay diferencia?
- Para el siguiente filtro, hay diferencia?

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

Convolución vs Correlación

- Para un filtro Gaussiano, hay diferencia?
- Para el siguiente filtro, hay diferencia?

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$$

- Si el input es una señal de impulso, cuál es la diferencia?

Convolución vs Correlación

- Propiedades de convolución

Commutative : $f * g = g * f$

Associative : $f * (g * h) = (f * g) * h$

Distributive : $f * (g + h) = f * g + f * h$

Assoc. with scalar multiplier : $\lambda \cdot (f * g) = (\lambda \cdot f) * g$

- La transformada de Fourier de dos imágenes convolucionadas es el producto de sus transformadas individuales

$$\mathcal{F}(f * g) = \mathcal{F}(f) \cdot \mathcal{F}(g)$$

Convolución vs Correlación

- Propiedades de convolución

$$\text{Commutative} : f * g = g * f$$

$$\text{Associative} : f * (g * h) = (f * g) * h$$

$$\text{Distributive} : f * (g + h) = f * g + f * h$$

$$\text{Assoc. with scalar multiplier} : \lambda \cdot (f * g) = (\lambda \cdot f) * g$$

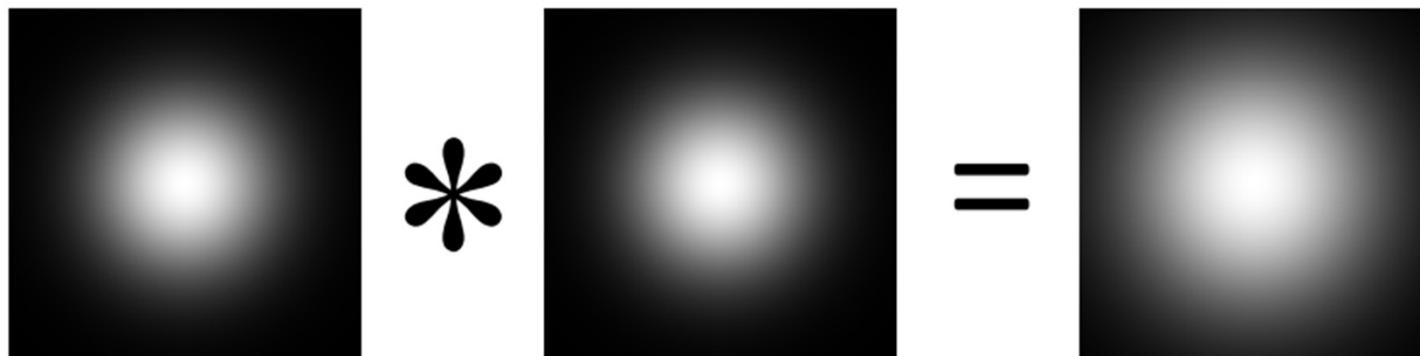
- La transformada de Fourier de dos imágenes convolucionadas es el producto de sus transformadas individuales

$$\mathcal{F}(f * g) = \mathcal{F}(f) \cdot \mathcal{F}(g)$$

- Correlación y convolución son **linear shift-invariant**.
 - El efecto del operador es el mismo en todas partes

Filtro Gaussiano

- Convolucionar dos veces con un kernel Gaussiano de ancho σ es lo mismo que convolucionar una vez con un kernel de ancho $\sigma\sqrt{2}$



- No es necesario filtrar dos veces, con un filtro más grande basta

Filtro Separables: Truco

- El proceso de realizar una convolución requiere K^2 operaciones por pixel, donde K es el tamaño del filtro de convolución
- Se puede hacer más rápido?

Filtro Separables: Truco

- El proceso de realizar una convolución requiere K^2 operaciones por pixel, donde K es el tamaño del filtro de convolución
- Se puede hacer más rápido?
- En algunos casos, la convolución se puede separar en dos convoluciones 1D, tomando $2K$ operaciones

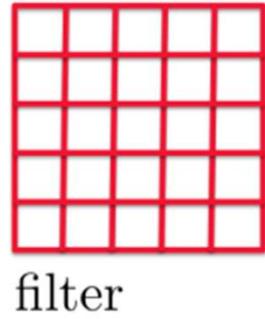
Filtro Separables: Truco

- El proceso de realizar una convolución requiere K^2 operaciones por pixel, donde K es el tamaño del filtro de convolución
- Se puede hacer más rápido?
- En algunos casos, la convolución se puede separar en dos convoluciones 1D, tomando $2K$ operaciones
- Si esto es posible, el filtro se llama **separable**

$$\mathbf{F} = \mathbf{v} \mathbf{h}^T$$

Filtro Separables: Truco

- Cómo funciona?



filter

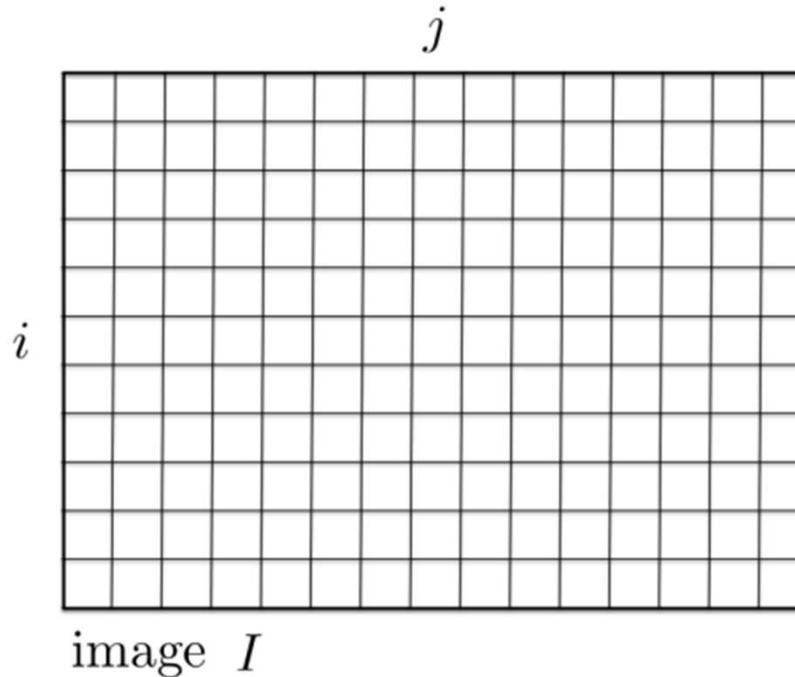
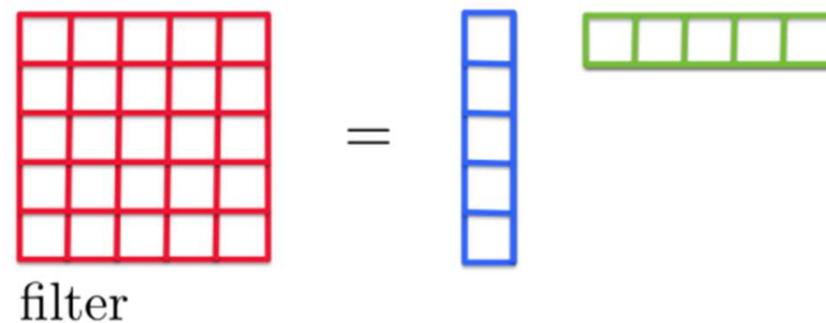


image I

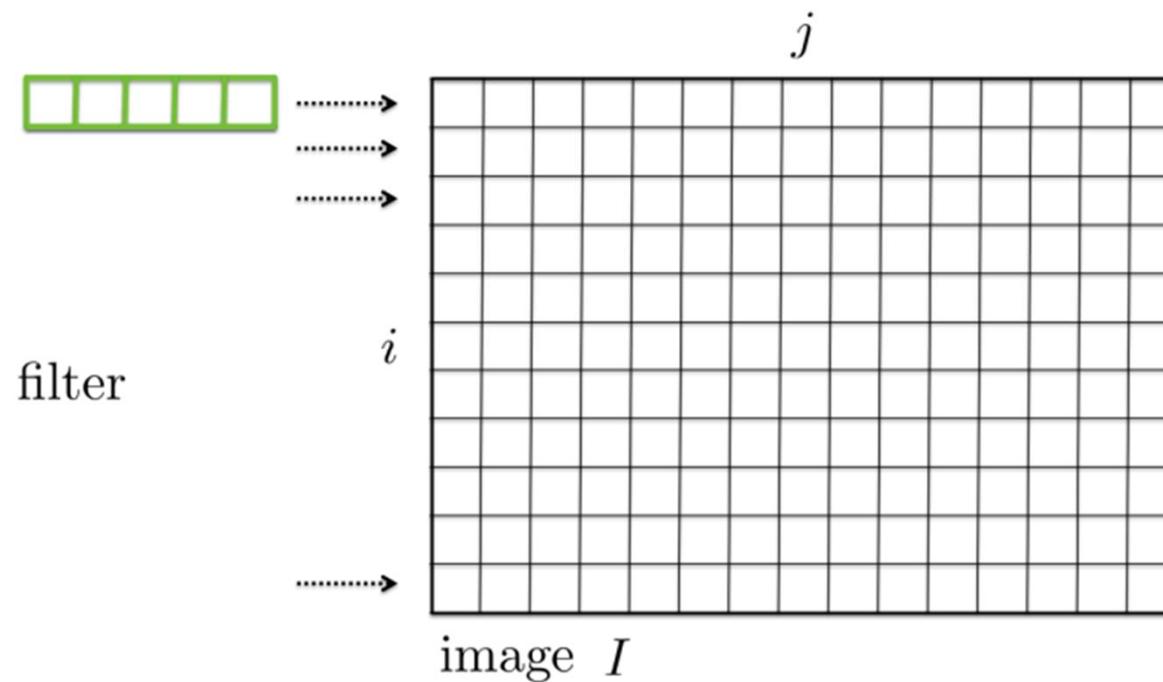
Filtro Separables: Truco

- Cómo funciona?



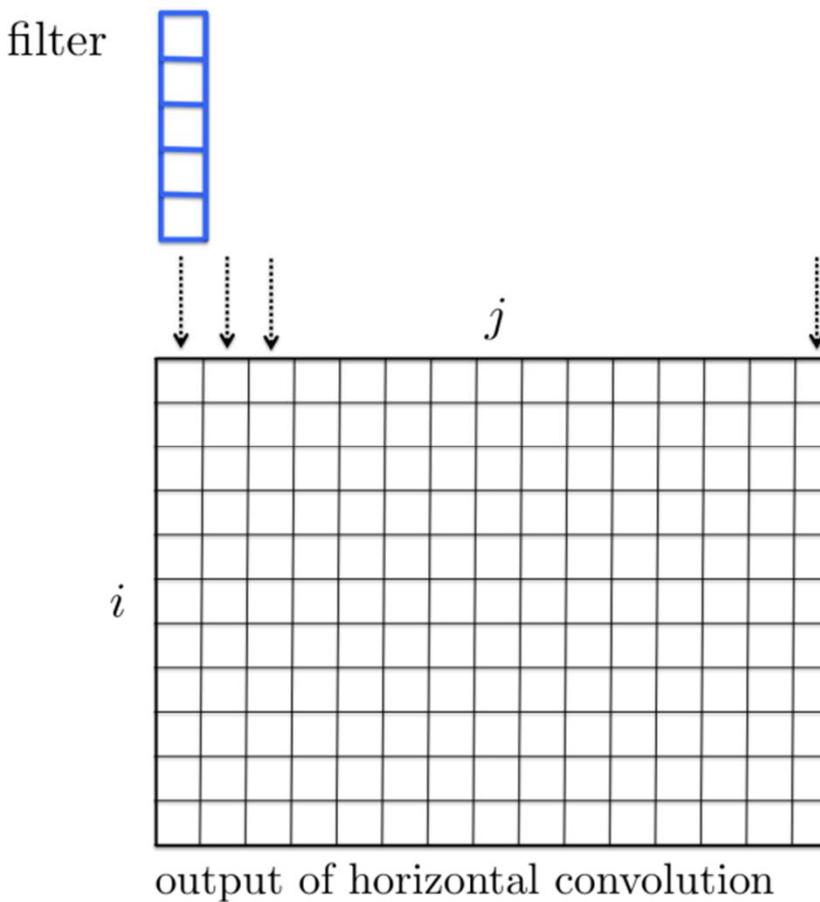
Filtro Separables: Truco

- Cómo funciona?



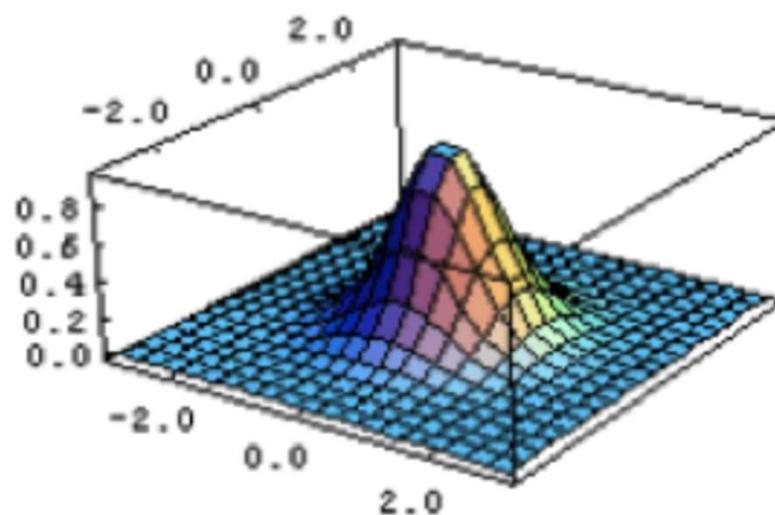
Filtro Separables: Truco

- Cómo funciona?



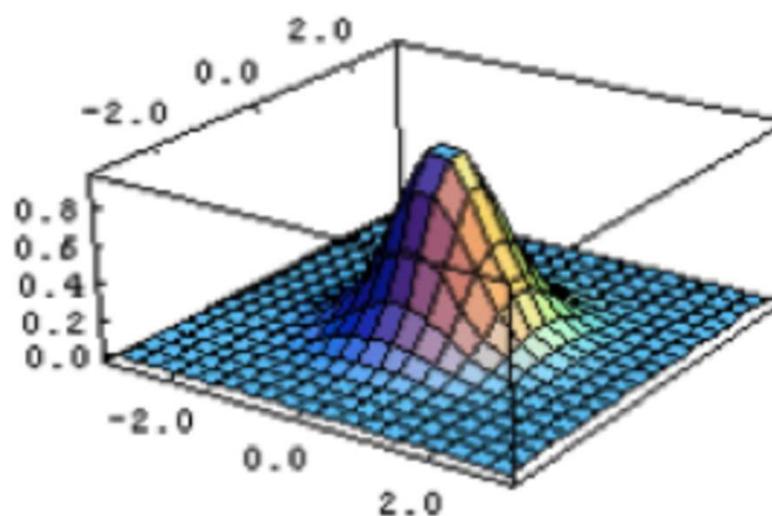
Filtro Separables: Filtros Gaussiano

Gaussian : $f(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{\sigma^2}}$



Filtro Separables: Filtros Gaussiano

$$\begin{aligned}\text{Gaussian} : f(x, y) &= \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{\sigma^2}} \\ &= \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{\sigma^2}} \right) \cdot \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{y^2}{\sigma^2}} \right)\end{aligned}$$



Filtro Separables:

- Este filtro es separable?
- Si la respuesta es si, cuál sería su versión separable?

$$\frac{1}{K^2} \begin{array}{|c|c|c|c|} \hline 1 & 1 & \cdots & 1 \\ \hline 1 & 1 & \cdots & 1 \\ \hline \vdots & \vdots & & \vdots \\ \hline 1 & 1 & \cdots & 1 \\ \hline \end{array}$$

Filtro Separables:

- Este filtro es separable?
- Si la respuesta es si, cuál sería su versión separable?

$$\frac{1}{K^2} \begin{array}{|c|c|c|c|} \hline 1 & 1 & \cdots & 1 \\ \hline 1 & 1 & \cdots & 1 \\ \hline \vdots & \vdots & 1 & \vdots \\ \hline 1 & 1 & \cdots & 1 \\ \hline \end{array} \quad \frac{1}{K} \begin{array}{|c|c|c|c|} \hline 1 & 1 & \cdots & 1 \\ \hline \end{array}$$

Filtro Separables:

- Este filtro es separable?
- Si la respuesta es si, cuál sería su versión separable?

$$\frac{1}{16} \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Filtro Separables:

- Este filtro es separable?
- Si la respuesta es si, cuál sería su versión separable?

$$\frac{1}{16} \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

$$\frac{1}{4} \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline \end{array}$$

Filtro Separables:

- Este filtro es separable?
- Si la respuesta es si, cuál sería su versión separable?

$$\frac{1}{8} \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$

Filtro Separables:

- Este filtro es separable?
- Si la respuesta es si, cuál sería su versión separable?

$$\frac{1}{8} \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array}$$
$$\frac{1}{2} \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline \end{array}$$

Filtro Separables:

- Cómo podemos saber si un filtro es separable?
- Mirando su forma analítica

Filtro Separables:

- Cómo podemos saber si un filtro es separable?
- Mirando su forma analítica
- Aplicamos singular value decomposition (SVD) y si algún valor singular es diferente de cero, el filtro es separable

$$F = \mathbf{U}\Sigma\mathbf{V}^T = \sum_i \sigma_i u_i v_i^T$$

- Filtros son $\sqrt{\sigma_1} \mathbf{u}_1 \quad \sqrt{\sigma_1} \mathbf{v}_1^T$