

## Introducción al procesamiento digital de imágenes

Podemos definir una imagen como una función bidimensional  $f(x_1, x_2)$  donde  $x = (x_1, x_2)$  son las **coordenadas espaciales**, y el valor de  $f$  en cualquier  $x$  es la **intensidad** de la imagen en dicho punto (ver figura 1)

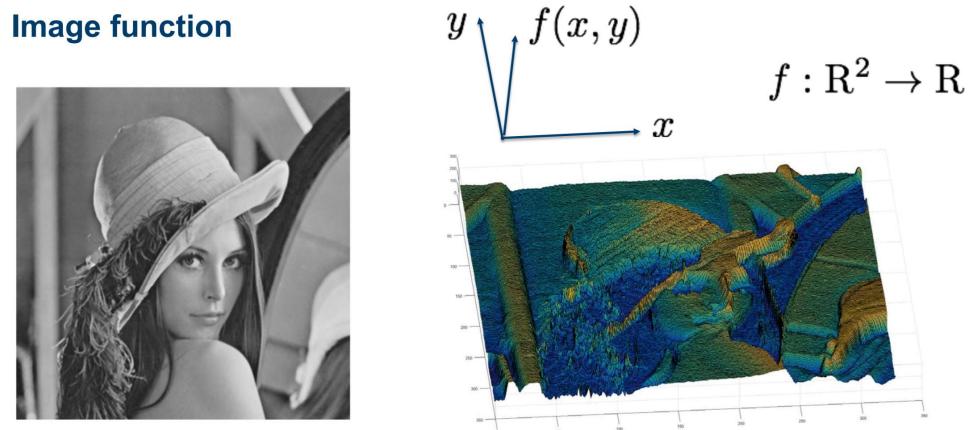


Figura 1: Ejemplo de imagen y función imagen. La mujer de la foto se llama Lena, su imagen se usa desde 1973 como prueba para algoritmos de compresión y diversas técnicas de procesado de imágenes. Hoy en día se ha convertido en un estándar industrial y científico.

Desde este punto de vista, una imagen puede considerarse como una función continua definida sobre un conjunto continuo (imagen analógica) o como una función discreta definida sobre un dominio discreto (imagen digital). Ambos puntos de vista resultan útiles en el procesamiento de imágenes.

Convertir una imagen analógica a digital requiere que tanto las coordenadas como la intensidad sean digitalizadas. Digitalizar las coordenadas se llama **muestrear**, mientras que digitalizar la intensidad se denomina **cuantizar**. Entonces, cuando todas las cantidades son discretas, llamamos a la imagen una **imagen digital** (figura 2). El camino opuesto, de digital a analógico, es también posible y se denomina **interpolación**.

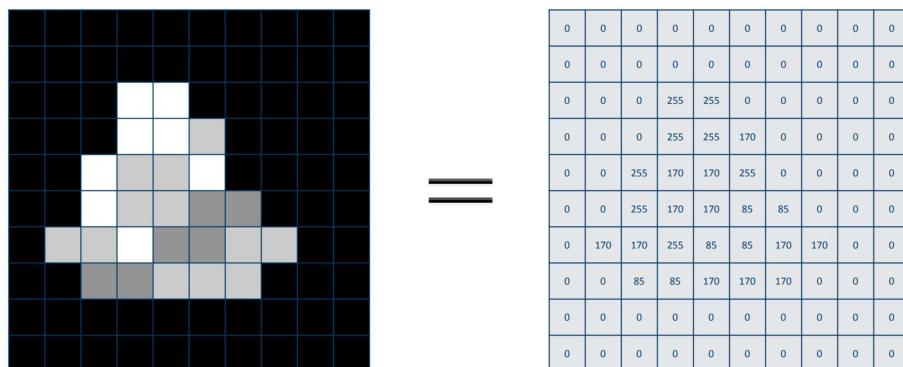


Figura 2: imagen digital. Es la versión discreta (muestreada y cuantizada) de una función imagen continua  $f(x_1, x_2)$

El tipo de dato habitual para una imagen es uint8, es decir, un entero sin signo representado en 8 bits. Esto nos da  $2^8=256$  valores que se distribuyen en el rango de [0, 255] para cada pixel.

**Tipos de imágenes y conversiones.** Existen tres tipos principales de imágenes:

- **Imagen de intensidad** es una matriz de datos cuyos valores han sido escalados para que representen intensidades de una escala de grises. Cuando los elementos de una imagen de intensidad son de clase uint8 (enteros almacenados en 8 bits) o de clase uint16 (enteros almacenados en 16 bits), pueden almacenar, respectivamente,  $2^8=256$  valores en el rango [0, 255] o  $2^{16}=65536$  valores en el rango [0, 65535]. Si la imagen es de clase *double*, los valores son números en punto flotante (que se almacenan en 32 bits). En este último caso, los valores suelen tomarse en el rango [0, 1], por convención.
- **La imagen binaria** es una imagen en blanco y negro. Cada pixel tiene asignado un valor lógico de 0 ó 1.
- **La imagen en color** es como la imagen de intensidad, pero tiene tres canales, es decir, a cada pixel le corresponden tres valores de intensidad (RGB) en lugar de uno.

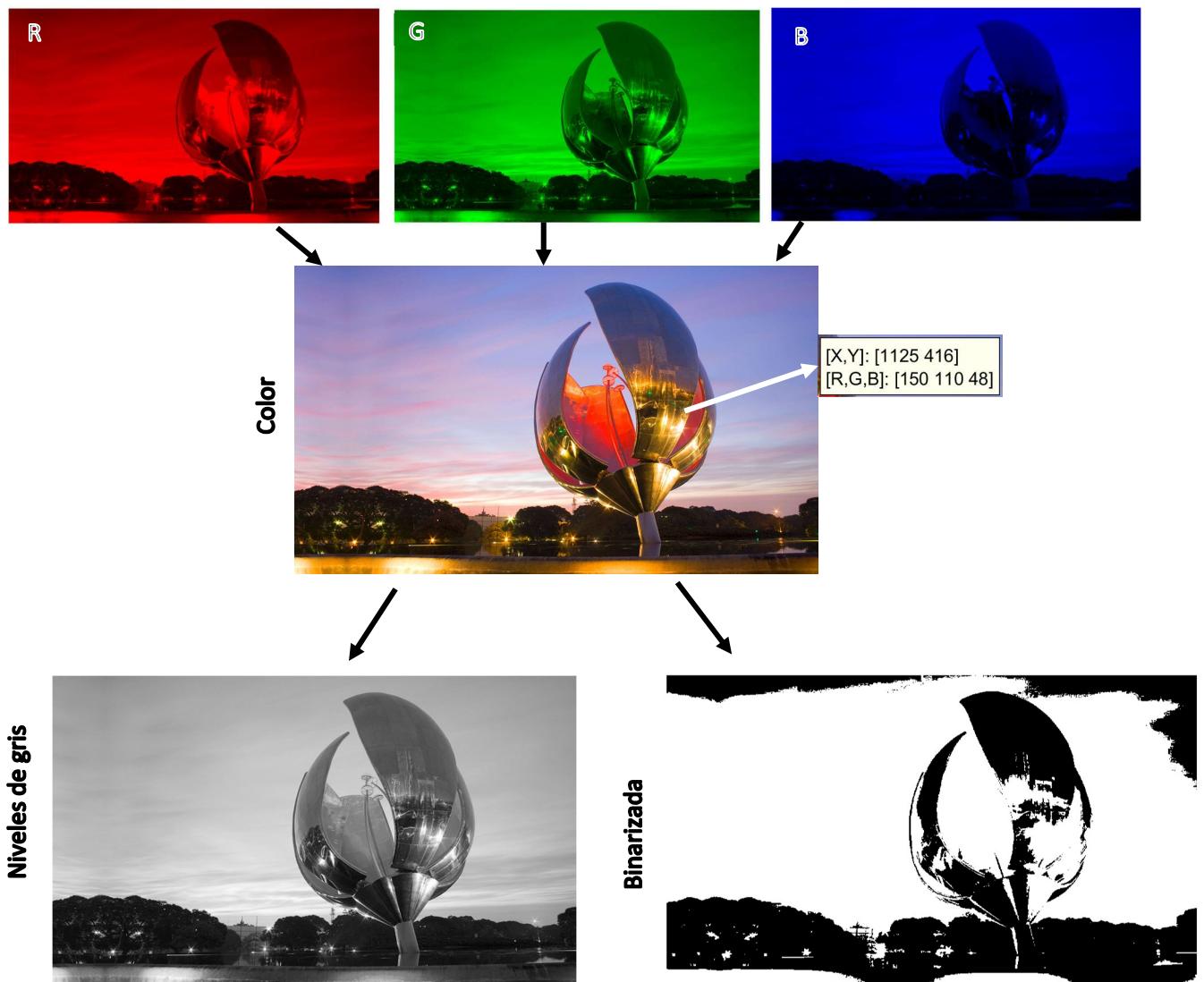


Figura 3: Composición de imagen RGB y niveles de gris

Cuando realizamos transformaciones matemáticas de imágenes, a menudo necesitamos que la imagen sea de tipo *double*. Pero cuando la leemos y almacenamos ahorraremos espacio usando codificación entera.

## Procesamiento digital de imágenes

El **procesamiento de imágenes digitales** es el conjunto de técnicas u operaciones que se aplican a las imágenes digitales con el objetivo de mejorar la calidad, facilitar la búsqueda de información o mejorar ciertas características de esta.

Estas operaciones también conocidas como filtros, puede ser llevada a cabo en el dominio de frecuencias o espacio. Los principales objetivos que se persiguen con la aplicación de filtros son:

- Suavizar la imagen: reducir la cantidad de variaciones de intensidad entre píxeles vecinos.
- Eliminar ruido: eliminar aquellos píxeles cuyo nivel de intensidad es muy diferente al de sus vecinos y cuyo origen puede estar tanto en el proceso de adquisición de la imagen como en el de transmisión.
- Realizar bordes: destacar los bordes que se localizan en una imagen.
- Detectar bordes: detectar los píxeles donde se produce un cambio brusco en la función intensidad.
- Reconocer formas

Hoy en día se ha desarrollado una enorme batería de operadores, aquí se muestran algunos a modo de ejemplo

a) **Operadores puntuales.** Son transformaciones pixel a pixel

$$g[i, j] = h(f[i, j])$$

En esta clasificación se pueden mencionar: ajuste de brillo, ajuste de contraste, transformaciones de color, ecualización de histogramas.

b) **Filtrado en el dominio espacial:** Es una operación matemática que permite modificar el valor de los pixels de una imagen empleando una función de un entorno local de cada pixel. En la figura 4, se observa esquemáticamente como aplicar un filtro (**kernel**) en la posición marcada con el cuadrado rojo, en el entorno del pixel  $ij$  de la imagen de la izquierda (**imagen local**) da como resultado un valor modificado en el lugar  $ij$  de la imagen de la derecha. Como ejemplos de filtros se pueden mencionar:

- Suavizado o resaltado
- Extracción de detalles (medición de textura, hallar bordes, patrones o valores distintivos)

Los filtros se aplican empleando la correlación cruzada en 2D. Por completitud la escribimos aquí, pero ya tienen experiencia en el uso de la correlación cruzada en 1D. Si  $f(i, j)$  es la función imagen de  $N \times M$  y  $h(i, j)$  es un kernel de  $(2k + 1) \times (2k + 1)$  la correlación es

$$g[i, j] = h \otimes f = \sum_{u=-k}^{+k} \sum_{v=-k}^{+k} h[u, v]f[i + u, j + v]$$

La función de Python que hace esta operación es `scipy.signal.correlate2d(args)` y en Matlab `xcorr(a,b)`

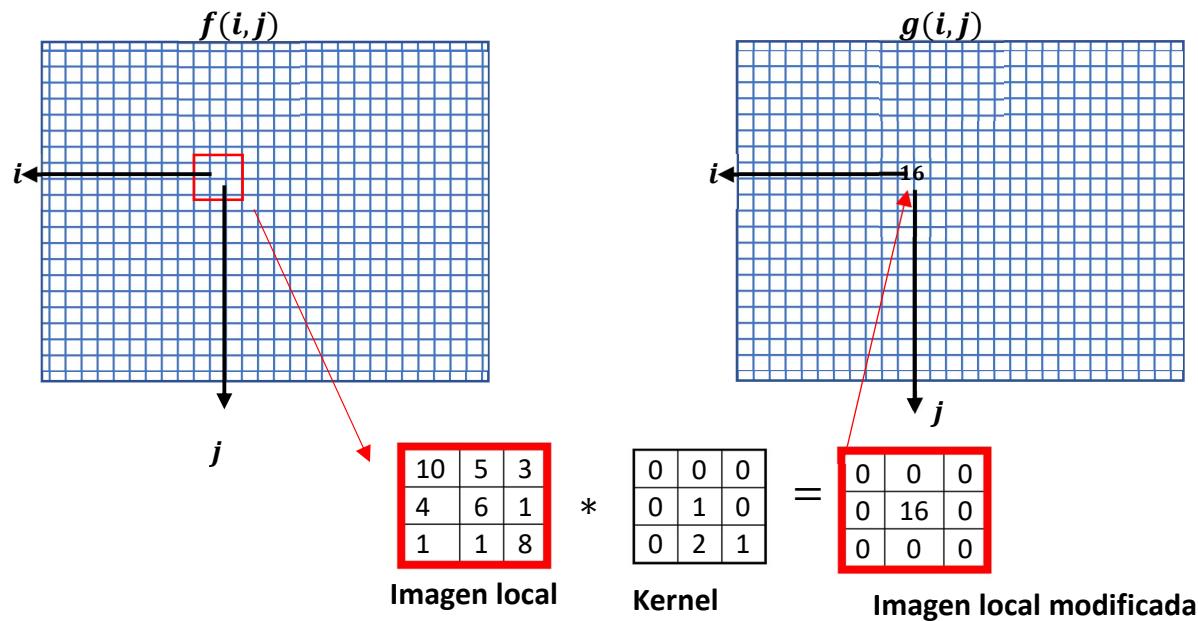
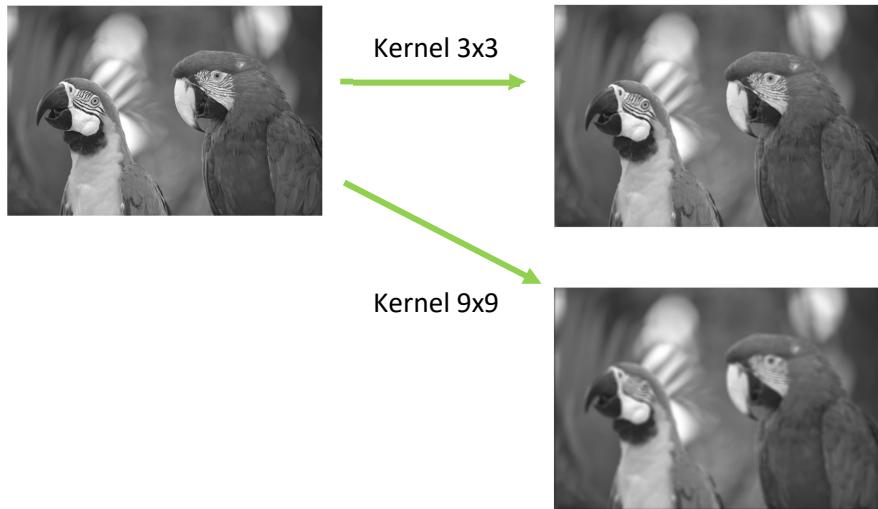


Figura 4: ejemplo filtrado espacial. En el ejemplo se ve el filtro de la media aplicado en la posición  $i,j$

Empleando la correlación cruzada se definen, por ejemplo, los siguientes filtros

b.1 Filtro de la media móvil (suavizado): Reemplaza cada pixel con un promedio de sus vecinos

## Ejemplo de aplicación



**b.2) Filtro para incrementar la nitidez: Resalta el valor de un pixel respecto a la media**



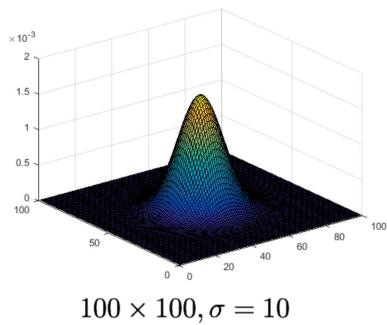
$$\begin{matrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{matrix} - \frac{1}{9} \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

**b.3) Filtro gaussiano (suaviza dando más peso a los valores centrales)**

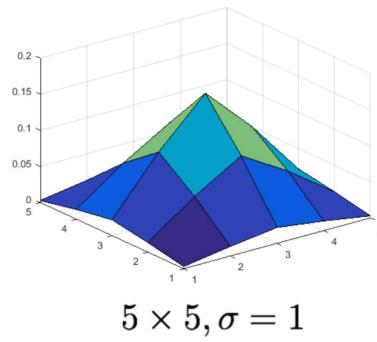
La función Gaussiana es

$$G_\sigma(i, j) = \frac{1}{2\pi\sigma^2} e^{-\frac{i^2+j^2}{2\sigma^2}}$$

Dos ejemplos con distinto ancho y numero de pixels:

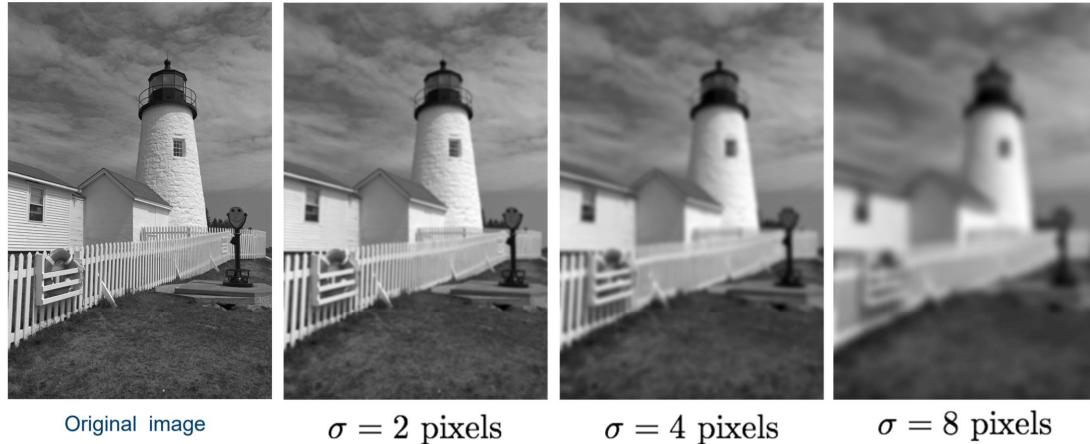


$100 \times 100, \sigma = 10$



$5 \times 5, \sigma = 1$

Ejemplo aplicado a una imagen



#### b.4) Detección de bordes

Se definen a partir de la aproximación discreta de las derivadas. Por ejemplo, para la derivada de primer orden se tiene

$$\frac{\partial f}{\partial x}[i, j] \approx f[i + 1, j] - f[i, j]$$

$$\frac{\partial f}{\partial y}[i, j] \approx f[i, j + 1] - f[i, j]$$

Y se aplican como gradiente vectorial o módulo

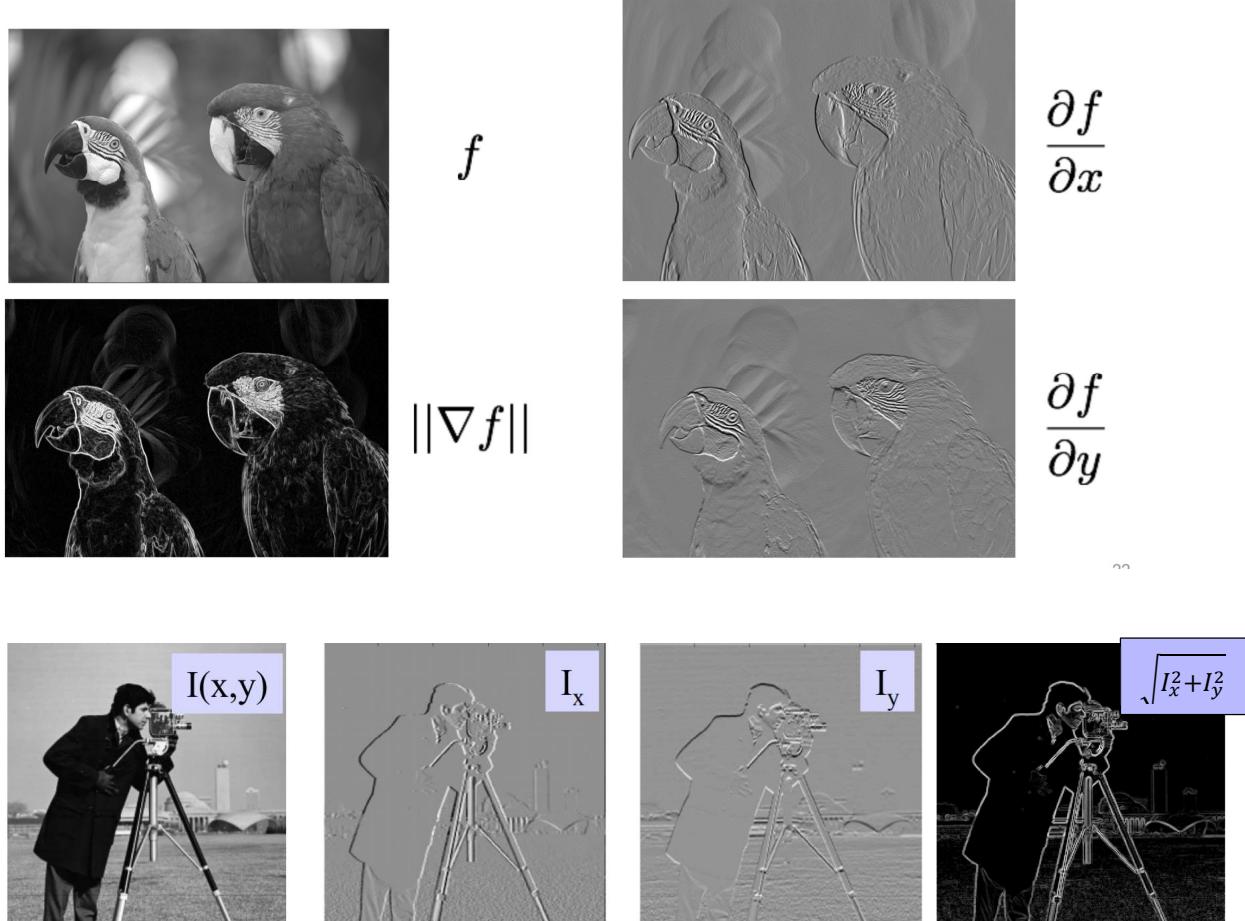
Gradiente  $\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$ , el gradiente es perpendicular al borde, para hallar su magnitud y ángulo puedo usar

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \text{ y } \theta = \text{atan2}\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

Los operadores derivadores en x e y más conocidos son los de Prewitt

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

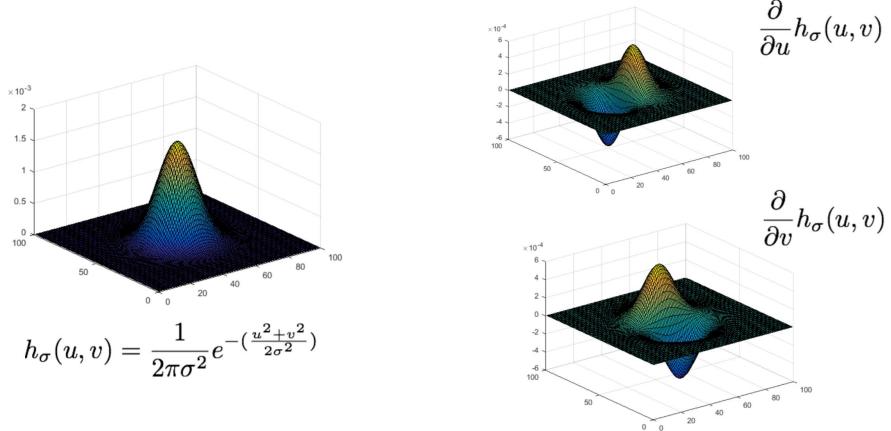
Ejemplos



Observar como el operador en x resalta más los bordes horizontales y el operador en y resalta más los verticales.

### b.5) Filtro derivativo Gaussiano

Se define a partir de la derivada de la gaussiana.



Dentro de esta categoría, el filtro de SOBEL es una aproximación al gaussiano derivativo

-1	0	1
-2	0	2
-1	0	1

x-direction

-1	-2	-1
0	0	0
1	2	1

y-direction

## Ejemplo



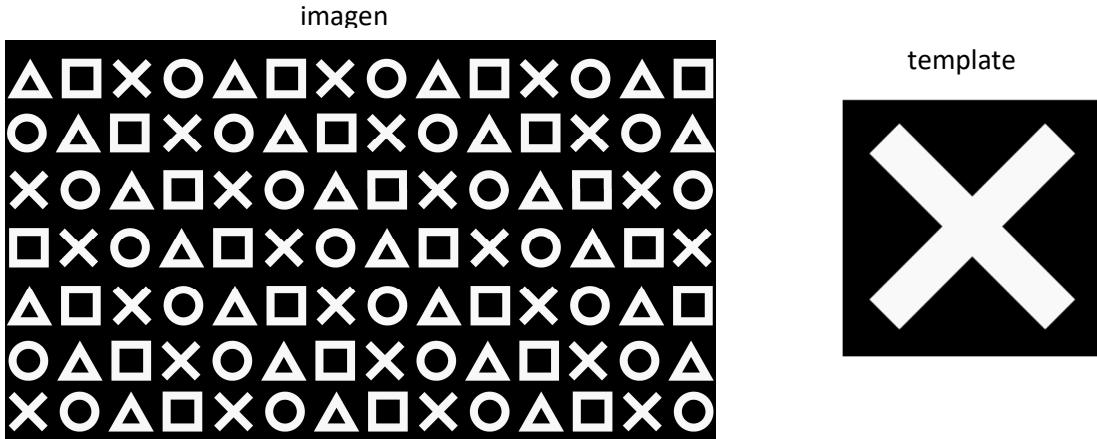
c) **Filtrado de imagen en el dominio de frecuencias:** Se modifican las frecuencias espaciales de la imagen para eliminar ruido, re-muestrear, comprimir la imagen

Este lo vamos a dejar para más adelante

## Template Matching

Es la técnica que se emplea para encontrar patrones en imágenes. Se puede considerar como un filtrado espacial porque está definido como una correlación en el espacio de posiciones.

Miremos de que se trata con un ejemplo (la explicación ya la vimos en la clase grabada para el caso 1D): Queremos encontrar determinados patrones en la siguiente imagen. Por ejemplo, una cruz. (previamente ajuste el brillo y contraste en esta imagen)



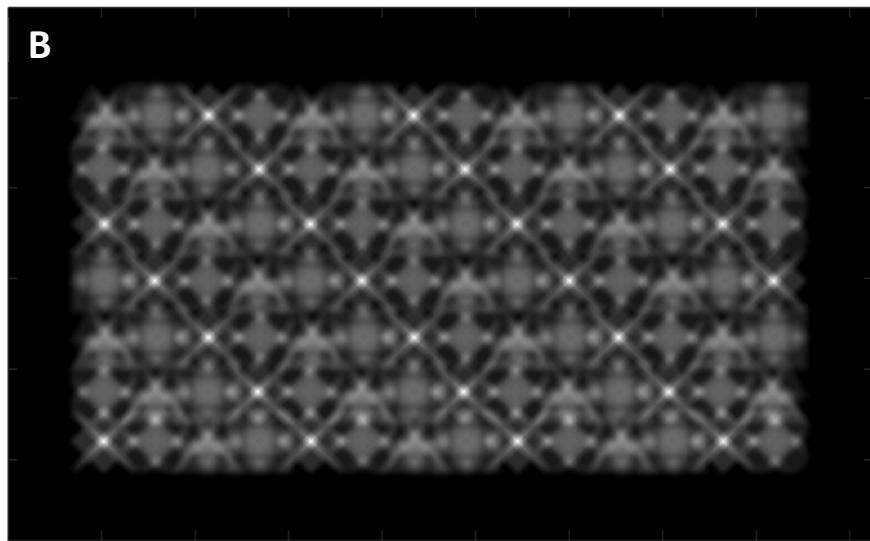
Vamos a calcular entonces la correlación entre la imagen y el kernel (template), restándole el valor medio de la imagen previamente, es decir:

$$C[i, j] = (F - \bar{F}) \otimes (K - \bar{K}) = \sum_{u=-k}^{+k} \sum_{v=-k}^{+k} (K[u, v] - \bar{K})(F[i + u, j + v] - \bar{F})$$

Podríamos además normalizar cada cantidad, en el caso que la cantidad de luz (suma todos los datos de la imagen) sea muy distinta en kernel y la imagen. Por ejemplo

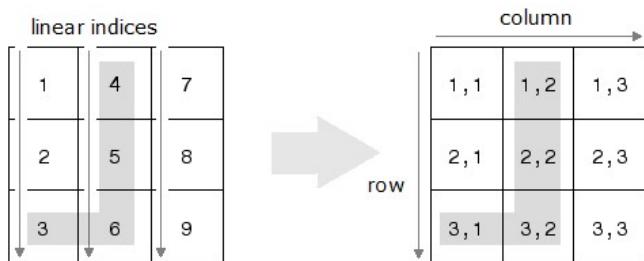
$$\hat{F} = \frac{F - \bar{F}}{\sqrt{\sum (F_{ij} - \bar{F})^2}}$$

Además, como queremos encontrar detalles que tal vez están muy cerca del borde, y vamos a trabajar con una correlación, nos conviene completar con ceros alrededor de la imagen con un tamaño igual al del kernel. La función que completa con ceros se llama padarray o padding según el programa. La imagen resulta como la imagen A en la siguiente figura. La correlación cruzada de la imagen y el kernel (restando a la imagen y kernel el valor medio) es la imagen B.



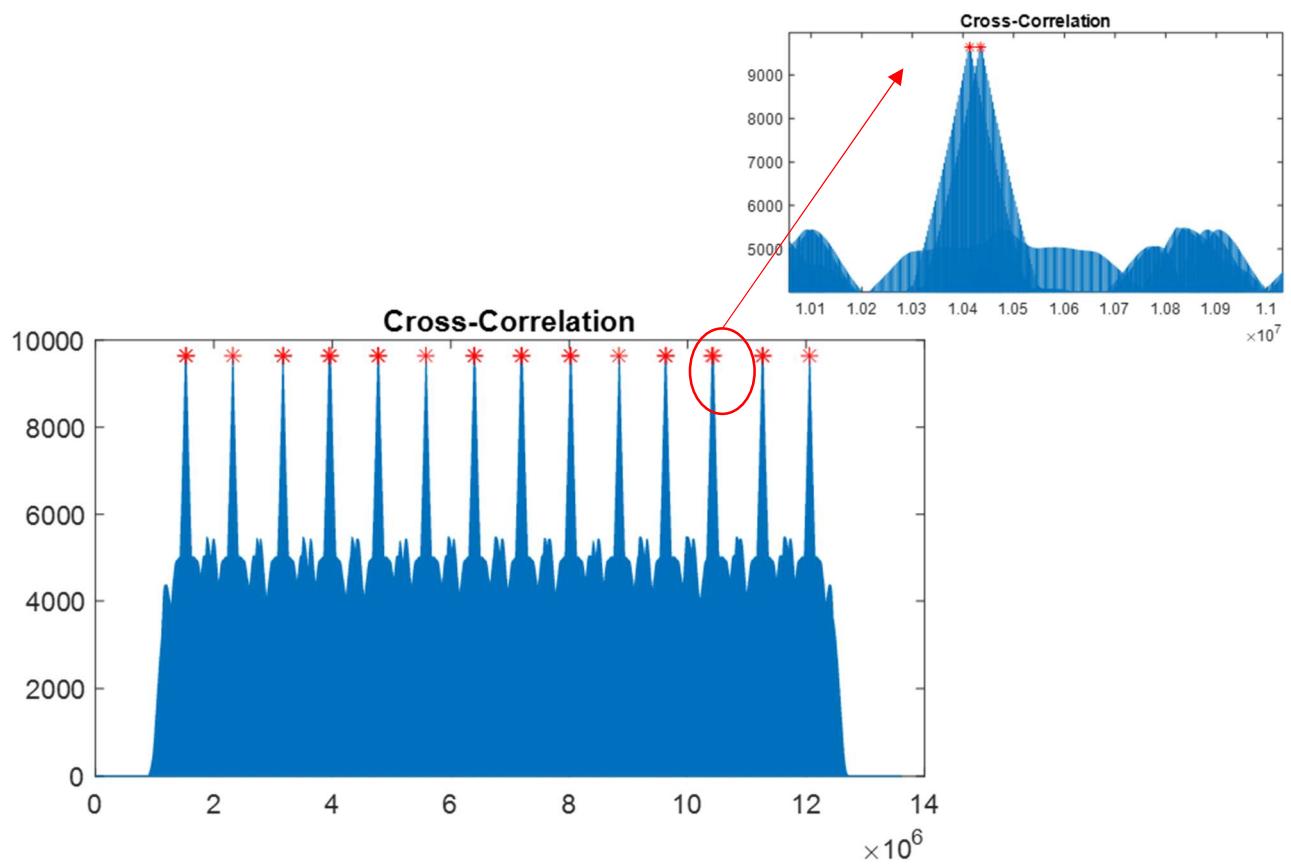
Se observa claramente un máximo en la posición en donde hay una cruz en la imagen, y podemos hallar la posición de cada máximo para relacionarla con las posiciones de las cruces en la matriz.

Otra forma de observar estos mismos datos es graficarlos como una función 1D en función de una variable índice. La relación entre índices y posiciones en una matriz es la observada en la siguiente figura

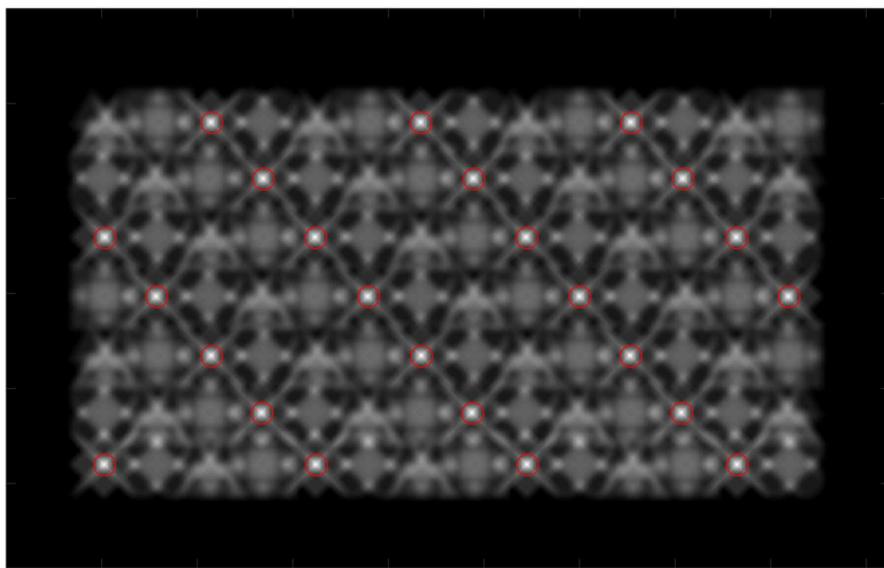


La función que relaciona índice y posición fila-columna en Matlab es `ind2sub`. Para la imagen del ejemplo, se puede observar el grafico de valor de nivel de gris en función del índice en la siguiente figura. Con cruces rojas marcamos

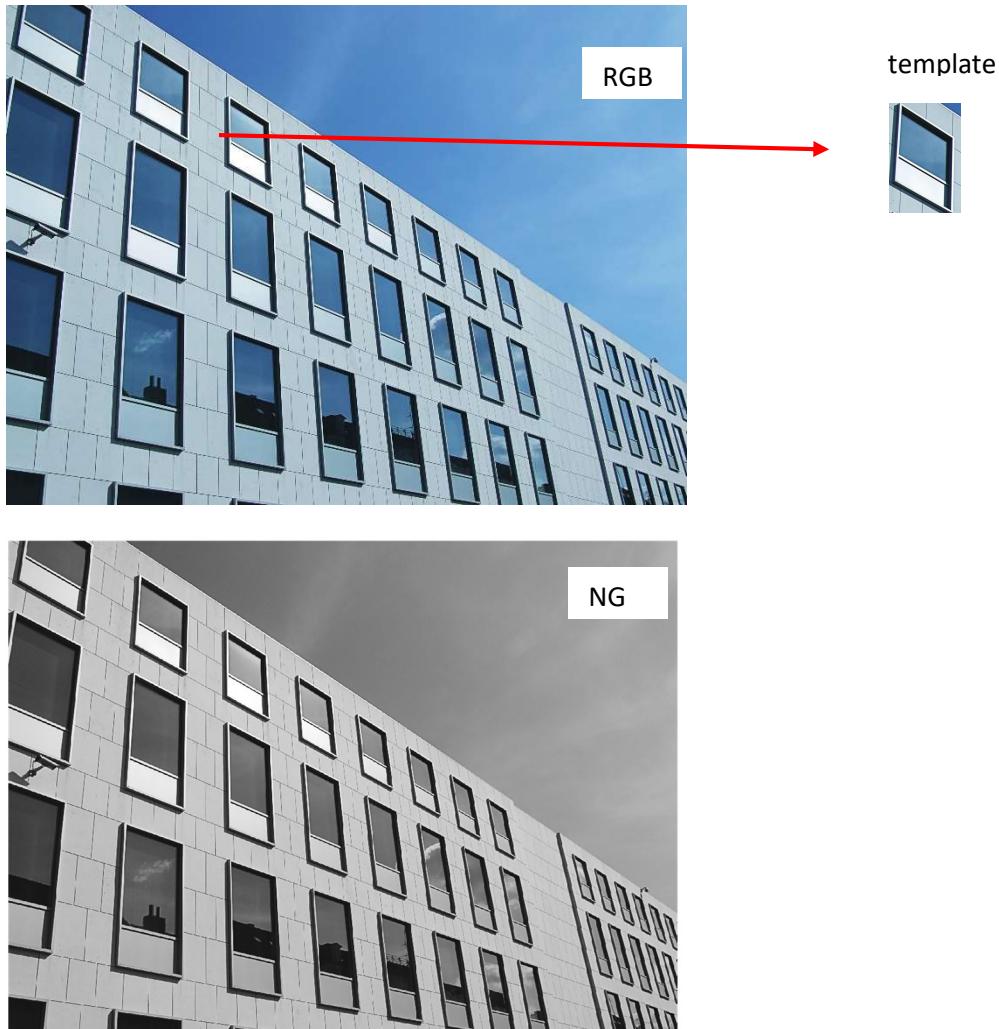
los máximos principales (ver en el zoom que en algunos lugares hay dos máximos muy cerca). En Matlab, cuando se calculan los máximos de una matriz, el resultado es el valor del máximo y su posición en índice.



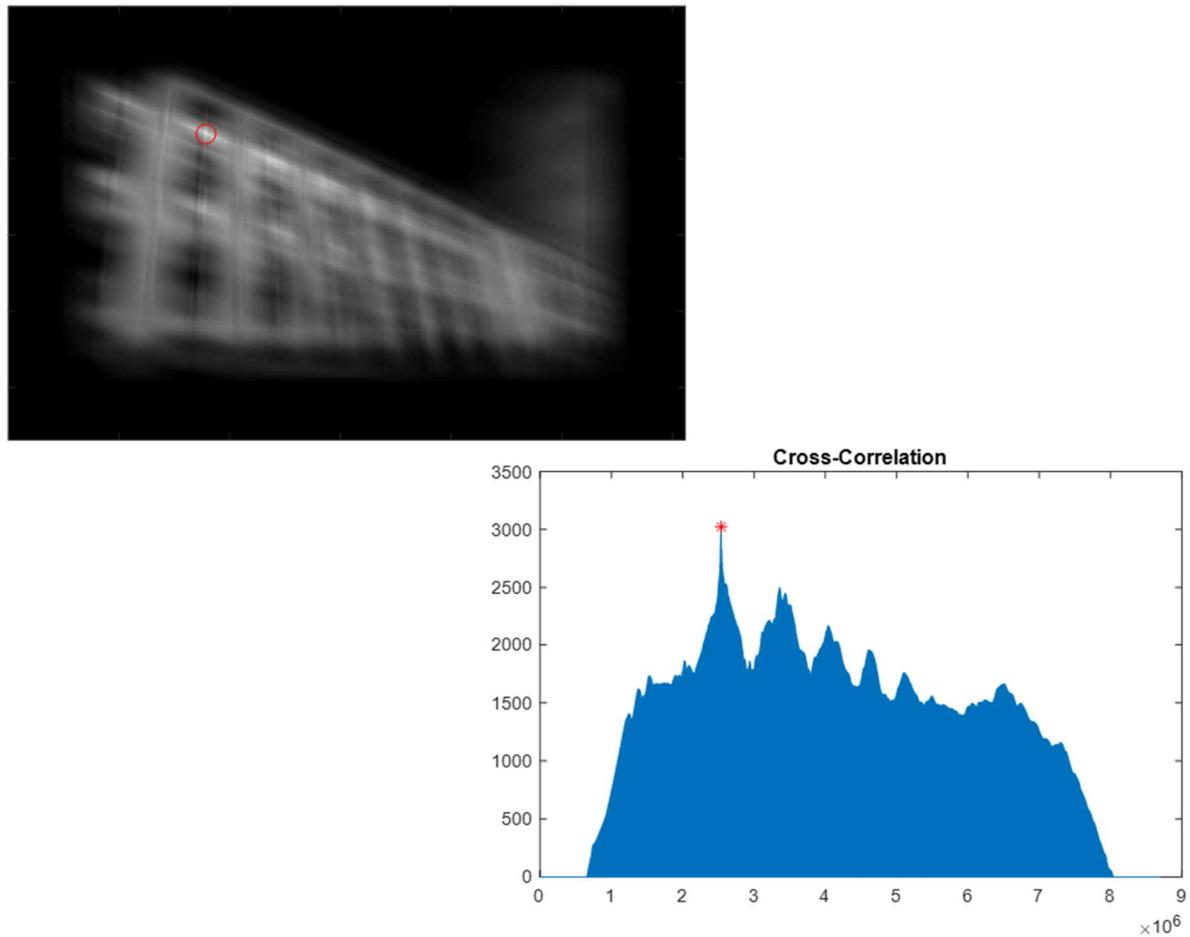
Por último, en la siguiente figura, vemos que esos son los máximos que corresponden a las posiciones de las cruces. Es decir, el lugar donde las imágenes son iguales es el lugar de máxima coincidencia entre la imagen y el template, y por lo tanto la correlación adquiere su valor máximo.



Veamos un ejemplo más sin tanto detalle. Queremos encontrar determinados patrones en la siguiente imagen (RGB). Por ejemplo, una ventana. A menos que queramos encontrar alguna estructura de color, vamos a trabajar con la imagen en niveles de gris (NG).



A continuación, observamos la imagen de correlación y el grafico en función del índice y vemos que tuvimos éxito en encontrar la ventana. ¿Ahora pensás que podrás encontrar a Wally?



Hay veces que la correlación genera falsos positivos, y depende mucho del template que esté buscando y las características de la imagen. Por ejemplo, que hubiera pasado si hubiéramos elegido como template una unión de ladrillos en el caso de las ventanas (brick joint). Como el template es muy uniforme, los valores de correlación van a ser grandes en zonas donde el template no coincide con la imagen. La correlación del template con la imagen se observa en la siguiente figura, y lo único que se obtuvo es un efecto de borroneado, como si se estuviera aplicando el filtro de la media, pero no se logró encontrar el detalle que se buscaba. (la imagen del experimento es ideal para usar la correlación)



Existen otros algoritmos que hacen posible encontrar patrones, por ejemplo, el que se usa en el tracker. Como fue explicado en la clase grabada, este algoritmo esencialmente busca minimizar la desviación cuadrática media de la imagen respecto del template. Veamos para el caso de imagen en niveles de gris.

El algoritmo define

$$DNG = \sum_{pixels} (NGI(i,j) - NGT(i,j))^2$$

Donde  $NGI$  son los niveles de gris de la imagen, y  $NGT$  los del template. El algoritmo busca minimizar este valor. No lo explicamos acá nuevamente porque está en el video, pero...desarrollemos el cuadrado y veamos

$$DNG = \sum_{pixels} NGI(i,j)^2 + NGT(i,j)^2 - 2 NGI(i,j)NGT(i,j)$$

Es decir que

$$DNG = \sum_{pixels} NGI(i,j)^2 + \sum_{pixels} NGT(i,j)^2 - 2 \sum_{pixels} NGI(i,j)NGT(i,j)$$

El último término es exactamente el valor de la correlación en la posición de máxima coincidencia.

## Referencias:

R. Gonzalez, R. Woods. Digital image processing. Pearson (2018)