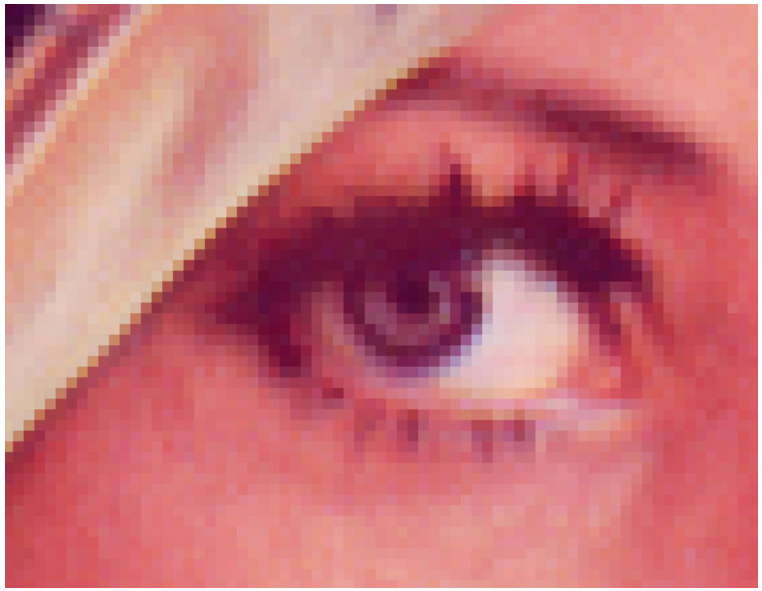
Introducción al procesamiento digital de imágenes usando python

Ricardo Amézquita Orozco

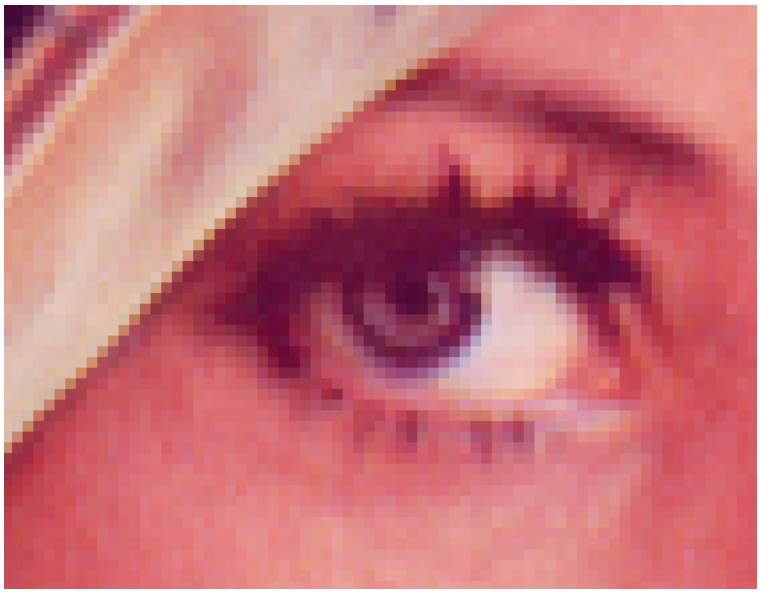
4 de noviembre de 2015

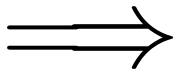
¿Que es una imagen?

¿Digitalmente que es una imagen?



¿Como representamos una imagen?





```
array([[[ 82,  22,  57],
        [ 82,  22,  57],
        [ 96,  32,  62],
        ...,
        [179,  70,  79],
        [181,  71,  81],
        [185,  74,  81]],

       [[ 82,  22,  57],
        [ 82,  22,  57],
        [ 96,  32,  62],
        ...,
        [179,  70,  79],
        [181,  71,  81],
        [185,  74,  81]],

       [[ 84,  18,  60],
        [ 84,  18,  60],
        [ 92,  27,  58],
        ...,
```

Tipos de imágenes

Color

Tonos de Gris

Indexado

Trabajando con imágenes

Abrir una imagen

```
imc=imread("lena_std.tif")
imb=imread("lena_bw.tif")
```

Mirando el tamaño de una imagen

```
imc.shape
imb.shape
```

Trabajando con imágenes

Visualizando una imagen

```
imshow(imc)
imshow(imb)
imshow(imb,origin="lower")
imshow(imb,origin="lower", cmap="gray")
```

Modificando una imagen

```
imc[:,:,0]=0
imb[200:300,200:300]=255
```

Binarizando una imagen

```
imbin=where(imb<128,0,255)
```

Trabajando con imágenes

Guardando una imagen

```
imsave("im.png",imb,cmap="gray",origin="lower")
```

Nota: La función imsave de pylab no guarda imágenes en RGB. Para poder hacer esto, toca usar:

```
scipy.misc.imsave
```

Scipy

Paquete con rutinas para calculo en ciencias e ingeniería basado en Numpy, posee un modulo con rutinas para procesamiento de imágenes:

```
>>>: import scipy.ndimage as nd
```

Que contiene herramientas para las siguientes operaciones:

Filtros	Filtros Fourier	Medidas	Morfología	Interpolación
convolve	fourier_elipsoid	center_of_mass	binary_clossing	affine_transform
correlate	fourier_gaussian	extrema	binary_dilation	geometric_transform
maximum_filter	fourier_shift	label	binary_erosion	map_coordinates
median_filter	fourier_uniform	maximum	binary_fill_hole	shift
minimum_filter		mean	binary_hit_or_miss	spline_filter
.....		.....	.....	zoom

Cuadro: Algunas funciones para procesamiento de imágenes definidas en scipy.ndimage

Filtrando imágenes

Imagen con ruido

Imagen original

Filtrando imágenes

Mediana

Imagen con ruido

Imagen filtrada

```
im_filtrada=nd.median_filter(im_ruido,(3,3))
```

Filtrando imágenes

Máximo

Imagen con ruido

Imagen filtrada

```
im_filtrada=nd.maximum_filter(im_ruido,(3,3))
```

Filtrando imágenes

Mínimo

Imagen con ruido

Imagen filtrada

```
im_filtrada=nd.minimum_filter(im_ruido,(3,3))
```

Filtrando imágenes

Convolución

1	2	3
4	5	6
7	8	9

Entrada

1	2	3
0	0	0
-1	-2	-3

Kernel $(m \times n)$

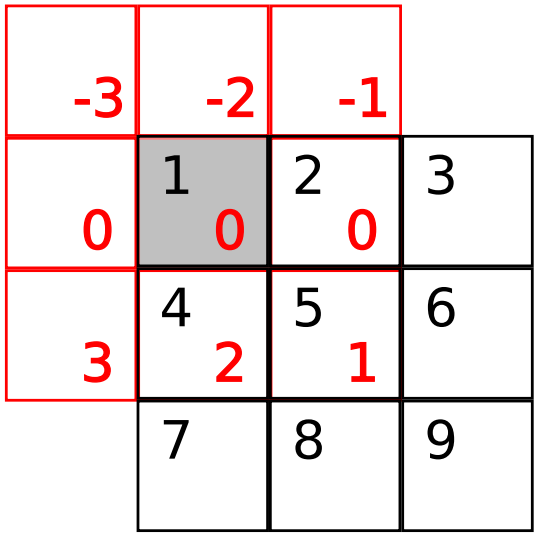
13	28	27
18	36	30
-13	-28	-27

Salida

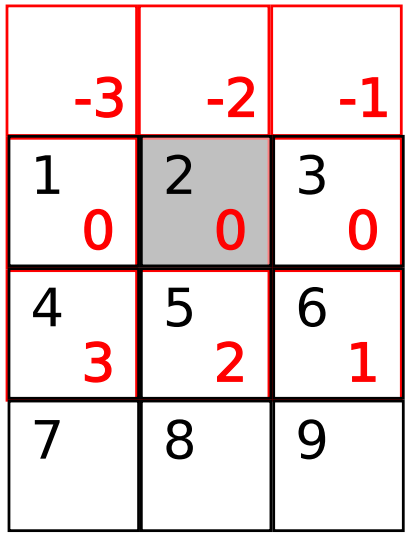
$$S = E * K = \sum_j \sum_i E[i,j] \times K[m-i, n-j]$$

Filtrando imágenes

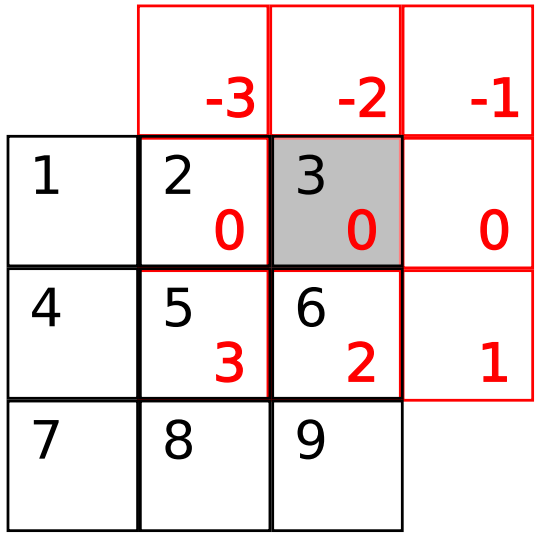
Convolución



13



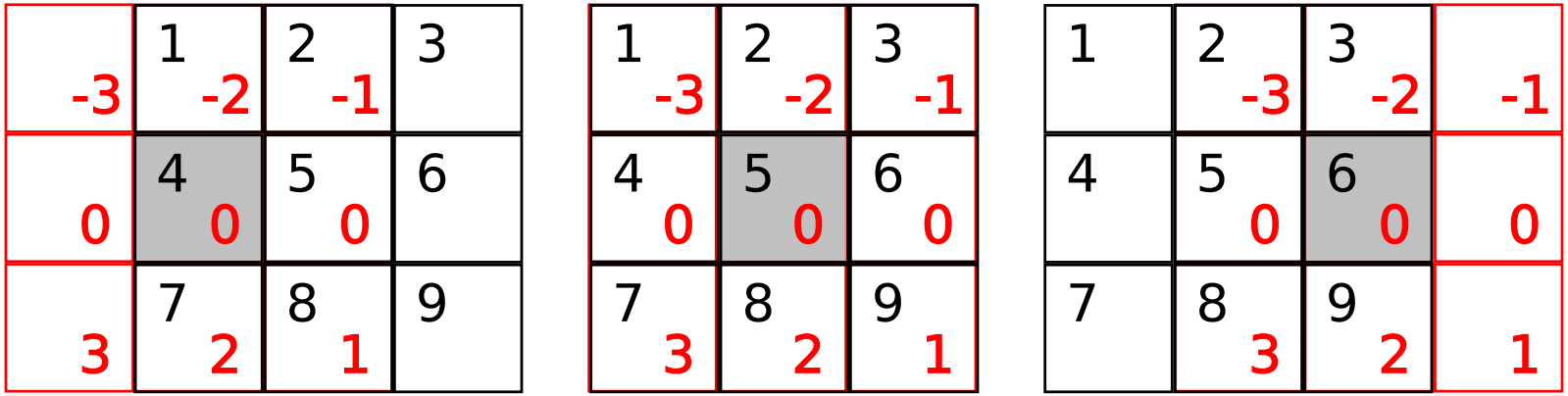
28



27

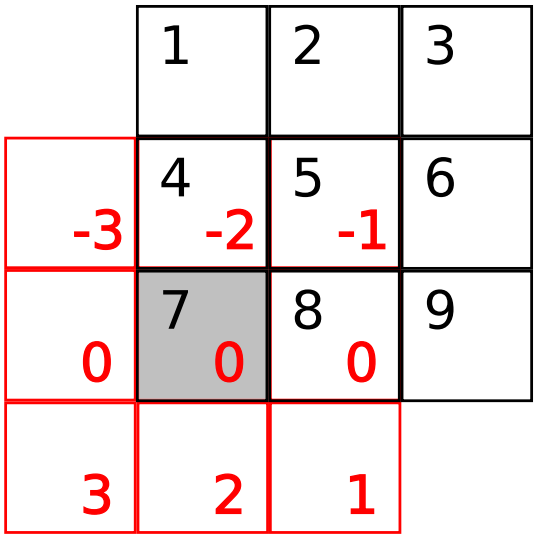
Filtrando imágenes

Convolución

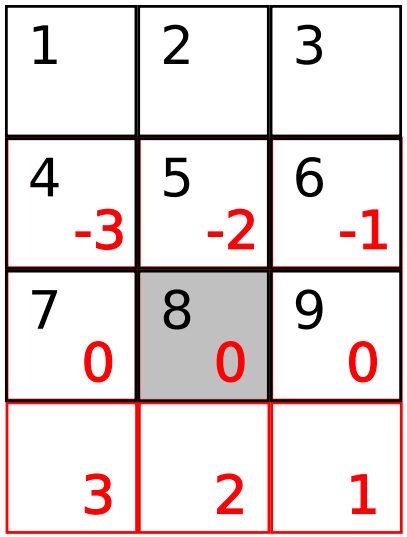


Filtrando imágenes

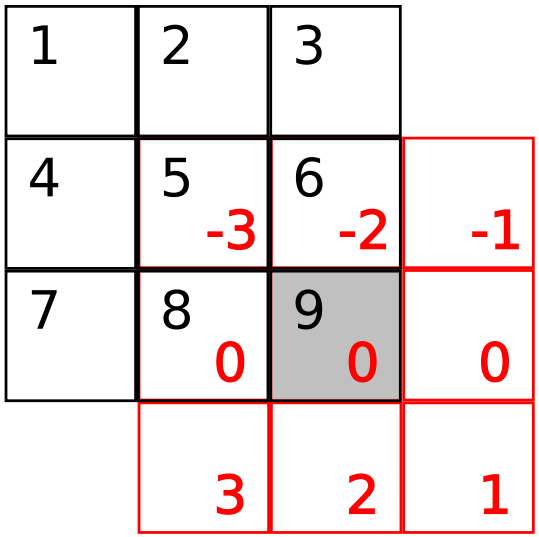
Convolución



-13



-28



-27

Filtrando imágenes

Convolución pasa bajo

Imagen con ruido

Imagen filtrada

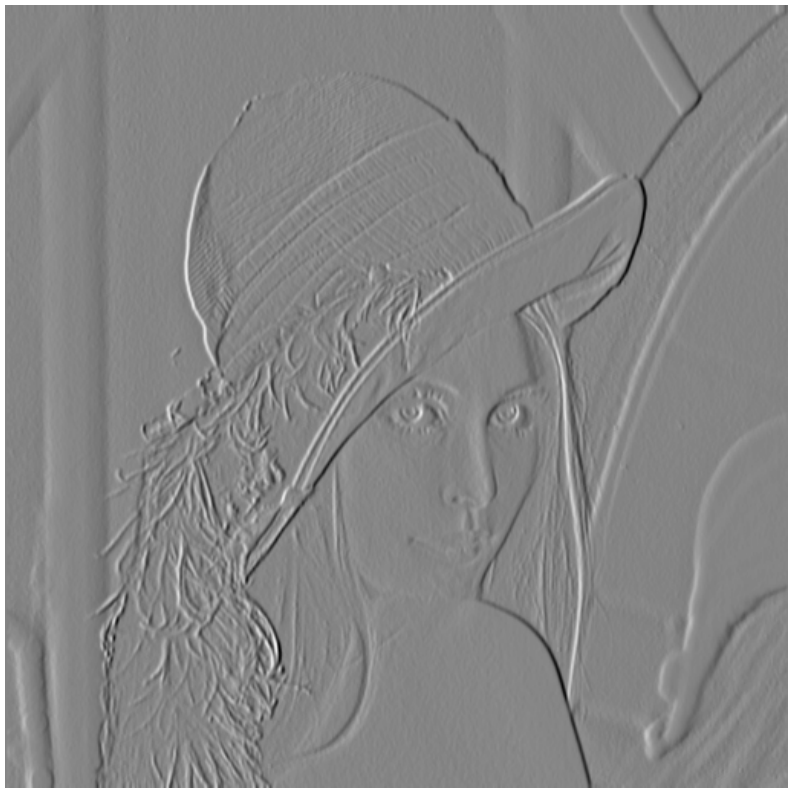
$$\text{Kernel}=\frac{1}{5}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

im_filtrada=nd.convolve(im_ruido,Kernel)

Filtrando imágenes

Convolución detección de bordes verticales

Imagen original



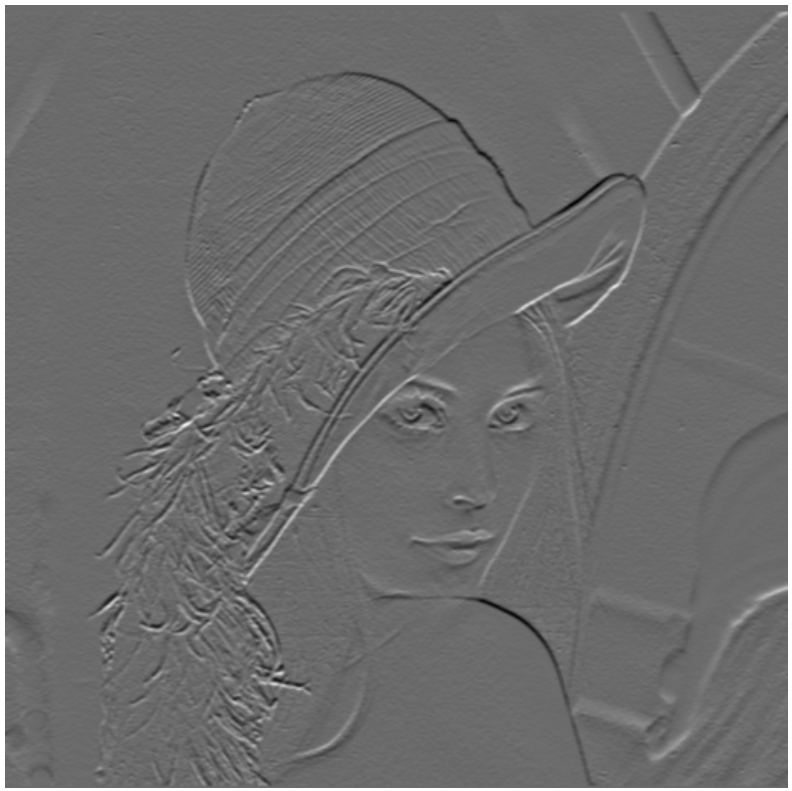
Bordes verticales

$$\text{Kernel}=\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad \text{im\_filtrada=nd.convolve(im\_ruido,Kernel)}$$

Filtrando imágenes

Convolución detección de bordes horizontales

Imagen original



Bordes verticales

$$\text{Kernel}=\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

im_filtrada=nd.convolve(im_ruido,Kernel)

Filtrado de imágenes

Correlación

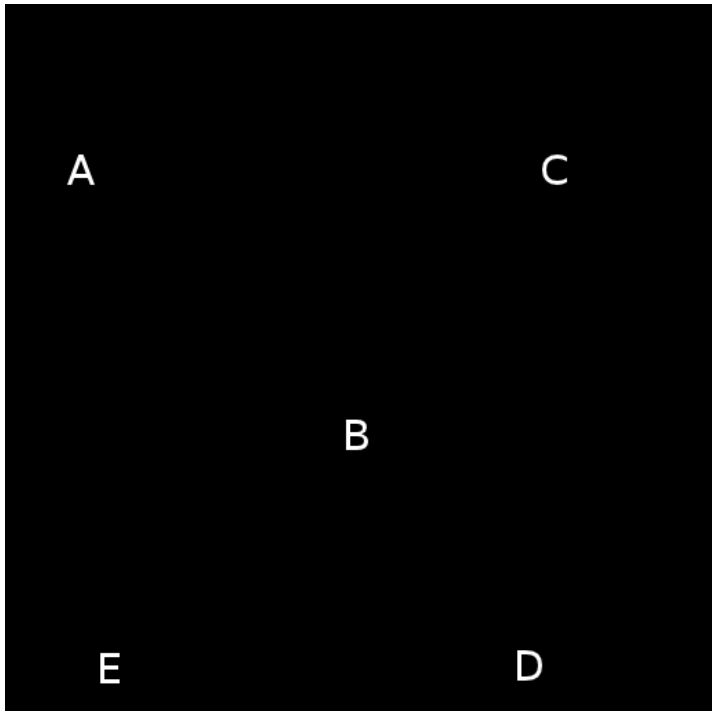
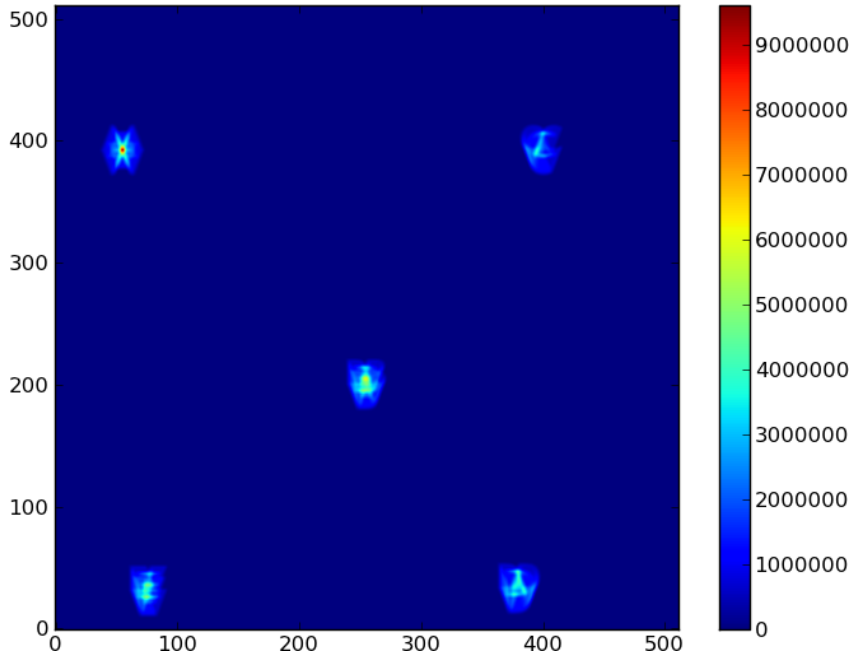


Imagen original

Patrón

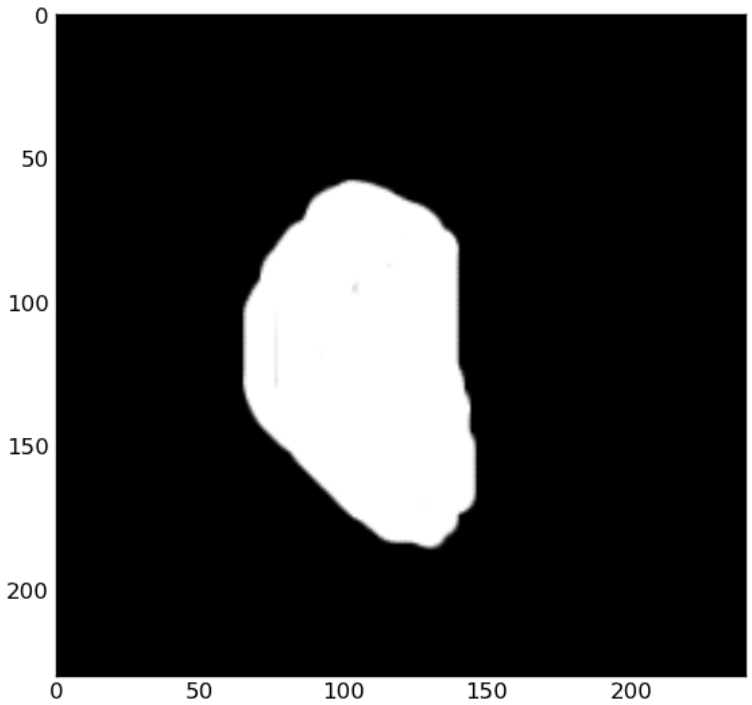


Correlación

```
im_filtrada=nd.correlate(original,patron)
```

Realizando medidas sobre imágenes

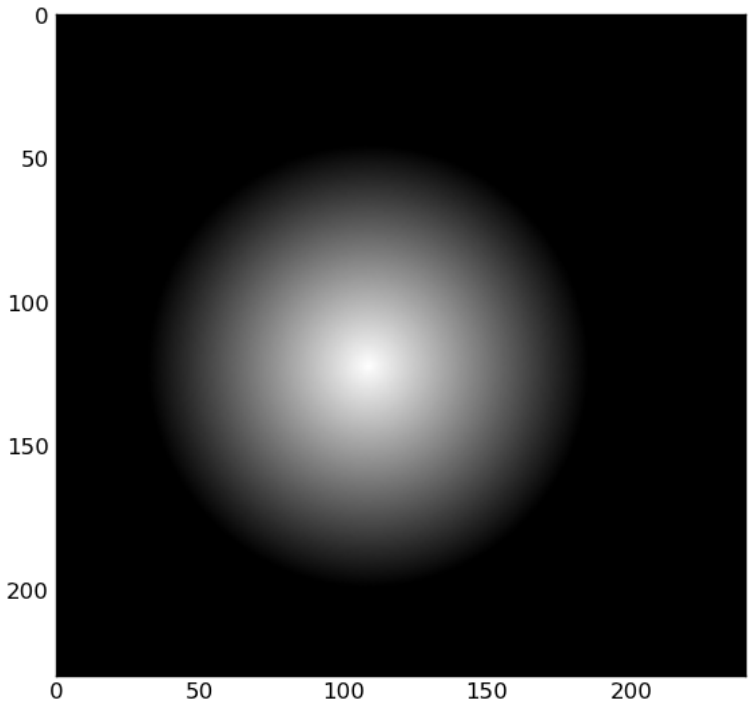
Centro de masa




```
In [4]: nd.center_of_mass(image)
Out[4]: (120.28234031492804, 107.43842755896448)
```

Realizando medidas sobre imágenes

extremos




```
In [29]: nd.extrema(image)
Out[29]: (0, 255, (0, 0), (122, 108))
```

Realizando medidas sobre imágenes

etiquetas

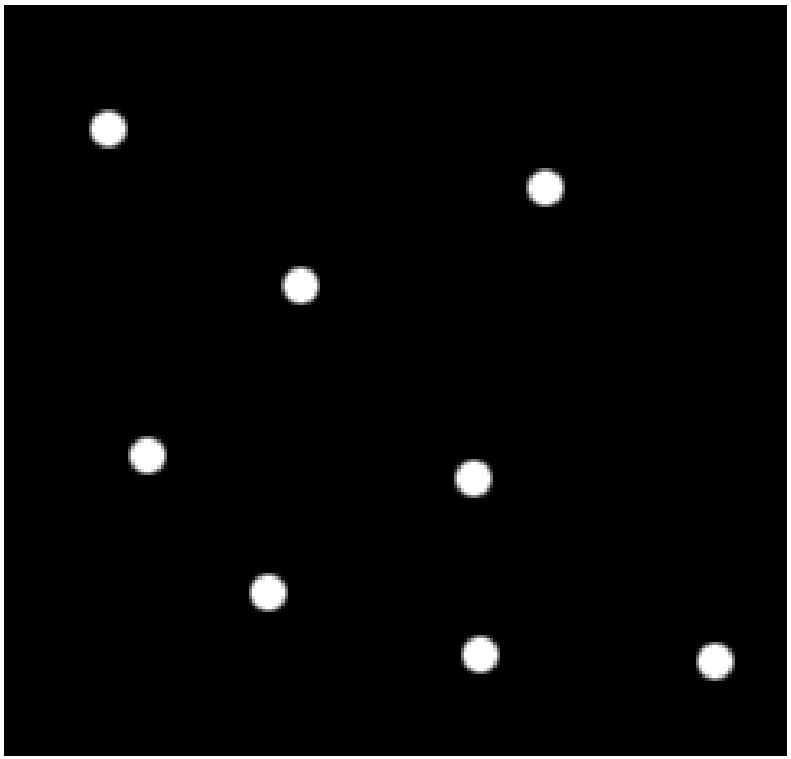


Imagen original

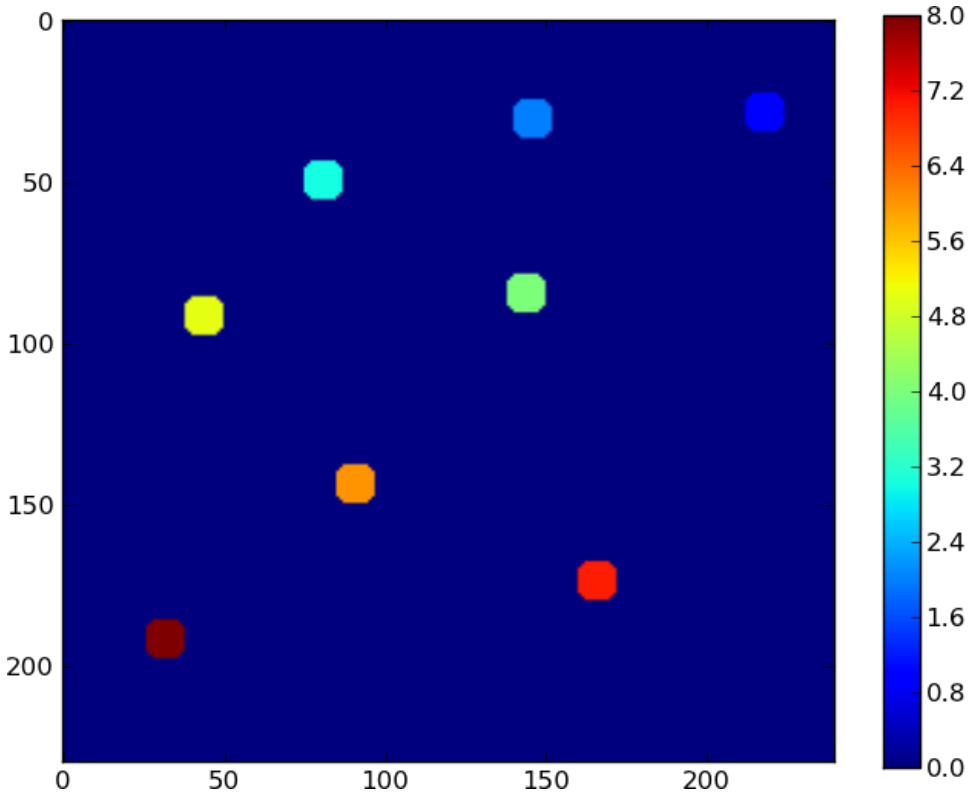


Imagen etiquetada

```
lblim,n=nd.label(image)
```

Taller

- Hacer un programa que tome la imagen de los granos de café, la procese y retorne los siguientes datos:
  - Cuantos granos de café hay en la imagen
  - Cual es la posición de los centros de masa de cada grano de café.
    - Ayuda: Busque la documentación de la función center_of_mass y verifique como puede usar esta junto con los resultados entregados por la función labels para resolver este problema.
  - Encuentre las coordenadas de las esquinas de los rectángulos que encierran cada grano de café