Procesamiento de Imágenes

UNIDAD 2 – Transformación y Filtrado

Transformaciones de Intensidad

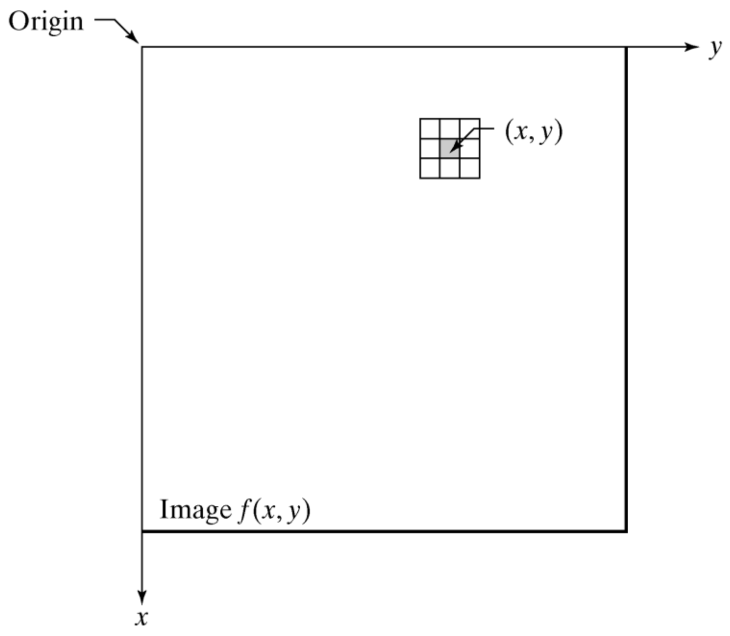


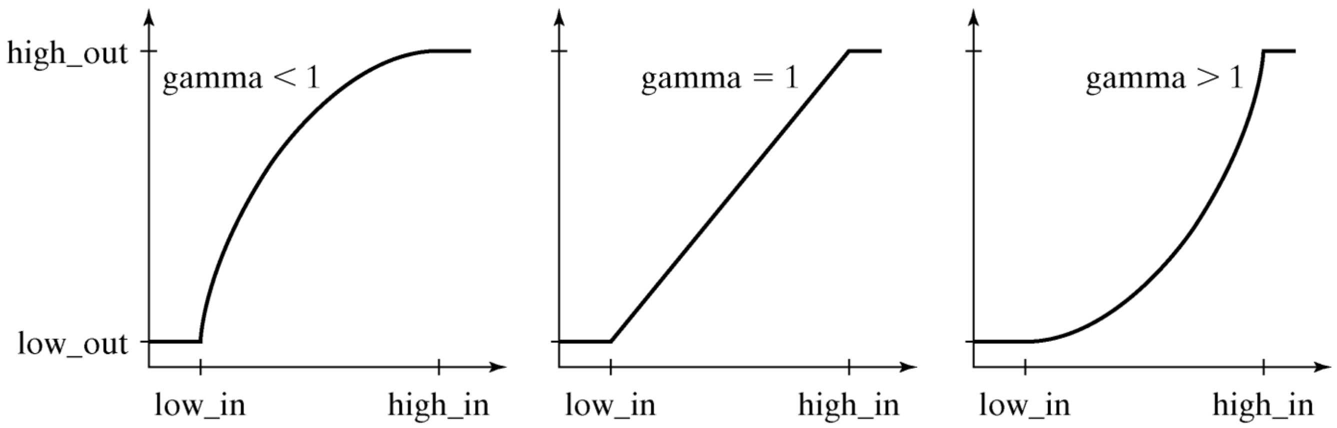
FIGURE 3.1 A neighborhood of size 3×3 about a point (x, y) in an image.

$f(x,y)$ pixel en imagen original

$$g(x,y) = T\left[f(x,y) \right]$$

T ⟹ operador en f definido en un entorno de (x,y)

Transformaciones de Intensidad



a b c

FIGURE 3.2 The various mappings available in function `imadjust`.

Ajuste de contraste ⟹ `imadjust()`

```
g = imadjust(f, vin=[low_in high_in], vout=[low_out high_out], gamma)
```

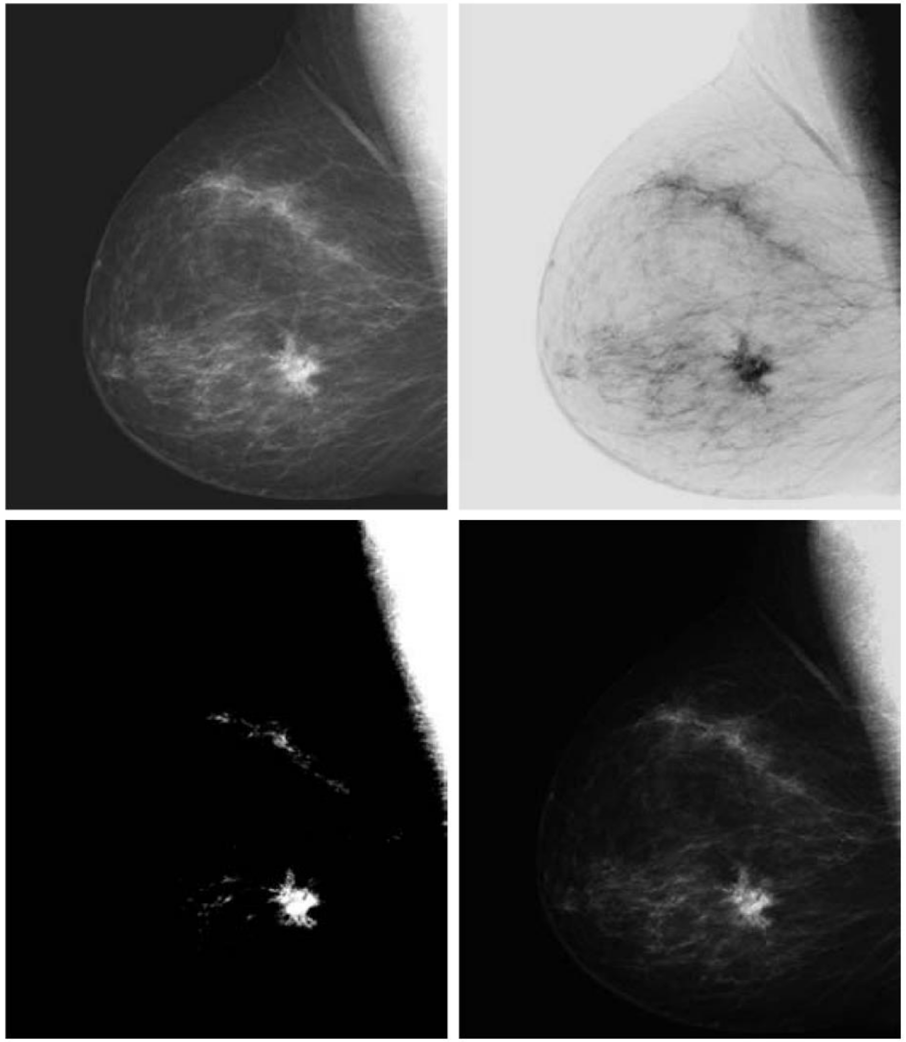
`gamma < 1` ⟹ más brillante

`gamma = 1` ⟹ lineal

`gamma > 1` ⟹ más oscura

Transformaciones de Intensidad

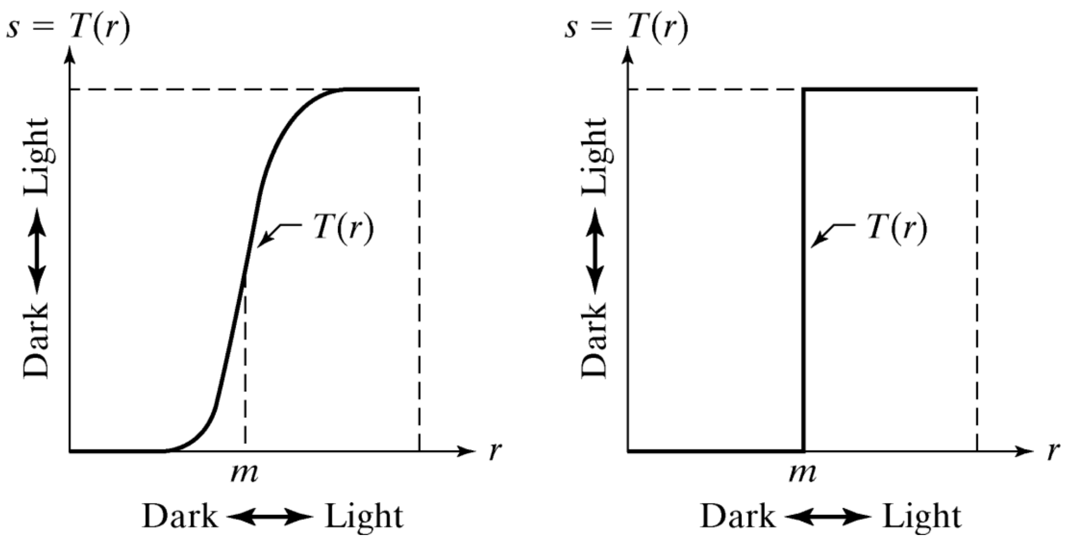
Tres diferentes transformaciones de intensidad para resaltar la región de interés (ROI), usando `imadjust()`.



a b
c d

FIGURE 3.3 (a) Original digital mammogram. (b) Negative image. (c) Result of expanding the intensity range [0.5, 0.75]. (d) Result of enhancing the image with `gamma = 2`. (Original image courtesy of G. E. Medical Systems.)

Transformaciones de Intensidad



a b

FIGURE 3.4 (a) Contrast-stretching transformation. (b) Thresholding transformation.

Transformación Logarítmica (compresión del rango dinámico):

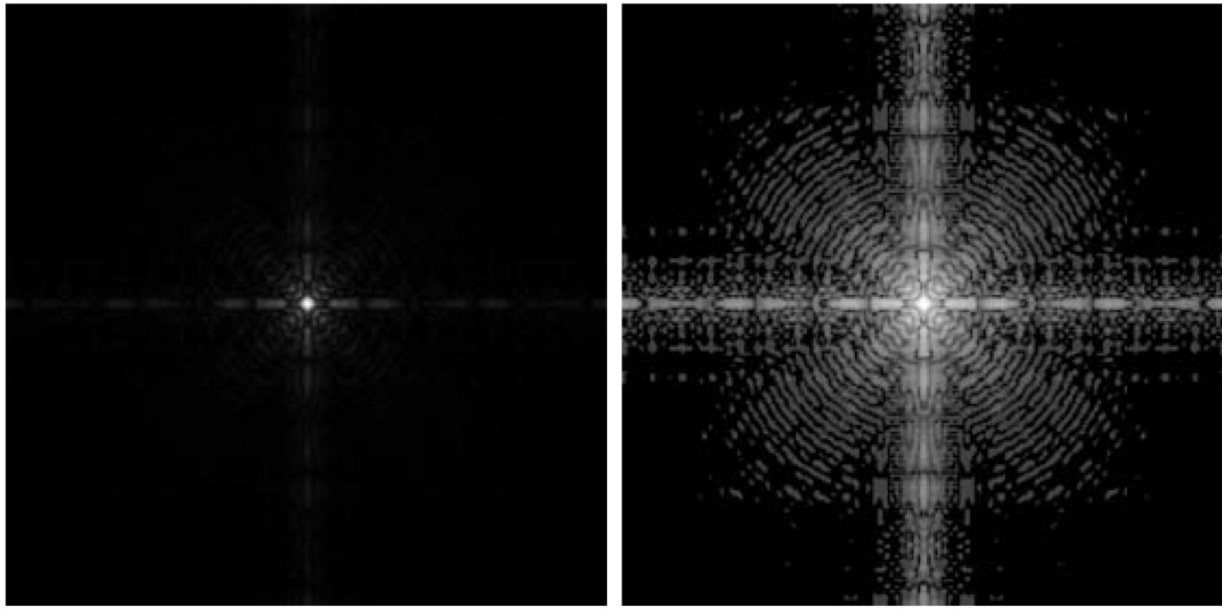
```
g = c*log(1+double(f))    ,  c constante
```

Transformación de estrechado de contraste:

$$s = T(r) = \frac{1}{1+(m/r)^E}$$

```
img2 = 1 / (1 + (m/(img + np.finfo(float).eps) )**E)
```

Transformaciones de Intensidad



a b

FIGURE 3.5 (a) A Fourier spectrum. (b) Result obtained by performing a log transformation.

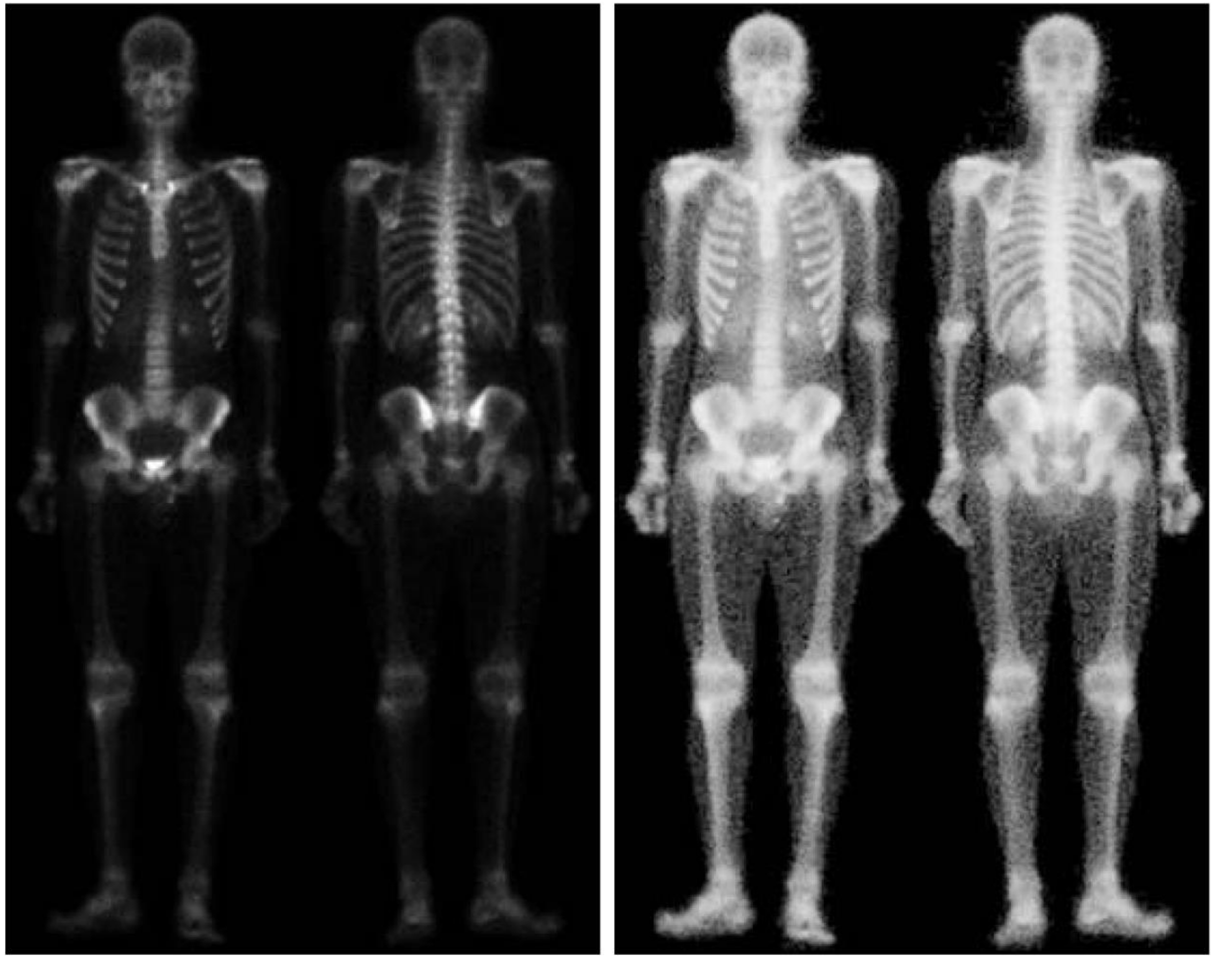
Los valores de la Transformada de Fourier suelen tener un gran rango dinámico [0 10^6] por lo que si se grafican en escala lineal se pierden los detalles visuales de los valores pequeños del espectro (Fig. 3.5(a)). Si se aplica la transformación log, un rango dinámico del orden de 10^6 se reduce a aproximadamente 14, que es más manejable (Fig. 3.5(b)).

Transformaciones de Intensidad

Se mejora la imagen haciendo estrechado de contraste con:

```
m = np.mean(img)

E = 0.9
```



a b

FIGURE 3.6 (a) Bone scan image. (b) Image enhanced using a contrast-stretching transformation. (Original image courtesy of G. E. Medical Systems.)

```
img2 = 1 / (1 + (m/(img + np.finfo(float).eps) )**E)
```

Transformaciones de Intensidad

Histograma

Histograma de una imagen

$$h(r_k) = n_k \qquad\qquad k = 1, 2, \cdots, L$$

donde r_k es el k-ésimo nivel de intensidad en el intervalo [0, G] y n_k es el número de pixeles de la imagen cuyo nivel de intensidad es r_k.

Histograma normalizado

$$p(r_k) = \frac{h(r_k)}{n} = \frac{n_k}{n}$$

n: número total de pixeles de la imagen

Transformaciones de Intensidad

Histograma

Vemos que $p(r_k)$ es una estima de la probabilidad de ocurrencia del nivel de intensidad r_k.

np.histogram(x, bins, range, ...)

```
hist, bins = np.histogram(img.flatten(), 256, [0, 256])
```

`img`: imagen de entrada

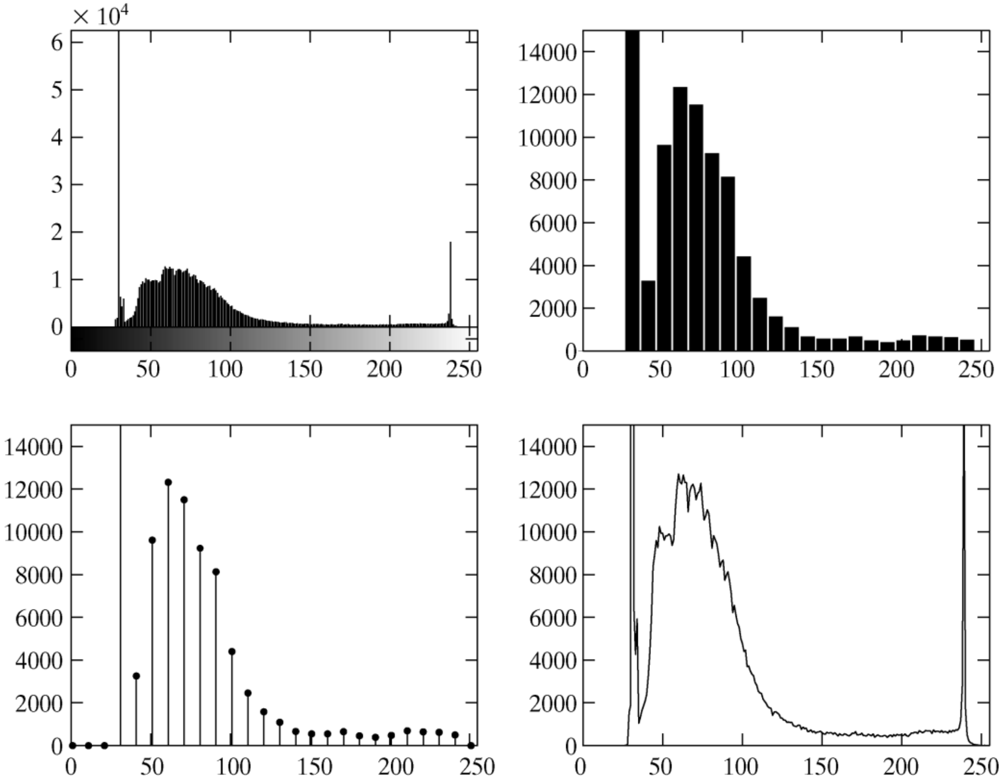
```
histn = hist.astype(np.double) / img.size
```

→ Histograma normalizado

Transformaciones de Intensidad

Histograma

Imagen de mamografía de Fig. 3.3(a)



a b
c d

FIGURE 3.7 Various ways to plot an image histogram. (a) `imhist`, (b) `bar`, (c) `stem`, (d) `plot`.

Transformaciones de Intensidad

Ecualización de histograma

Asuma que los niveles de intensidad de una imagen son continuos y que están normalizados en el rango [0, 1], y denotemos con $p_r(r)$ a la función de densidad de probabilidad (PDF) de los niveles de intensidad de una imagen. Llevemos a cabo la siguiente transformación:

$$s = T(r) = \int_0^r p_r(w)dw$$

Puede probarse que la función de densidad de probabilidad de los niveles de intensidad de la imagen de salida es uniforme, es decir:

Transformaciones de Intensidad

Ecualización de histograma

$$p_s(s) = \begin{cases} 1 & 0 \leq s \leq 1 \\ 0 & \text{c.o.c.} \end{cases}$$

Es decir, la transformación genera una imagen cuyos niveles de intensidad son igualmente probables, y cubren el rango completo [0 1]. El resultado es una imagen con mayor contraste. Note que $p_s(s)$ no es más que la función de distribución acumulada (CDF).

Para el caso de cantidades discretas, la transformación de ecualización resulta:

Transformaciones de Intensidad

Ecualización de histograma

$$s_k = T(r_k) = \sum_{j=1}^{k} p_r(r_j) = \sum_{j=1}^{k} \frac{n_j}{n}$$

donde s_k es el valor de intensidad de la imagen transformada correspondiente al valor r_k en la imagen de entrada.

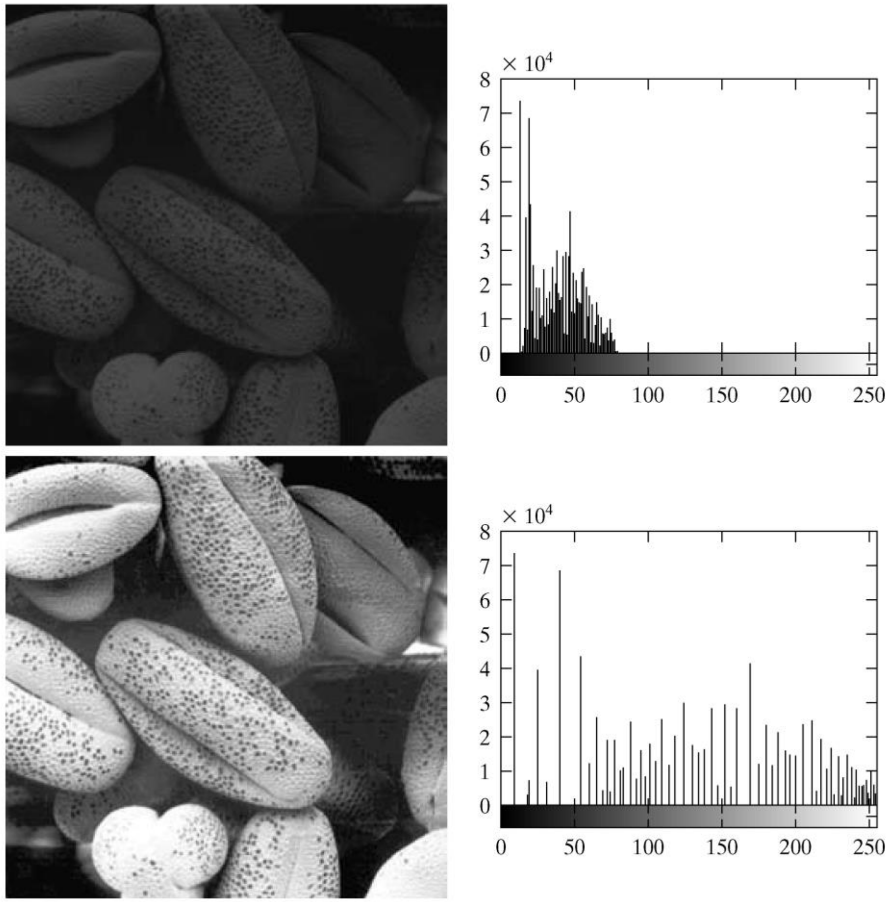
equalizeHist()

```
img_heq = cv2.equalizeHist(img)
```

Transformaciones de Intensidad

Ecualización de histograma

```
img_heq = cv2.equalizeHist(img)
ax1=plt.subplot(221)
plt.imshow(img,cmap='gray',vmin=0,vmax=255)
plt.subplot(222)
plt.hist(img.flatten(), 256, [0, 256])
plt.subplot(223,sharex=ax1,sharey=ax1)
plt.imshow(img_heq,cmap='gray',vmin=0,vmax=255)
plt.subplot(224)
plt.hist(img_heq.flatten(), 256, [0, 256])
plt.show()
```



a b
c d

FIGURE 3.8 Illustration of histogram equalization. (a) Input image, and (b) its histogram. (c) Histogram-equalized image, and (d) its histogram. The improvement between (a) and (c) is quite visible. (Original image courtesy of Dr. Roger Heady, Research School of Biological Sciences, Australian National University, Canberra.)

Transformaciones de Intensidad

Ecualización de histograma

```
hist,_= np.histogram(img.flatten(),256,[0,256])
histn = hist.astype(np.double) / img.size
cdf = histn.cumsum()
plt.plot(cdf)
plt.title('cdf')
plt.show()
```

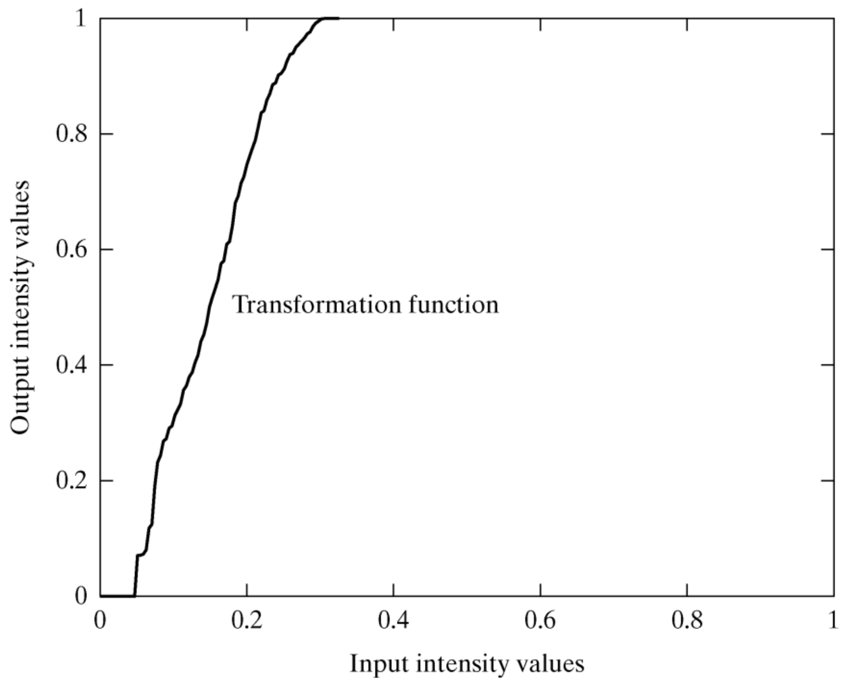


FIGURE 3.9 Transformation function used to map the intensity values from the input image in Fig. 3.8(a) to the values of the output image in Fig. 3.8(c).

Transformaciones de Intensidad

Ajuste (Especificación) de histograma

Sabemos que la transformación:

$$s = T(r) = \int_0^r p_r(w)dw$$

resulta en niveles de intensidad, s, que tienen una función de densidad de probabilidad $p_s(s)$ uniforme. Lo que queremos es generar una imagen procesada, con niveles de intensidad z, que tenga un histograma especificado. Sea $p_z(z)$ la función de densidad de probabilidad deseada, entonces el objetivo es hallar la transformación $H(z)$ tal que:

$$s = H(z) = \int_0^z p_z(w)dw$$

Transformaciones de Intensidad

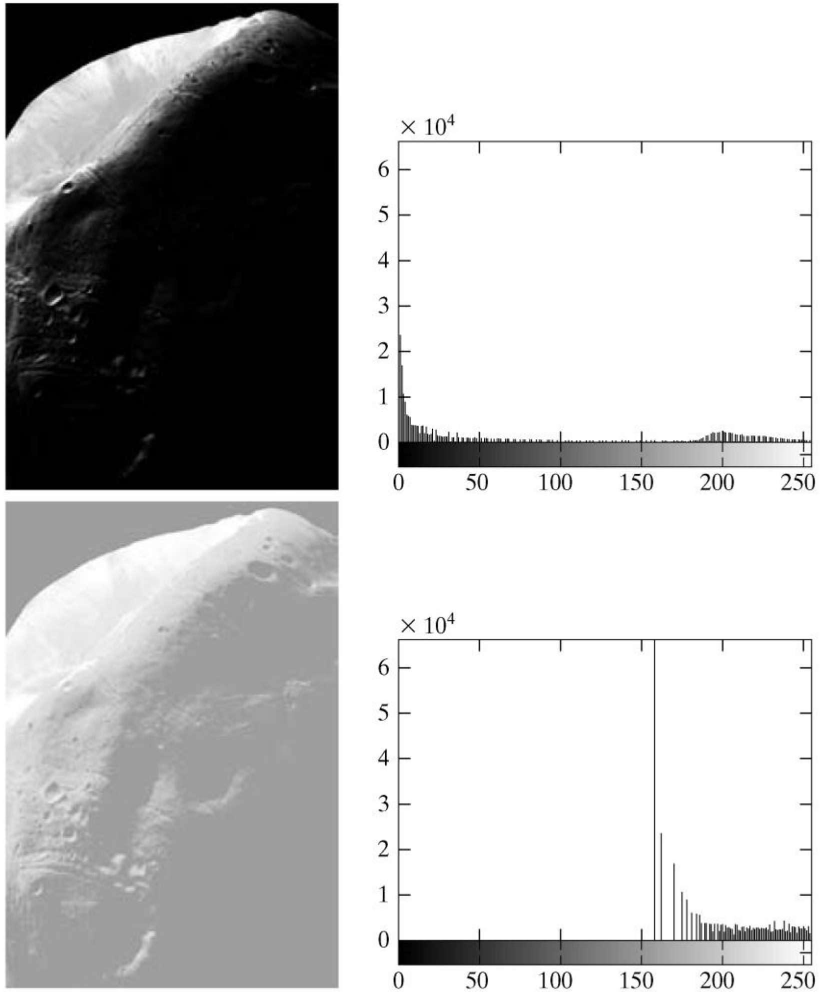
Ajuste (Especificación) de histograma

$$z = H^{-1}(s) = H^{-1}\left(T\left(r\right)\right)$$

Podemos hallar $T(r)$ con el método de ecualización de histograma visto anteriormente, por lo que podremos hallar los niveles de intensidad transformados z, cuya PDF es la especificada $p_z(z)$, siempre que podamos encontrar H^{-1}. Para el caso discreto H^{-1} existe siempre que $p_z(z)$ sea un histograma válido (área unitaria y todos sus valores no negativos).

Transformaciones de Intensidad

Ajuste (Especificación) de histograma

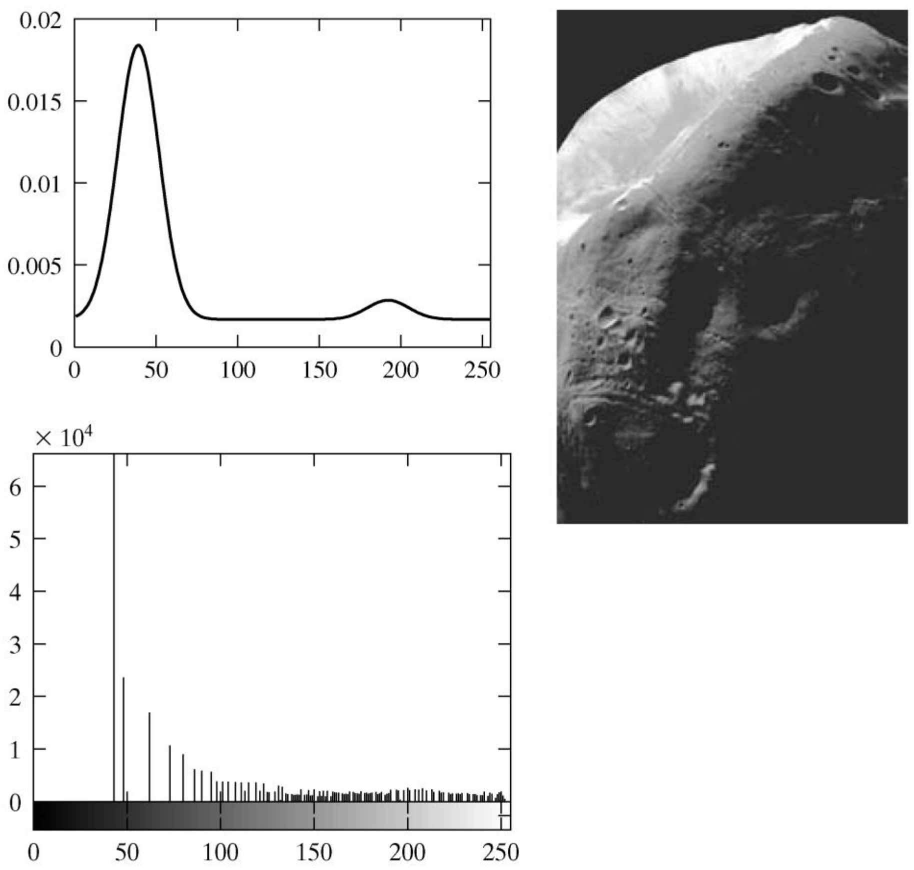


a b
c d

FIGURE 3.10 (a) Image of the Mars moon Phobos. (b) Histogram. (c) Histogram-equalized image. (d) Histogram of (c). (Original image courtesy of NASA).

Transformaciones de Intensidad

Ajuste (Especificación) de histograma



a b
c

FIGURE 3.11
(a) Specified histogram.
(b) Result of enhancement by histogram matching.
(c) Histogram of (b).

Transformaciones de Intensidad

Filtrado Espacial

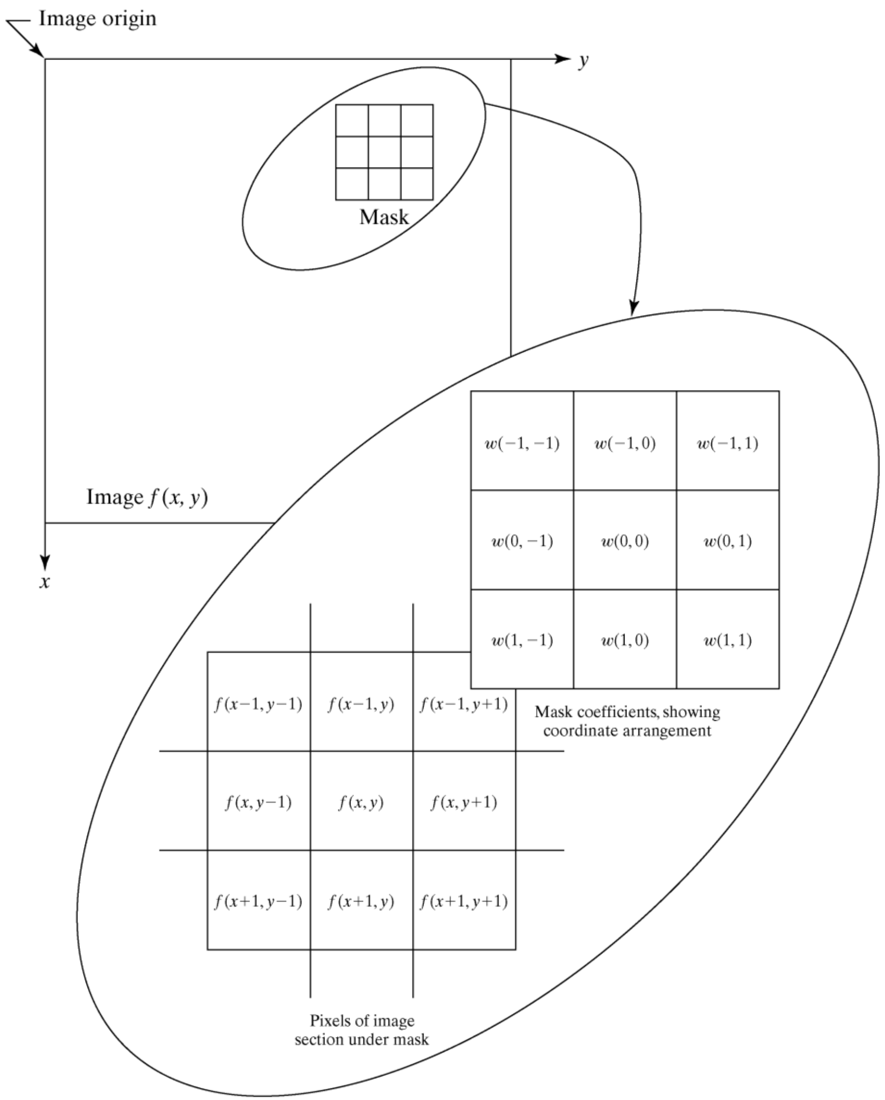


FIGURE 3.12 The mechanics of linear spatial filtering. The magnified drawing shows a 3×3 mask and the corresponding image neighborhood directly under it. The neighborhood is shown displaced out from under the mask for ease of readability.

Transformaciones de Intensidad

Filtrado Espacial Lineal

Para el caso de filtrado espacial lineal con una máscara de 3x3, el nivel de gris de cada pixel de la imagen filtrada, $g(x,y)$, se computa como:

$$
\begin{aligned}
g(x,y) &= f(x,y)w(0,0) + f(x-1,y-1)w(-1,-1) + f(x-1,y)w(-1,0) + \\
&+ f(x-1,y+1)w(-1,1) + f(x,y-1)w(0,-1) + f(x,y+1)w(0,1) + \\
&+ f(x+1,y-1)w(1,-1) + f(x+1,y)w(1,0) + f(x+1,y+1)w(1,1)
\end{aligned}
$$

La máscara se desplaza a lo largo de la imagen, esto equivale a hacer la convolución entre la máscara y la imagen.

Transformaciones de Intensidad

Filtrado Espacial Lineal

`filter2D(src, ddepth, kernel, dst, anchor, delta, borderType)`

- `src`: Imagen original.
- `ddepth`: Tipo de dato de la imagen de salida (profundidad) `dst`.
- `kernel`: Máscara del filtro.
- `anchor`: Ancla del kernel. Indica la posición relativa de un punto filtrado dentro del kernel.
- `delta`: Valor opcional que se suma a los píxeles filtrados antes de almacenarlos en dst.
- `borderType`: Se refiere a como se completan los bordes (padding) (ver `BorderTypes`).

```
w = np.ones((5, 5), np.float32) / (5*5)
img_fil = cv2.filter2D(img, -1, w, borderType=cv2.BORDER_DEFAULT)
```

Transformaciones de Intensidad

Filtrado Espacial Lineal

`filter2D(src, ddepth, kernel, dst, anchor, delta, borderType)`

Esta función, en realidad calcula la correlación, no la convolución:

$$\texttt{dst}(x,y) = \sum_{\substack{0 \le x' < \texttt{kernel.cols} \\ 0 \le y' < \texttt{kernel.rows}}} \texttt{kernel}(x',y') * \texttt{src}(x+x'-\texttt{anchor.x}, y+y'-\texttt{anchor.y})$$

Es decir, el kernel no se refleja alrededor del punto de anclaje. Pero, como la mayoría de los kernels utilizados son simétricos, el cálculo de la correlación coincide con la convolución.

Si se necesita realizar una convolución, se debe reflejar el kernel usando `flip` y se debe establecer una nueva ancla en:

`(kernel.cols - anchor.x - 1, kernel.rows - anchor.y - 1)`

Transformaciones de Intensidad

Filtrado Espacial Lineal

Correlación y Convolución en 1D

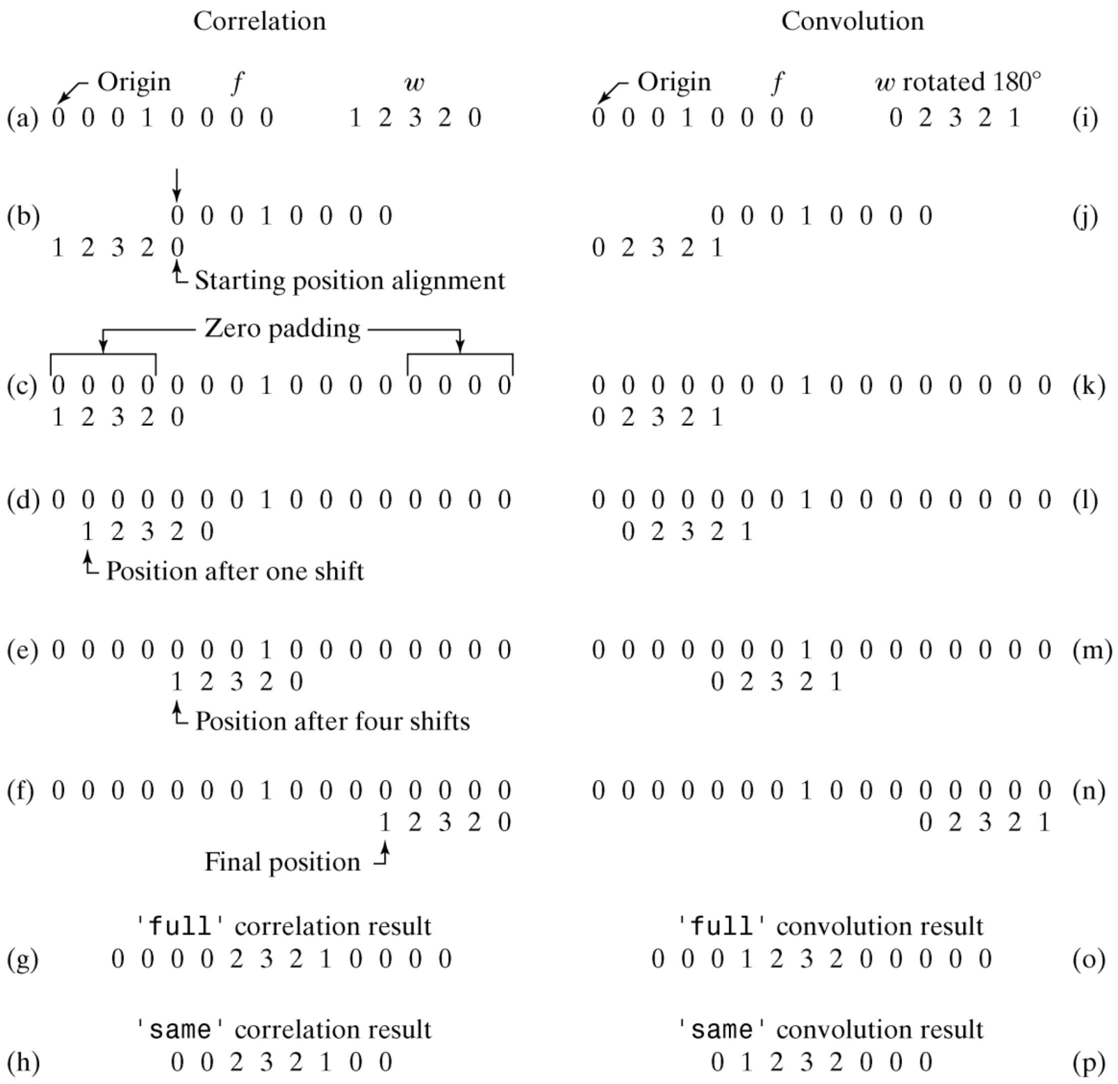


FIGURE 3.13 Illustration of one-dimensional correlation and convolution.

Transformaciones de Intensidad

Filtrado Espacial Lineal

Correlación y Convolución en 2D

Origin of $f(x, y)$

$$\begin{matrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{matrix} \qquad \begin{matrix} w(x, y) \\ \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix} \end{matrix}$$

(a)

Padded f

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

(b)

Initial position for w

$$\begin{matrix} 1&2&3&0&0&0&0&0&0 \\ 4&5&6&0&0&0&0&0&0 \\ 7&8&9&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{matrix}$$

(c)

'full' correlation result

$$\begin{matrix} 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&9&8&7&0&0&0 \\ 0&0&0&6&5&4&0&0&0 \\ 0&0&0&3&2&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{matrix}$$

(d)

'same' correlation result

$$\begin{matrix} 0&0&0&0&0 \\ 0&9&8&7&0 \\ 0&6&5&4&0 \\ 0&3&2&1&0 \\ 0&0&0&0&0 \end{matrix}$$

(e)

Rotated w

$$\begin{matrix} 9&8&7&0&0&0&0&0&0 \\ 6&5&4&0&0&0&0&0&0 \\ 3&2&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{matrix}$$

(f)

'full' convolution result

$$\begin{matrix} 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&2&3&0&0&0 \\ 0&0&0&4&5&6&0&0&0 \\ 0&0&0&7&8&9&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0 \end{matrix}$$

(g)

'same' convolution result

$$\begin{matrix} 0&0&0&0&0 \\ 0&1&2&3&0 \\ 0&4&5&6&0 \\ 0&7&8&9&0 \\ 0&0&0&0&0 \end{matrix}$$

(h)

FIGURE 3.14 Illustration of two-dimensional correlation and convolution. The 0s are shown in gray to simplify viewing.

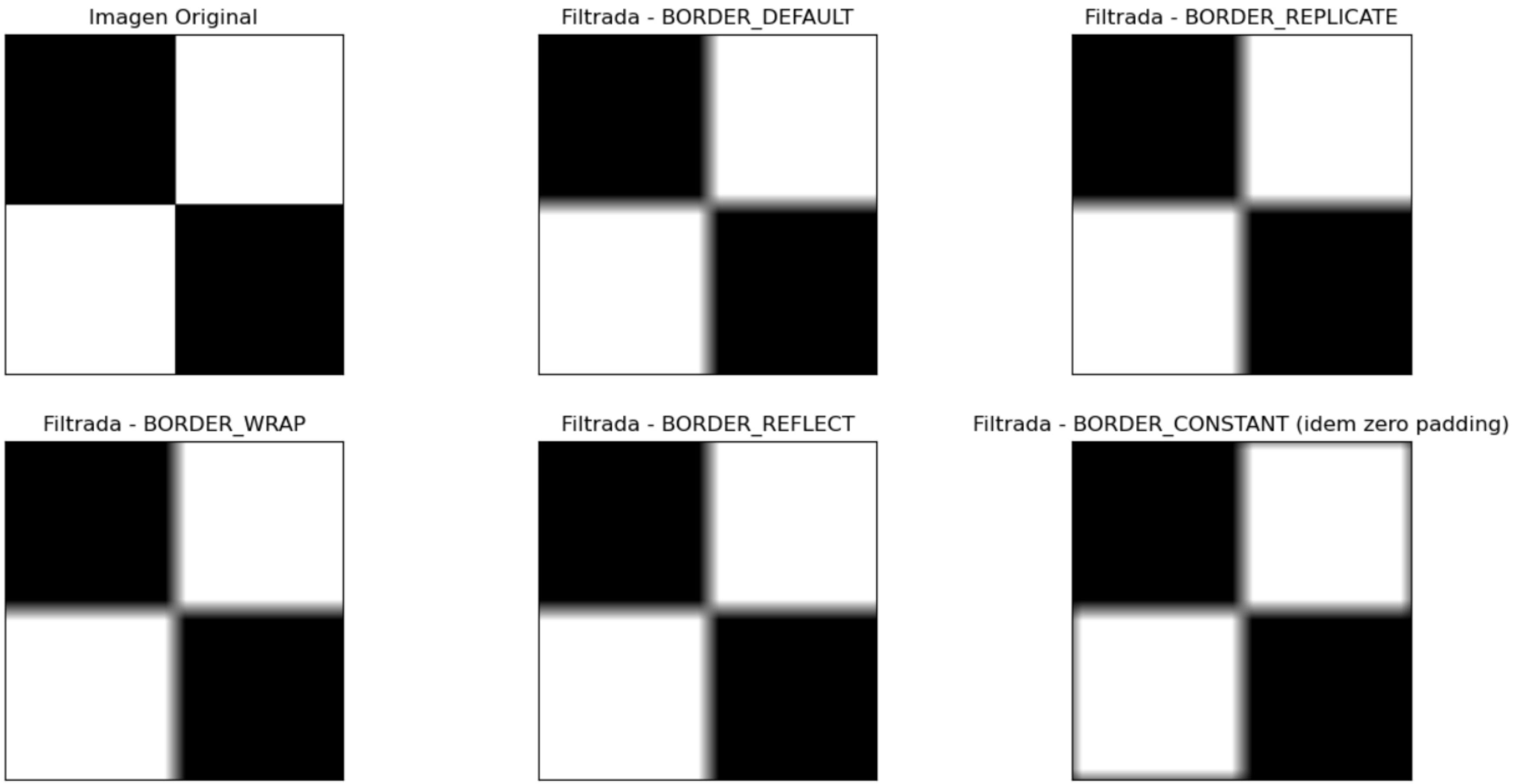
Transformaciones de Intensidad

Filtrado Espacial Lineal

Enumerator	
BORDER_CONSTANT Python: cv.BORDER_CONSTANT	`iiiiii|abcdefgh|iiiiiii` with some specified `i`
BORDER_REPLICATE Python: cv.BORDER_REPLICATE	`aaaaaa|abcdefgh|hhhhhhh`
BORDER_REFLECT Python: cv.BORDER_REFLECT	`fedcba|abcdefgh|hgfedcb`
BORDER_WRAP Python: cv.BORDER_WRAP	`cdefgh|abcdefgh|abcdefg`
BORDER_REFLECT_101 Python: cv.BORDER_REFLECT_101	`gfedcb|abcdefgh|gfedcba`
BORDER_TRANSPARENT Python: cv.BORDER_TRANSPARENT	`uvwxyz|abcdefgh|ijklmno`
BORDER_REFLECT101 Python: cv.BORDER_REFLECT101	same as BORDER_REFLECT_101
BORDER_DEFAULT Python: cv.BORDER_DEFAULT	same as BORDER_REFLECT_101
BORDER_ISOLATED Python: cv.BORDER_ISOLATED	do not look outside of ROI

Transformaciones de Intensidad

Filtrado Espacial Lineal




```
w = np.ones((31, 31), np.float32) / (31*31)
img_fil = cv2.filter2D(img, -1, w, borderType=cv2.BORDER_DEFAULT)
```

Transformaciones de Intensidad

Filtrado Espacial Lineal

Filtro Pasabajos

Los coeficientes de la máscara (filtro) deben ser todos positivos. El caso más simple es con todos los coeficientes iguales a 1. Para evitar que los valores de nivel de gris se vayan fuera del rango, se divide por la suma de los coeficientes del filtro.

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} \Rightarrow \text{Filtro de Promediación}$$

Transformaciones de Intensidad

Filtrado Espacial Lineal

El efecto del filtro es agregar borrosidad y la pérdida de detalles de la imagen. Usualmente se utiliza en una etapa de pre-procesamiento para eliminar pequeños detalles previo a la segmentación de objetos (más grandes) en la imagen.

Transformaciones de Intensidad

Filtrado Espacial Lineal

Filtro Pasa altos

Los coeficientes del filtro deben ser positivos cerca del centro y negativos en la periferia. Por ejemplo para una máscara de 3 x 3, un filtro pasa altos típico es el de más abajo. Notar que la suma de los coeficientes es nula, de manera que cuando se encuentra sobre una zona de nivel de gris constante, la salida de la máscara es nula.

$$\frac{1}{9}\times \begin{array}{|c|c|c|} \hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

Transformaciones de Intensidad

Filtrado Espacial Lineal

Notar que el filtro pasa altos elimina los componentes de frecuencia cero, por lo que el valor medio de niveles de gris de la imagen se reduce a cero, reduciendo de esta forma el contraste global de la imagen, pero resaltando los bordes.

Filtro High Boost

Una imagen filtrada pasa alto puede ser calculada como la diferencia entre la imagen original y una versión que ha pasado por un filtro pasabajo, es decir:

Pasa Alto = Original – Pasa Bajo

Si ahora multiplicamos a la imagen original por un factor de amplificación A, se obtiene un filtro high boost, o de énfasis de las altas frecuencias, *i.e.*

Transformaciones de Intensidad

Filtrado Espacial Lineal

High Boost = A x Original – Pasa Bajo

= (A -1) x Original + Original – Pasa Bajo

= (A -1) x Original + Pasa Alto

Un valor A = 1, da el filtro pasa alto normal. Cuando A > 1, parte de la imagen original se añade al resultado del filtro pasa alto, lo que devuelve parcialmente las bajas frecuencias. El resultado es que la imagen high boost se parece más a la original con un grado de mejora de los bordes, que depende del valor de A.

Transformaciones de Intensidad

Filtrado Espacial Lineal

Una máscara 3 x 3 para filtrado high boost, se muestra a continuación:

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline -1 & -1 & -1 \\ \hline -1 & w & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$$

donde w = 9.A - 1, con A >= 1

Transformaciones de Intensidad

Filtrado Espacial Lineal

Filtros derivadores

Destacan los bordes, donde el gradiente de intensidad es grande. El método más común de calcular la derivada es a través del gradiente. La magnitud del gradiente de una función *f(x,y)* viene dada por:

$$\nabla f = \left[\left(\frac{\partial f}{\partial x} \right)^2 + \left(\frac{\partial f}{\partial y} \right)^2 \right]^{1/2}$$

Transformaciones de Intensidad

Filtrado Espacial Lineal

Región de la imagen

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

El gradiente en el pixel $z_5 = f(x,y)$ puede aproximarse por:

$$z_5 = f\left(x, y\right)$$

$$\nabla f = \left|\left(z_7 + z_8 + z_9\right) - \left(z_1 + z_2 + z_3\right)\right| + \\ + \left|\left(z_3 + z_6 + z_9\right) - \left(z_1 + z_4 + z_7\right)\right|$$

Aproximación de Prewitt

-1	-1	-1
0	0	0
1	1	1

-1	0	1
-1	0	1
-1	0	1

Operadores de Prewitt

Transformaciones de Intensidad

Filtrado Espacial Lineal

Región de la imagen

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

$$z_5 = f\left(x, y\right)$$

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

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

Operadores de Sobel

Transformaciones de Intensidad

Filtrado Espacial Lineal

Filtro Laplaciano

Otra forma de enfatizar los bordes de una imagen es usando un filtro Laplaciano. El Laplaciano de una imagen *f(x,y)* se define como:

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

Las derivadas segundas pueden aproximarse digitalmente por:

$$\frac{\partial^2 f}{\partial x^2} \approx f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$$\frac{\partial^2 f}{\partial y^2} \approx f(x, y+1) + f(x, y-1) - 2f(x, y)$$

Transformaciones de Intensidad

Filtrado Espacial Lineal

La correspondiente aproximación digital del Laplaciano resulta entonces:

$$\nabla^2 f(x,y) \approx f(x+1,y) + f(x-1,y) + \\ + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

que corresponde a la máscara espacial:

0	1	0
1	-4	1
0	1	0

$\Rightarrow$ Máscara de filtro Laplaciano

Transformaciones de Intensidad

Filtrado Espacial Lineal

Una aproximación digital alternativa del Laplaciano, que tiene en cuenta los elementos diagonales, puede implementarse con la siguiente máscara:

1	1	1
1	-8	1
1	1	1

→ Máscara alternativa del filtro Laplaciano

Transformaciones de Intensidad

Filtrado Espacial Lineal

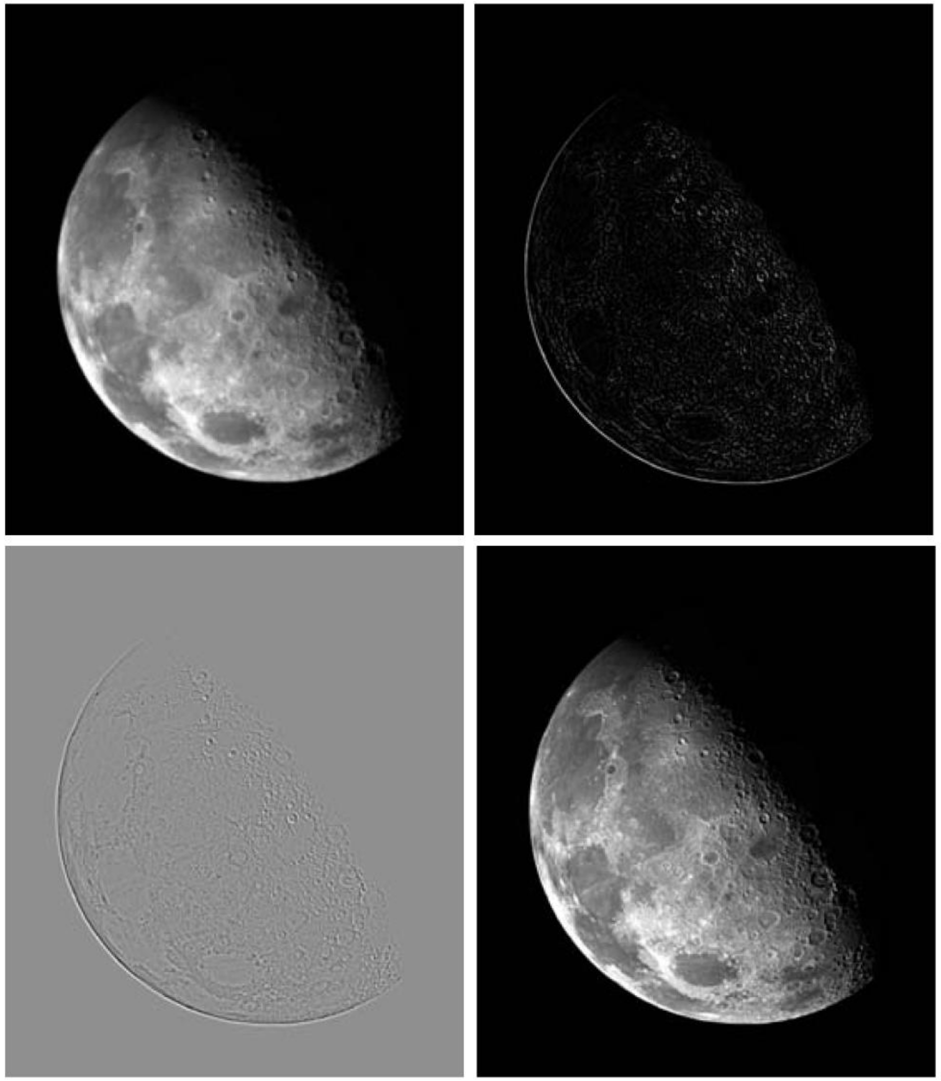
El mejoramiento de una imagen usando el Laplaciano se basa en la siguiente ecuación:

$$g(x,y) = f(x,y) + c\,\nabla^2 f(x,y)$$

donde $f(x,y)$ es la imagen original, $g(x,y)$ es la imagen mejorada, y c es 1 si el coeficiente central de la máscara es positivo, o -1 si es negativo. Como el Laplaciano es un operador derivada, enfatiza los bordes pero lleva las áreas de intensidad constante a cero. Al sumar la imagen original se restablecen los niveles de gris de la imagen (ver filtros high boost).

Transformaciones de Intensidad

Filtrado Espacial Lineal



a	b
c	d

FIGURE 3.16 (a) Image of the North Pole of the moon. (b) Laplacian filtered image, using `uint8` formats. (c) Laplacian filtered image obtained using `double` formats. (d) Enhanced result, obtained by subtracting (c) from (a). (Original image courtesy of NASA.)

Transformaciones de Intensidad

Filtrado Espacial Lineal

FIGURE 3.17 (a) Image of the North Pole of the moon. (b) Image enhanced using the Laplacian filter `'laplacian'`, which has a -4 in the center. (c) Image enhanced using a Laplacian filter with a -8 in the center.

Transformaciones de Intensidad

Filtrado Espacial No Lineal

Los filtros no lineales también operan en entornos de un pixel, pero, en general su operación está basada directamente en los valores de intensidad de los pixeles del entorno. No se asignan coeficientes a una máscara y se realiza la convolución con la imagen.

Filtro de mediana

El nivel de gris de cada pixel es reemplazado por la mediana de los niveles de gris del entorno del pixel, en vez de por el promedio. Este filtrado es efectivo cuando el patrón de ruido es de tipo impulsivo y se quieren preservar los bordes.

Transformaciones de Intensidad

Filtrado Espacial No Lineal

`imnoise_salt_pepper(img, p)`

Agrega ruido salt and peper a la imagen de entrada `img`. `p` es el porcentaje de píxeles afectados por el ruido.

`medianBlur(img, ksize)`

Aplica un filtro de mediana sobre la imagen de entrada `img`. El tamaño del entorno está definido por `ksize`.

```
img_sap = imnoise_salt_pepper(img, 0.3)
img_sap_filt = cv2.medianBlur(img_sap, 3)
```

Transformaciones de Intensidad

Filtrado Espacial No Lineal