



**UAI**

UNIVERSIDAD ADOLFO IBÁÑEZ  
FACULTAD DE INGENIERÍA Y CIENCIAS



**iUAI**  
UNIVERSIDAD ADOLFO IBÁÑEZ  
FACULTAD DE INGENIERIA Y CIENCIAS

MDS<sup>2019</sup> PROCESAMIENTO  
DE IMÁGENES

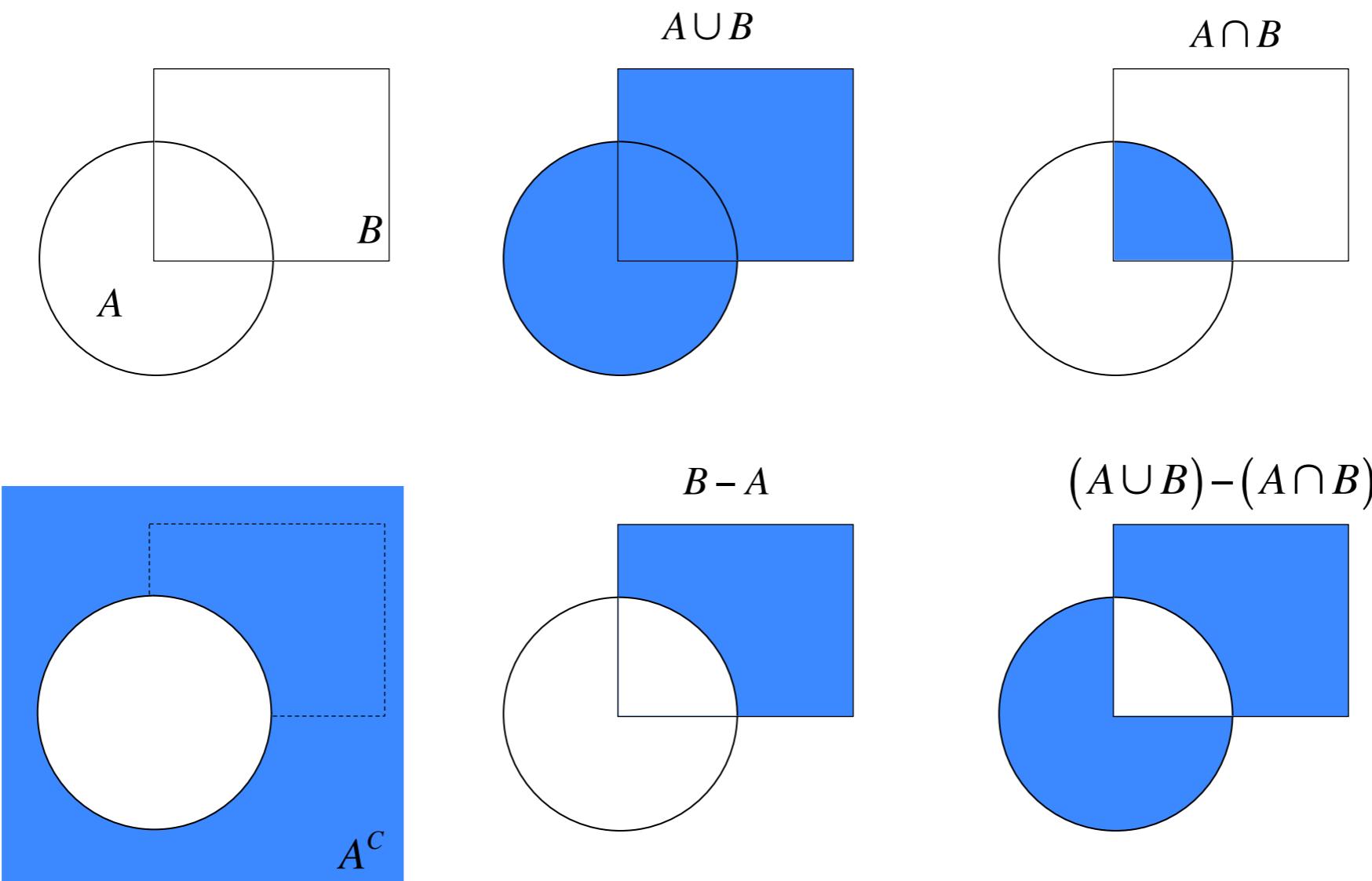
PROCESAMIENTO MORFOLÓGICO

Miguel Carrasco  
[miguel.carrasco@uai.cl](mailto:miguel.carrasco@uai.cl)  
1er Semestre 2020

- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y cierre
  - Algoritmos básicos

## ▶ Teoría de conjuntos

- Repasemos brevemente algunos conceptos básicos de teoría de conjunto.  
Supongamos que tenemos dos objetos A y B.



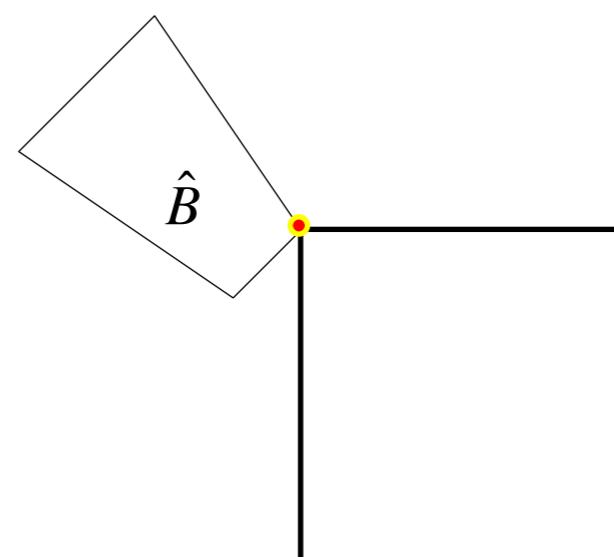
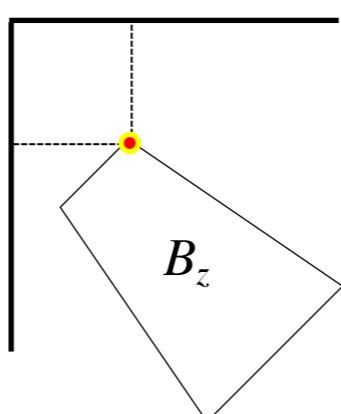
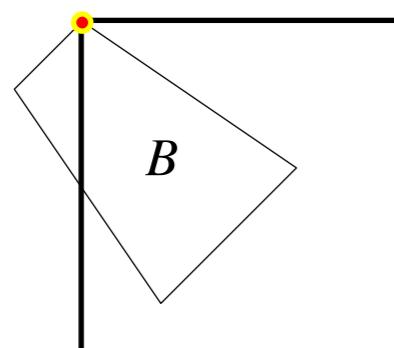
## ▶ Teoría de conjuntos

- Repasemos brevemente algunos conceptos básicos de teoría de conjunto.  
Supongamos que tenemos dos objetos A y B.
- Traslación de A por  $z$  define como

$$A_z = \{x \mid x = a + z, \quad a \in A\}$$

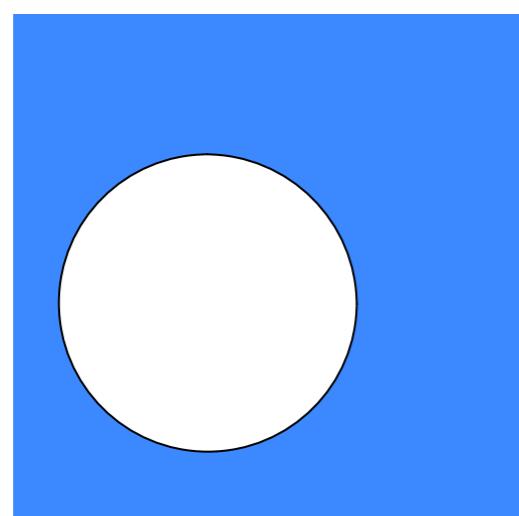
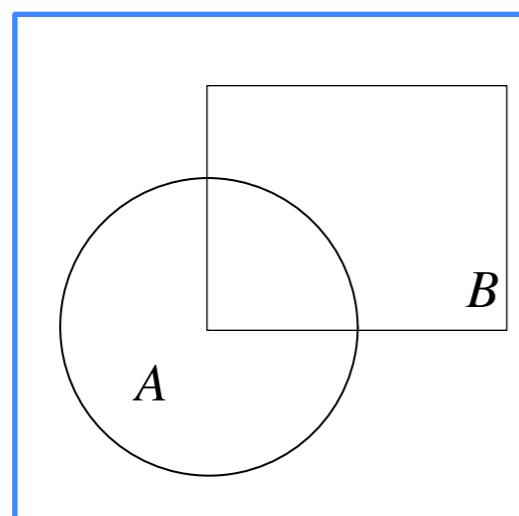
- La reflexión de B se define como

$$\hat{B} = \{x \mid x = -b, \quad b \in B\}$$

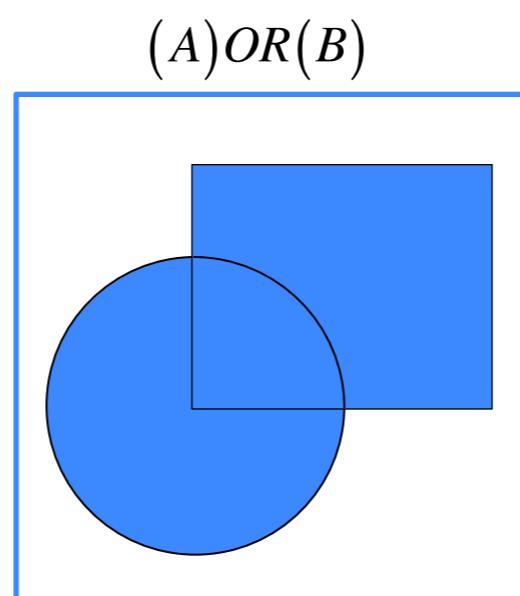


## ▶ Operadores lógicos

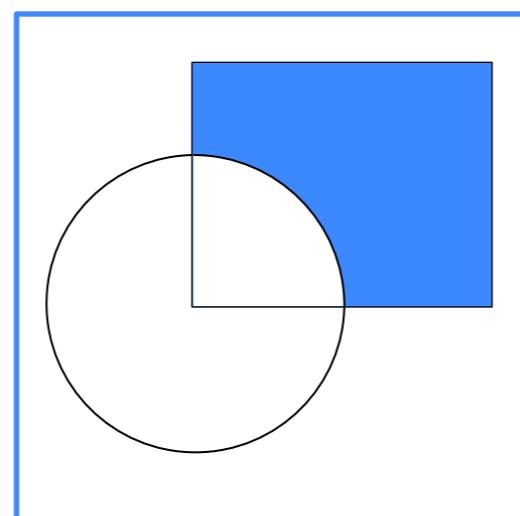
- Veamos los mismos operadores anteriores pero con operadores lógicos



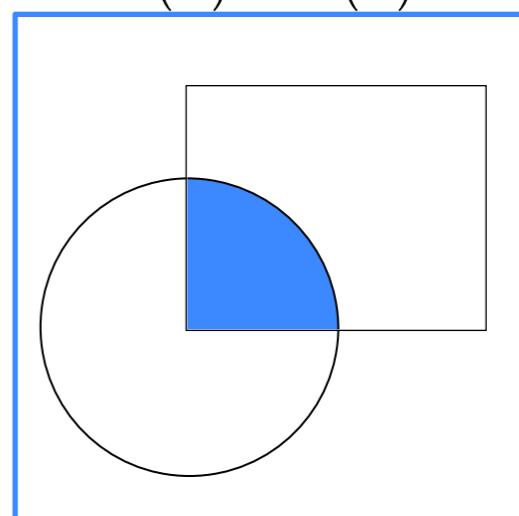
$NOT(A)$



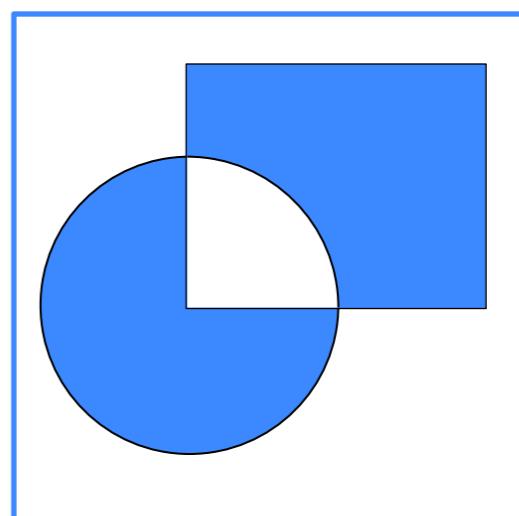
$(A)OR(B)$



$(NOT(A))AND(B)$



$(A)AND(B)$



$(A)XOR(B)$

- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y clausura
  - Algoritmos básicos



## ► Matemática morfológica

- Es el estudio de técnicas de análisis de señal basada en la teoría de conjuntos. Fue propuesto por Matheron y Serra en los años sesenta.
- Los operadores morfológicos son muy útiles ya que permiten realizar funciones complejas a partir de la combinación de operadores simples. Comúnmente son empleados para la reducción del ruido, realce de la imagen, detección de bordes, segmentación, análisis de textura, etc.
- Veamos un ejemplo.



Imagen original

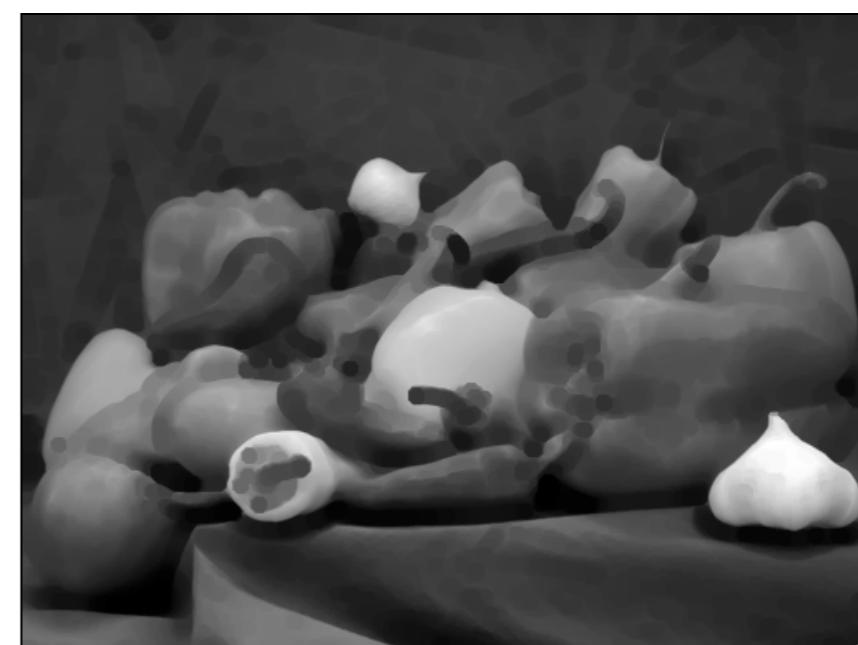
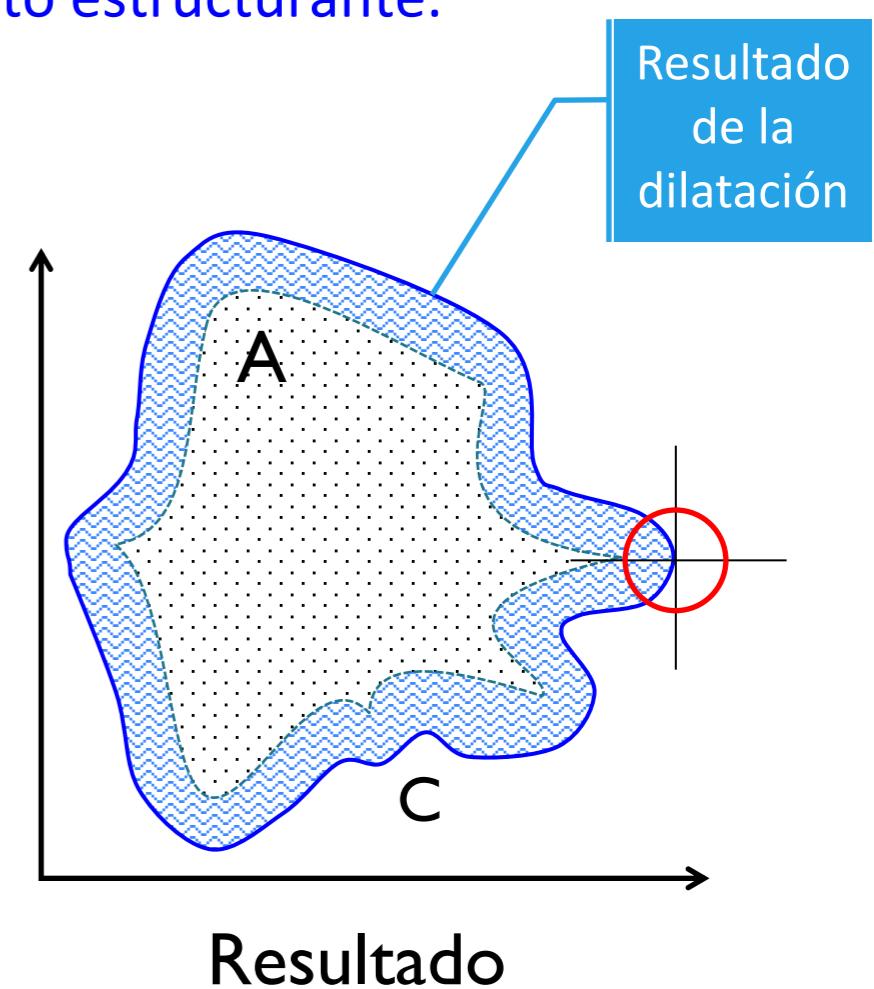
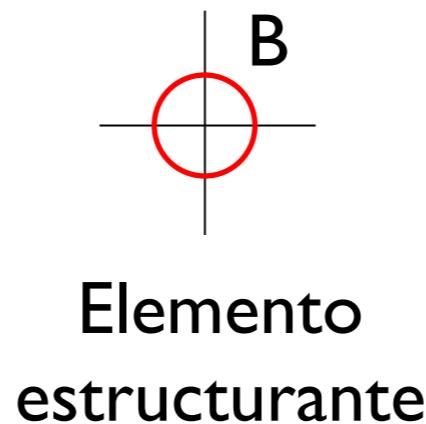
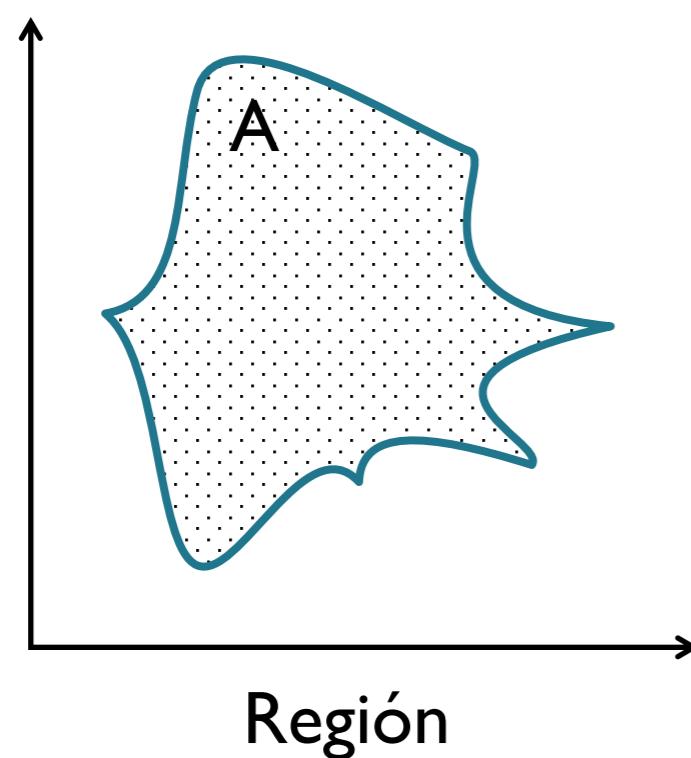


Imagen erosionada

## ► Dilatación

- Uno de los operadores más importantes en la matemática morfológica es la dilatación. Desde un punto de vista geométrico, la dilatación consiste en el crecimiento de la **región** (o conjunto) por un **elemento estructurante**.



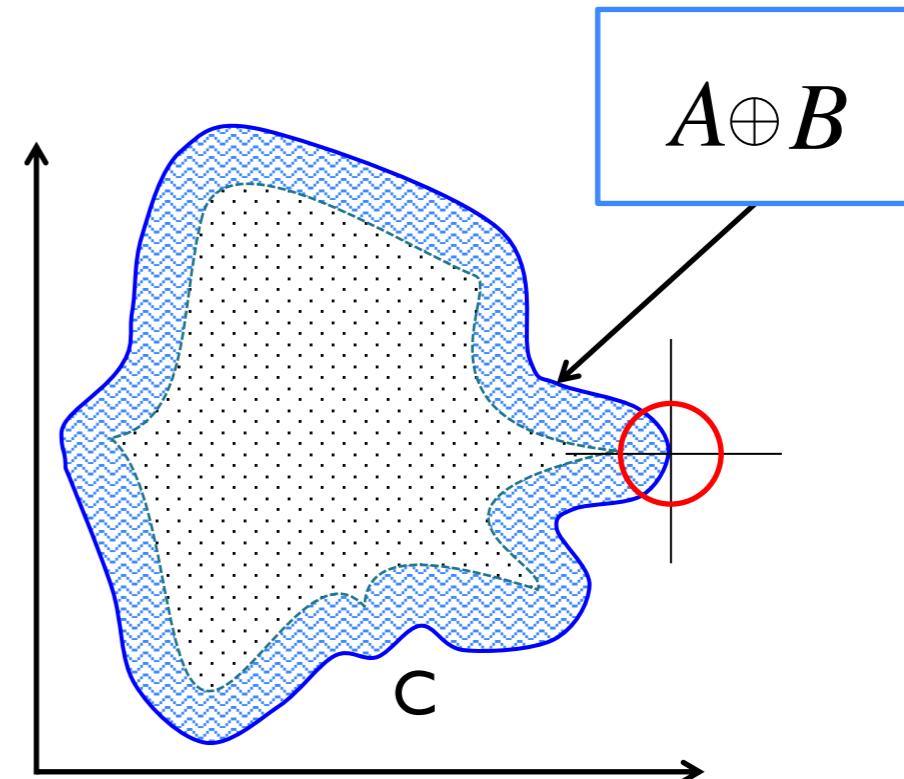
Región

Elemento estructurante

Resultado

## Dilatación

- Uno de los operadores más importantes en la matemática morfológica es la dilatación. Desde un punto de visto geométrico, la dilatación consiste en el crecimiento de la **región** (o conjunto) por un **elemento estructurante**.



Resultado

```
img = cv2.imread('j.png',0)
```

```
kernel = cv2.getStructuringElement(cv2.MORPH_CROSS,(3,3))
```

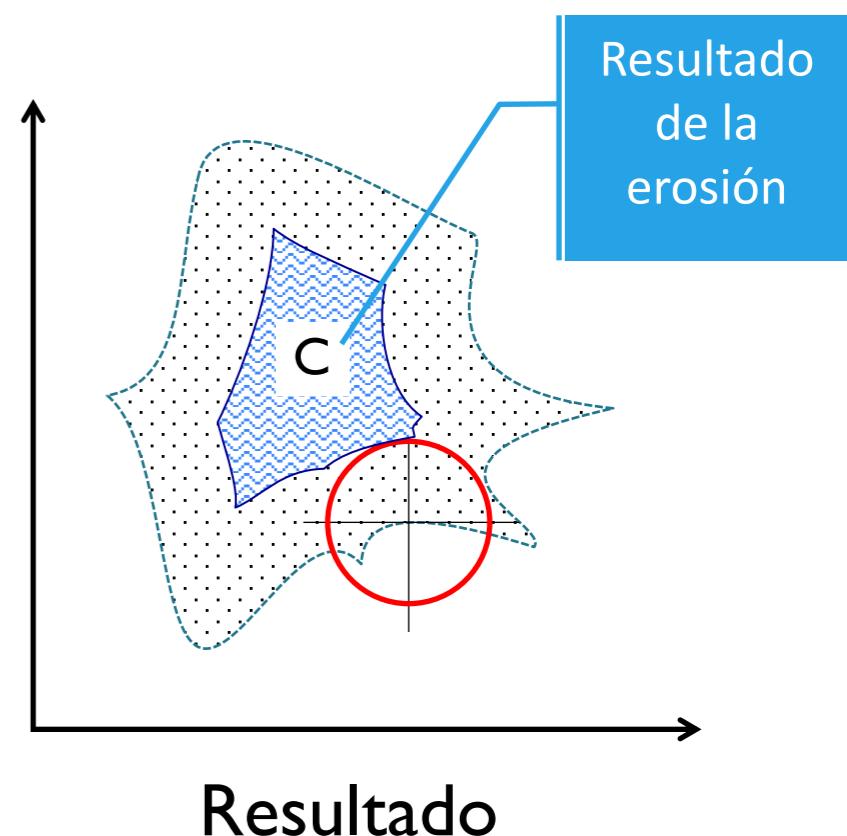
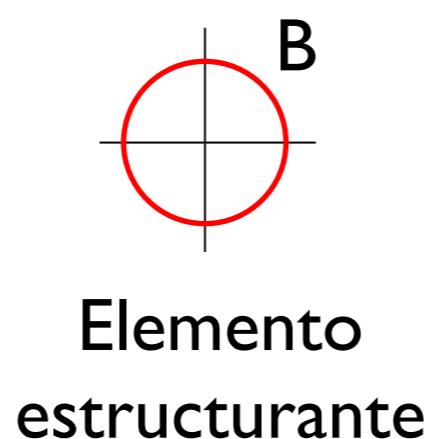
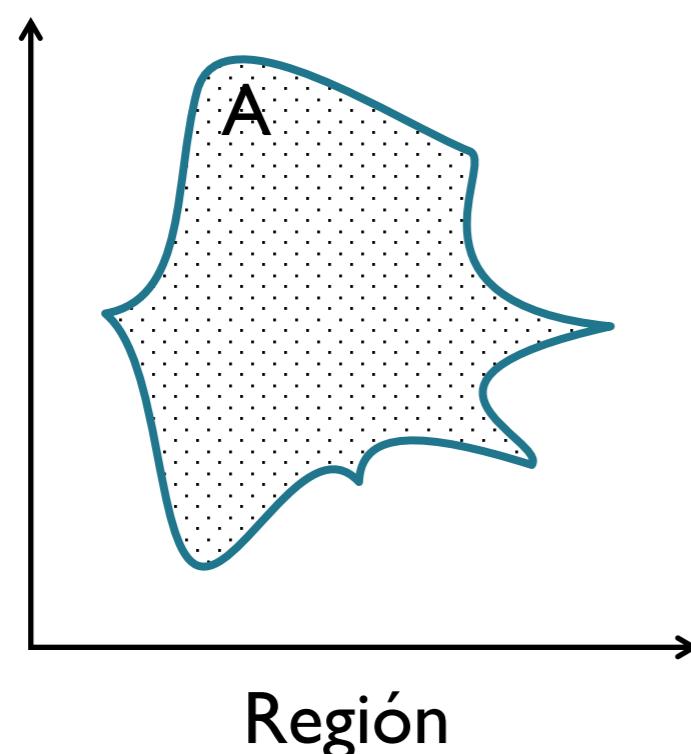
```
dilate = cv2.dilate(img, kernel)
```

El elemento estructurante se define con getStructuringElement

La función imdilate realiza la dilatación

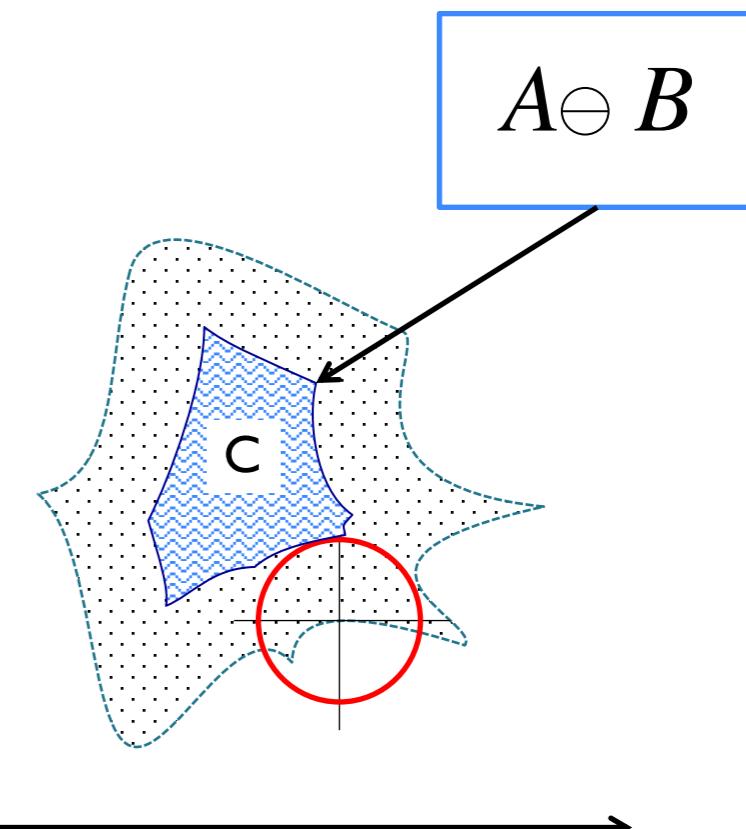
## ▶ Erosión

- El segundo operador más importantes en la matemática morfológica es la erosión. Desde un punto de vista geométrico, la erosión consiste en la reducción de parte de la **región** (o conjunto) por un **elemento estructurante**.



## ▶ Erosión

- El segundo operador más importantes en la matemática morfológica es la erosión. Desde un punto de visto geométrico, la erosión consiste en la reducción de parte de la **región** (o conjunto) por un **elemento estructurante**.



Resultado

```
img = cv2.imread('j.png',0)  
  
kernel = cv2.getStructuringElement(cv2.MORPH_CROSS,(3,3))  
  
dilate = cv2.erode(img, kernel)
```

El elemento estructurante se define con getStructuringElement

La función imdilate realiza la dilatación

- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y clausura
  - Algoritmos básicos

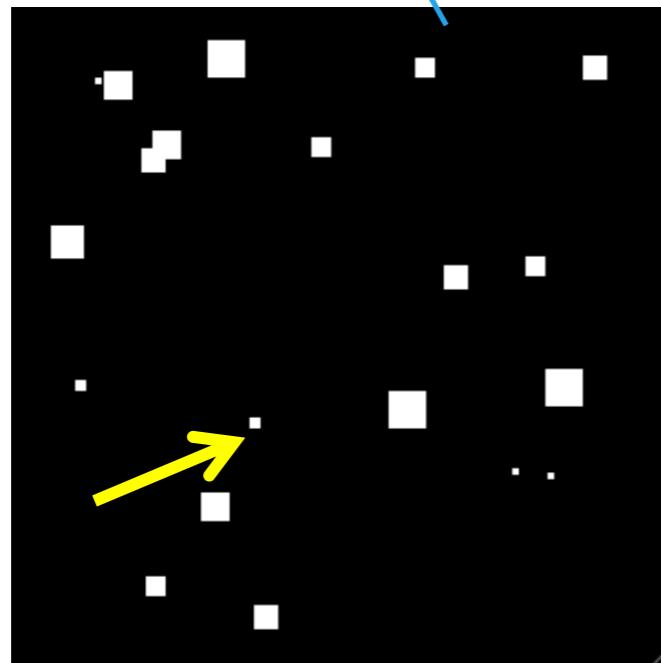


## ► Apertura

- La apertura es una función que tiene dos pasos. Primero aplicamos erosión y luego a dicho resultado una dilatación. El efecto más relevante es la reducción del ruido. Matemáticamente se representa con la siguiente expresión:

Muchas regiones desaparecen al aplicar la erosión

$$A \circ B = (A \ominus B) \oplus B$$



$$B = \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

$A$

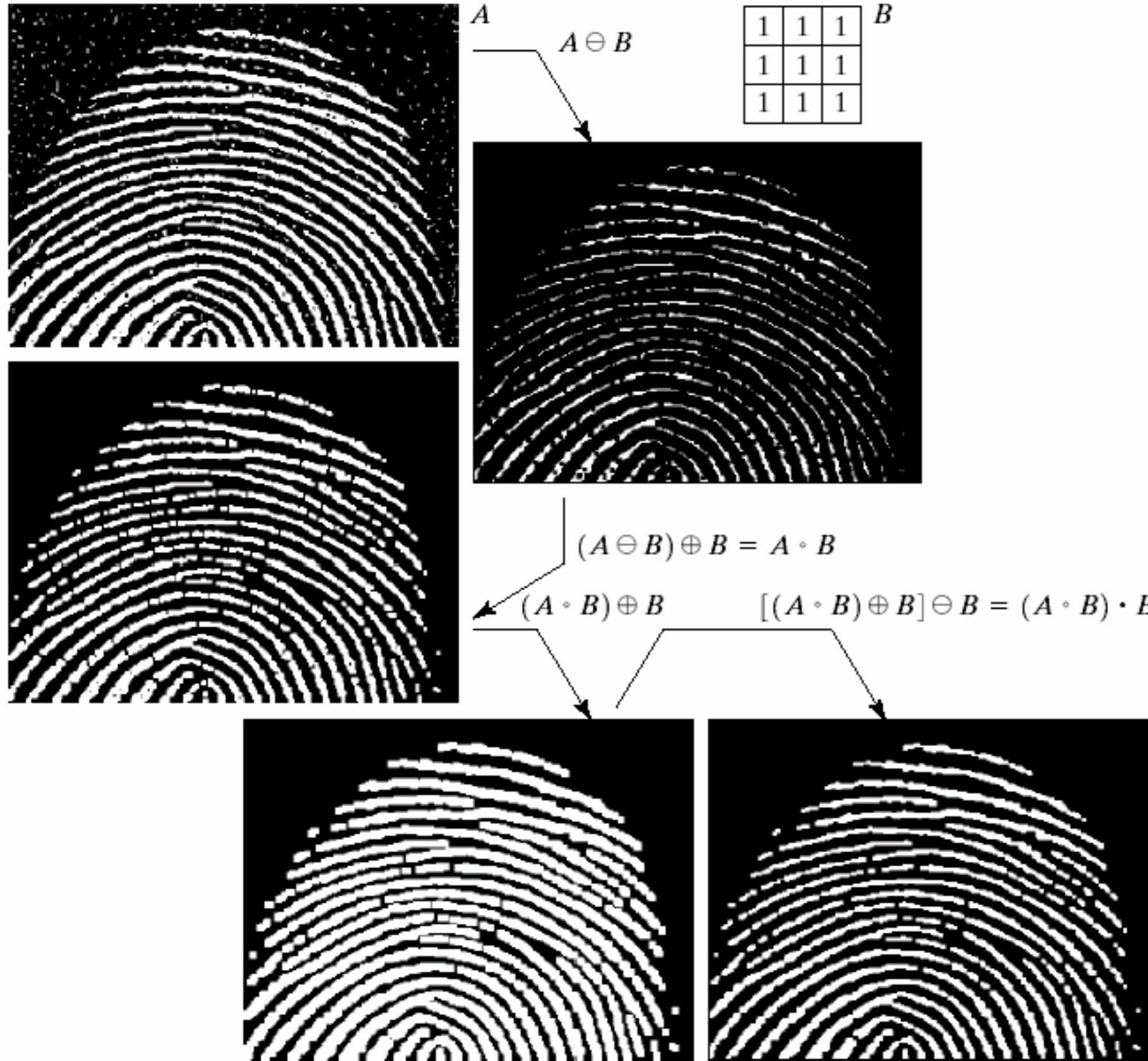


$(A \ominus B)$

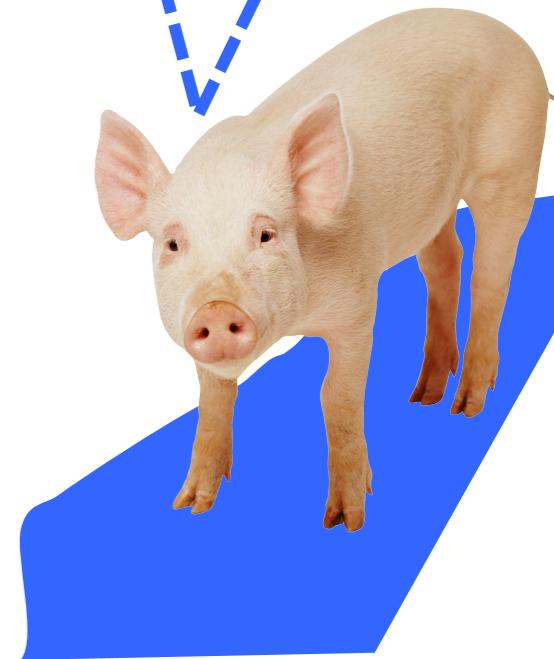


$((A \ominus B) \oplus B)$

# Procesamiento morfológico



Que fácil podemos  
mejorar una  
imagen



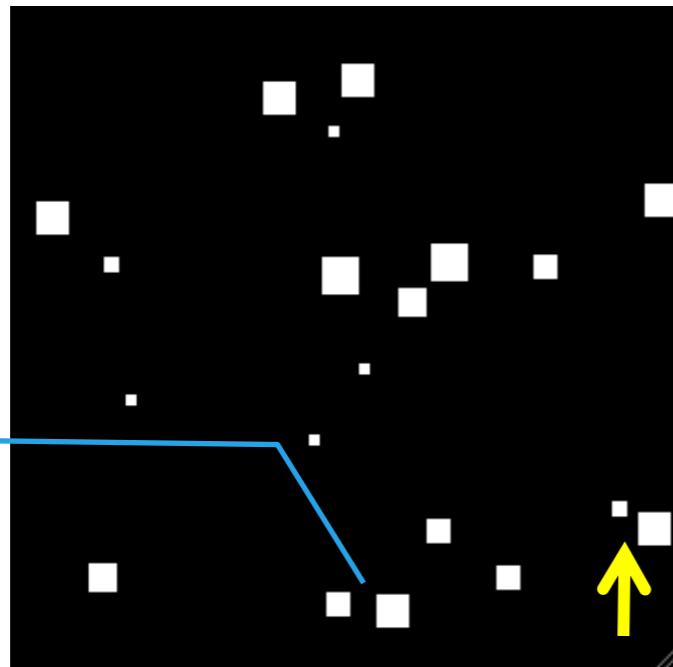
## ▶ Clasura

- La clausura es una función que tiene dos pasos. Primero aplicamos una dilatación y luego a dicho resultado una erosión. Esta técnica permite unir regiones que están muy cercanas. Matemáticamente se representa con la siguiente expresión:

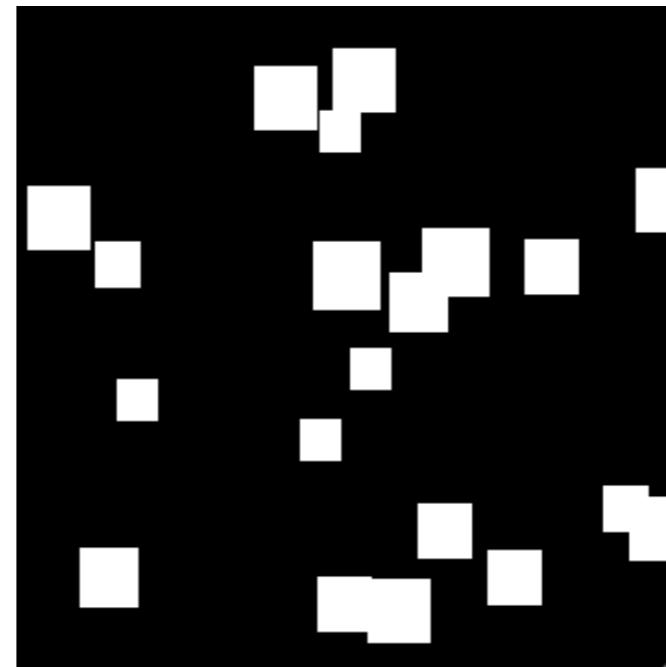
$$B = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

$$A \bullet B = (A \oplus B) \ominus B$$

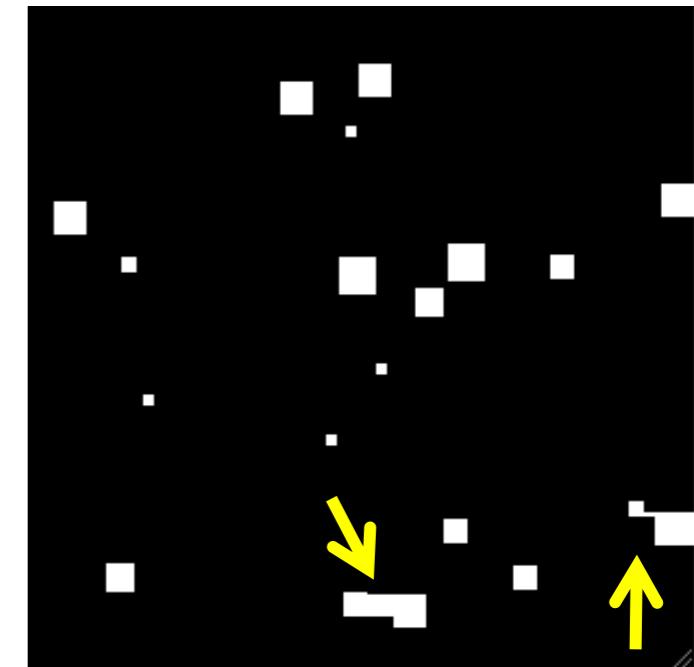
Algunas regiones se unen formando puentes



$A$



$(A \oplus B)$



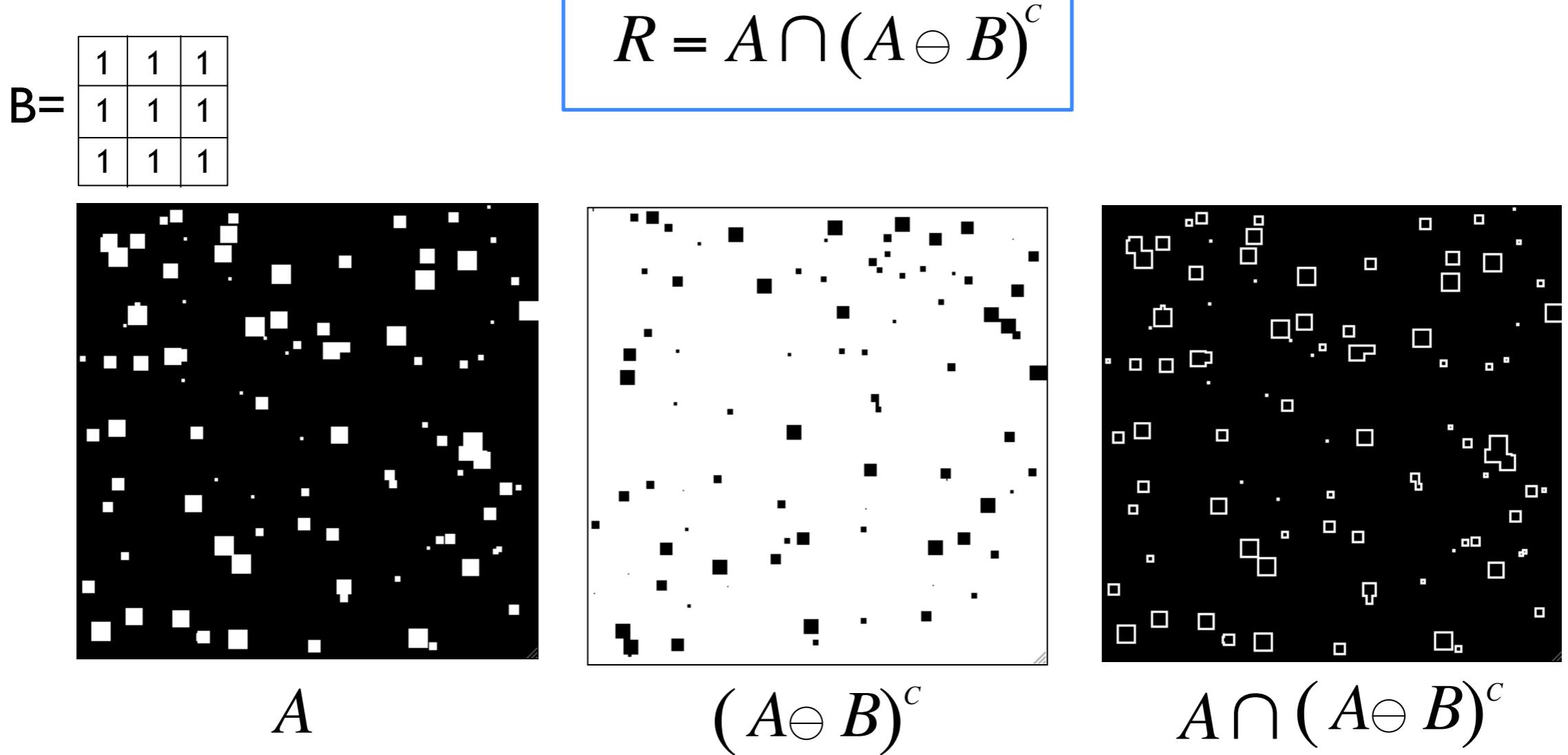
$((A \oplus B) \ominus B)$

- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y clausura
  - Algoritmos básicos
    - Detector de bordes
    - Relleno de regiones
    - Componentes conectadas
    - Transformación Hit-or-Miss



## ▶ Detector de bordes (simple)

- Una forma simple para detectar los bordes consiste en intersectar la imagen original con una versión erosionada. Matemáticamente equivale a decir:

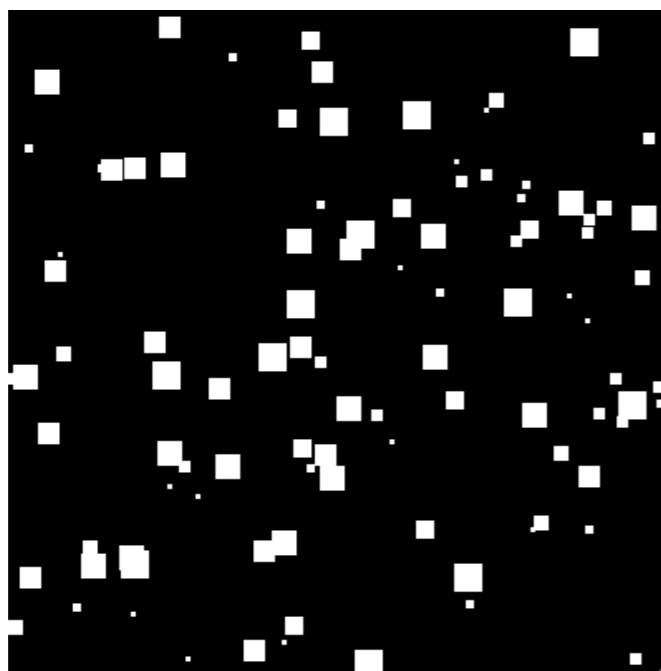


- ▶ Detector de bordes (elaborado)
  - Lamentablemente el resultado anterior refleja el ruido. Para reducir el ruido primero aplicamos el **operador apertura**. Luego aplicamos el mismo proceso anterior

$$B = \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

$$R = A \circ B = (A \ominus B) \oplus B$$

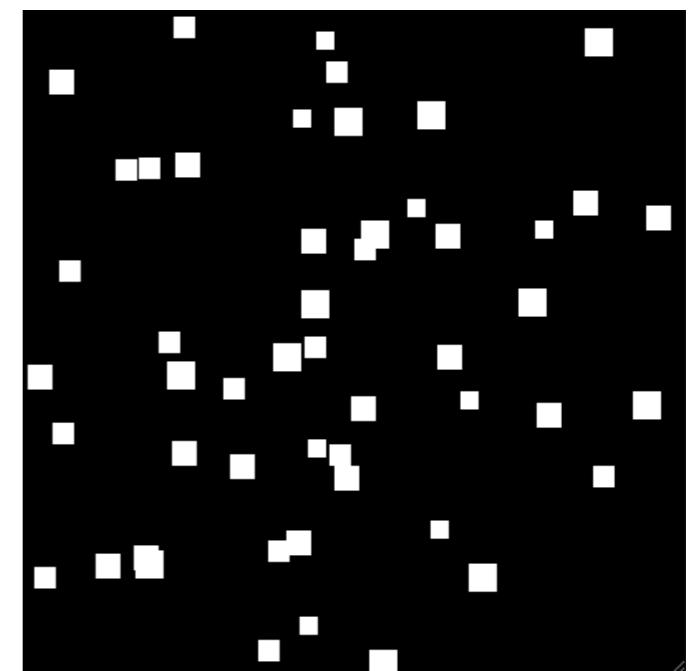
1ra Etapa



$A$



$(A \ominus B)$



$(A \ominus B) \oplus B$

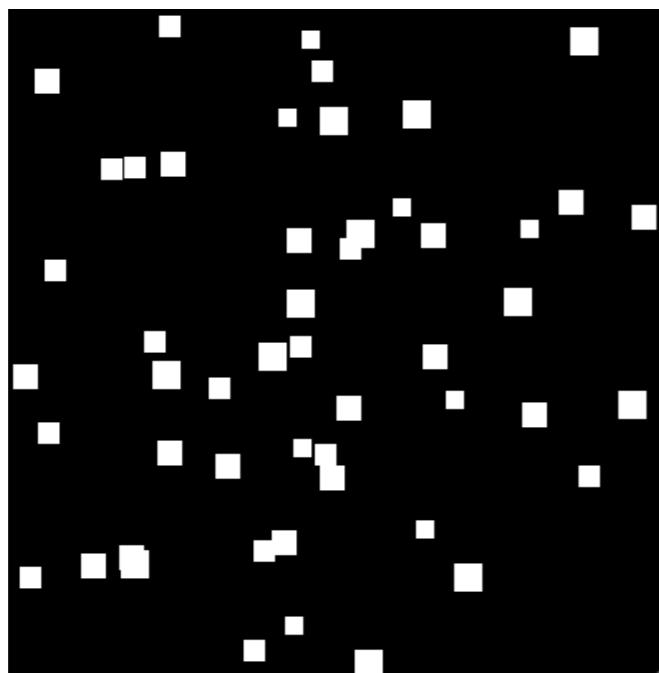
- ▶ Detector de bordes (elaborado)
  - Lamentablemente el resultado anterior refleja el ruido. Para reducir el ruido primero aplicamos el **operador apertura**. Luego aplicamos el mismo proceso anterior

$B =$

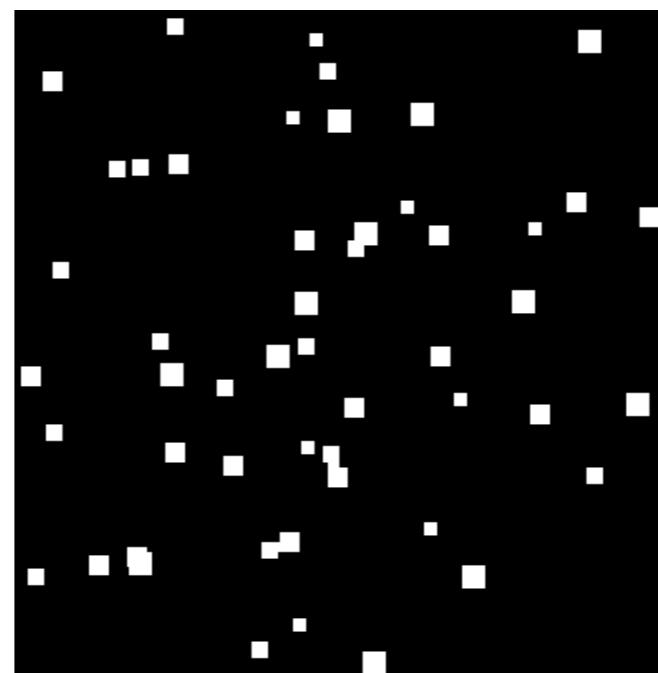
1	1	1
1	1	1
1	1	1

$$P = R \cap (R \ominus B)^c$$

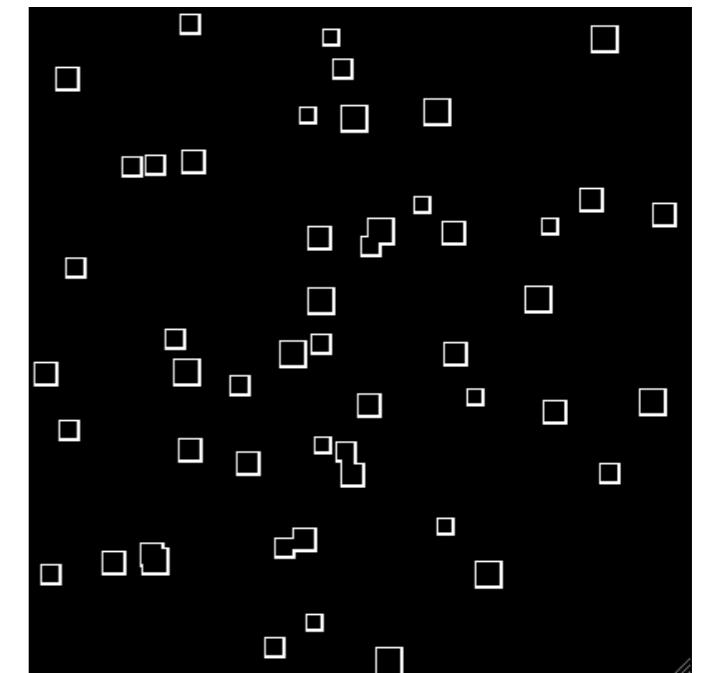
2da Etapa



$R$



$(R \ominus B)$



$R \cap (R \ominus B)^c$

- ▶ Detector de bordes (elaborado)

- En resumen, y reemplazando las operaciones.

$$P = A \circ B \cap (A \circ B \ominus B)^c$$

1

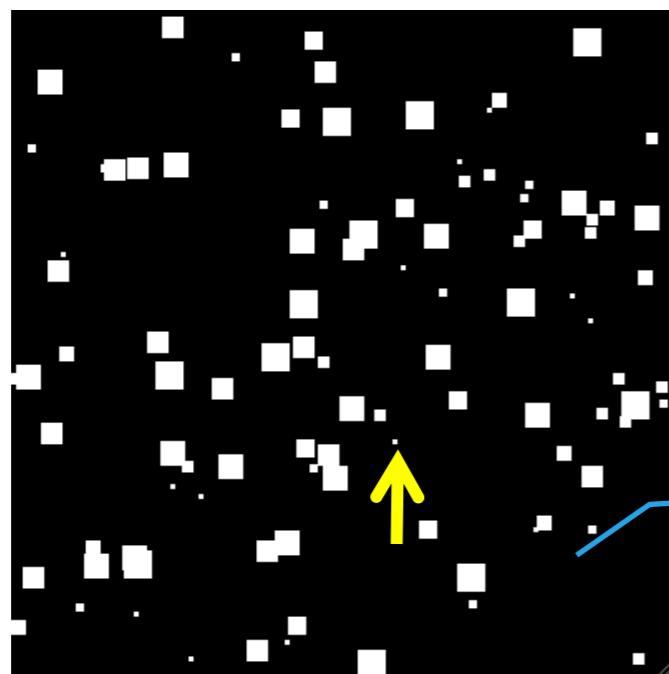
$$R = A \circ B = (A \ominus B) \oplus B$$

2

$$P = R \cap (R \ominus B)^c$$

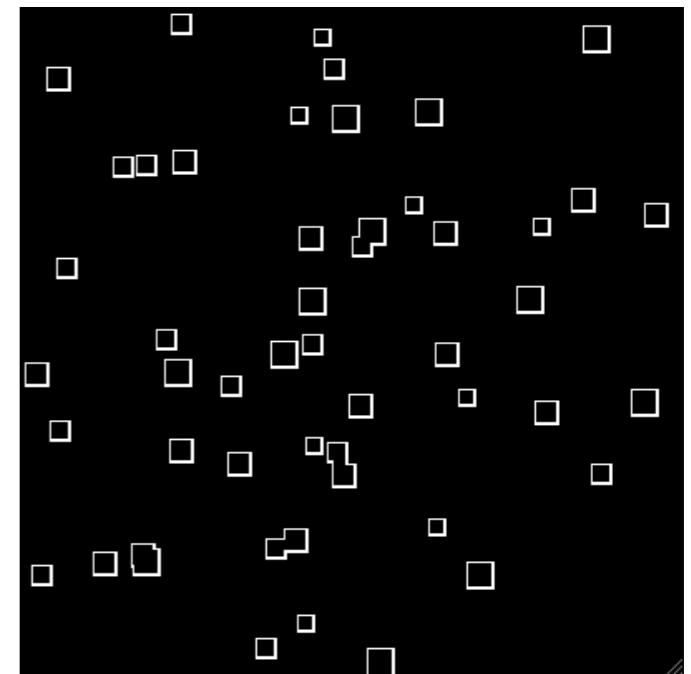
3

$$P = ((A \ominus B) \oplus B) \cap (((A \ominus B) \oplus B) \ominus B)^c$$



*A*

Como se observa,  
gran parte del  
ruido se redujo



*P*

- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y clausura
  - Algoritmos básicos
    - Detector de bordes
    - Relleno de regiones
    - Componentes conectadas
    - Transformación Hit-or-Miss

- ▶ Crecimiento de regiones
  - El crecimiento de regiones permite llenar regiones conectadas o cerradas

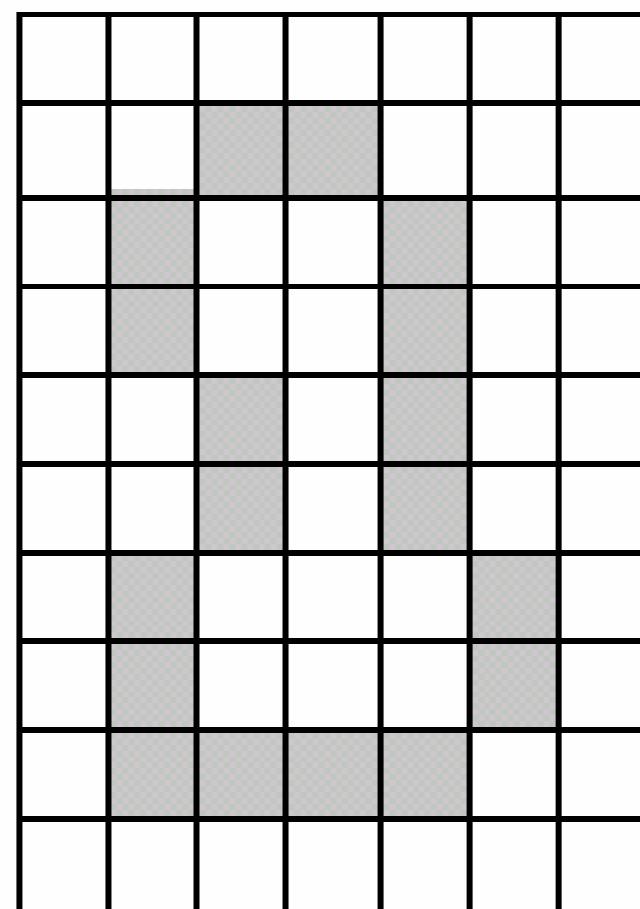
$B =$

0	1	0
1	1	1
0	1	0

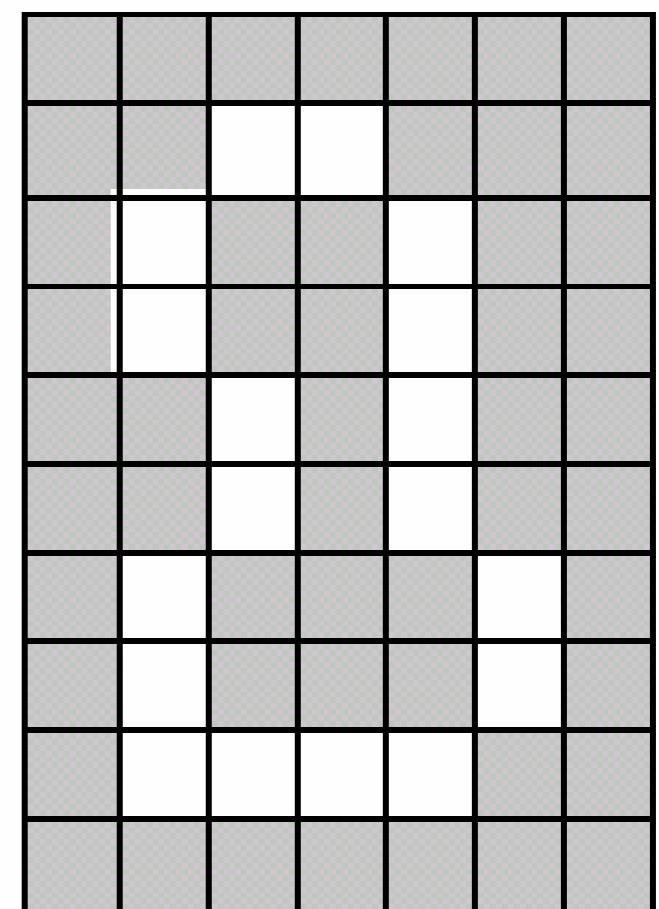
$$X_{i+1} = (X_i \oplus B) \cap A^c$$

Elemento  
estructurante

Imagen



$A$



$A^c$

## ▶ Crecimiento de regiones

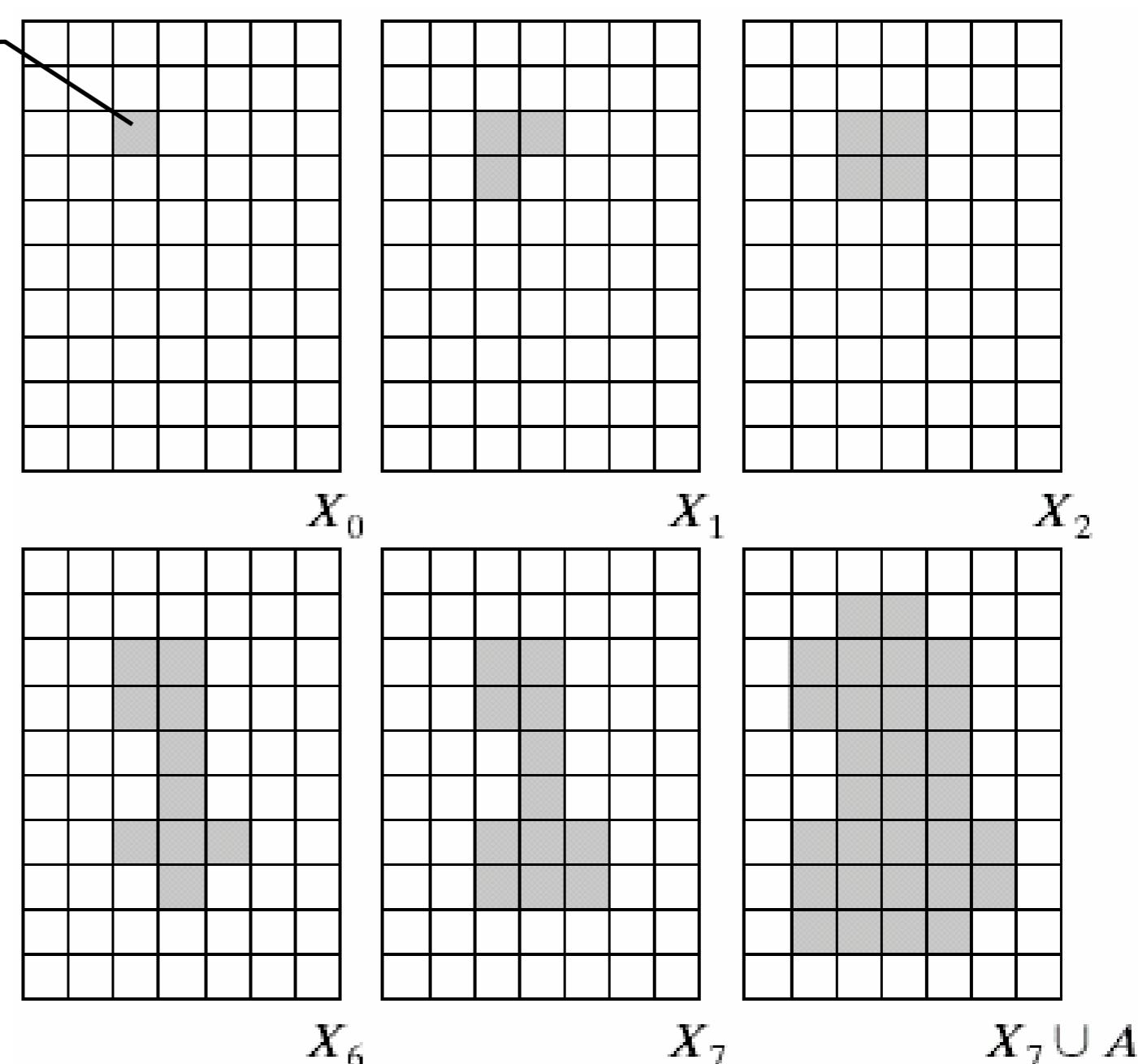
- El crecimiento de regiones permite llenar regiones conectadas o cerradas

$$B = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

Importante: El algoritmo comienza con un pixel del borde

$$X_{i+1} = (X_i \oplus B) \cap A^C$$

Imagen  
Elemento estructurante



## ▶ Crecimiento de regiones

- El crecimiento de regiones permite llenar regiones conectadas o cerradas

$$B = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

$$X_{i+1} = (X_i \oplus B) \cap A^c$$

```

import cv2
import numpy as np

#imagen original
img = cv2.imread('test.png')
A = img[:, :, 0]
AC = cv2.bitwise_not(A).astype('uint8')
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5, 5))

#creacion imagen
m, n = AC.shape
M = np.zeros([m, n], dtype=np.uint8)
BIN = np.zeros([m, n], dtype=np.uint8)

#buscamos un valor donde img sea blanco
ind = np.where(AC == 255)
BIN[ind[0], ind[1]] = 255

#inicializamos un valor en 255
M[ind[0][0], ind[1][0]] = 255

for i in range(0, 300):
    M = cv2.bitwise_and(cv2.dilate(M, kernel), BIN)

K = cv2.bitwise_not(M).astype('uint8')
R = cv2.bitwise_and(K, BIN)

cv2.imshow('Imagen Binaria', AC)
cv2.imshow('componentes_conectadas', R)
cv2.waitKey(0)

```



- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y clausura
  - Algoritmos básicos
    - Detector de bordes
    - Relleno de regiones
    - Componentes conectadas
    - Transformación Hit-or-Miss



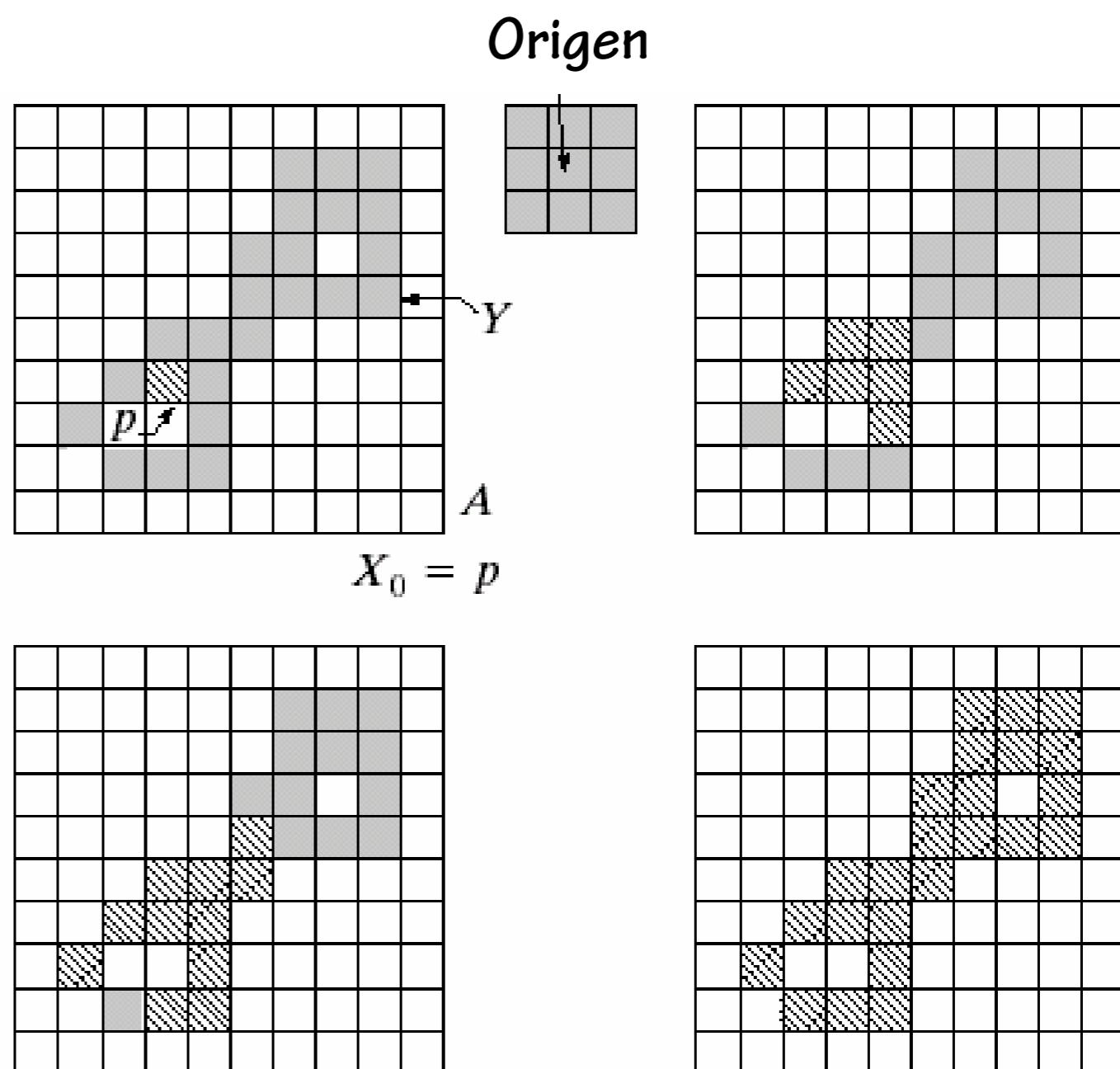
## ▶ Extracción de componentes

- La extracción de componentes conectados permite determinar los píxeles que están conectados de una región

$$B = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

$$X_{i+1} = (X_i \oplus B) \cap A$$

Imagen  
Elemento  
estructurante



## ▶ Crecimiento de regiones

- El crecimiento de regiones permite llenar regiones conectadas o cerradas

$$B = \begin{array}{|c|c|c|} \hline 0 & 1 & 0 \\ \hline 1 & 1 & 1 \\ \hline 0 & 1 & 0 \\ \hline \end{array}$$

$$X_{i+1} = (X_i \oplus B) \cap A$$

Imagen  
Elemento  
estructurante

```
import cv2
import numpy as np

#imagen original
img = cv2.imread('test.png')
A = img[:, :, 0].astype('uint8')
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (5,5))

#creacion imagen
m,n = A.shape
M = np.zeros([m,n], dtype=np.uint8)
BIN= np.zeros([m,n], dtype=np.uint8)

#buscamos un valor donde img sea blanco
ind= np.where(A==255)
BIN[ind[0], ind[1]] = 255

#inicializamos un valor en 255
M[ind[0][0], ind[1][0]]=255

for i in range(0,300):
    M= cv2.bitwise_and(cv2.dilate(M, kernel), BIN)

cv2.imshow('Imagen Binaria',A)
cv2.imshow('Componentes conectadas',M)

cv2.waitKey(0)
```

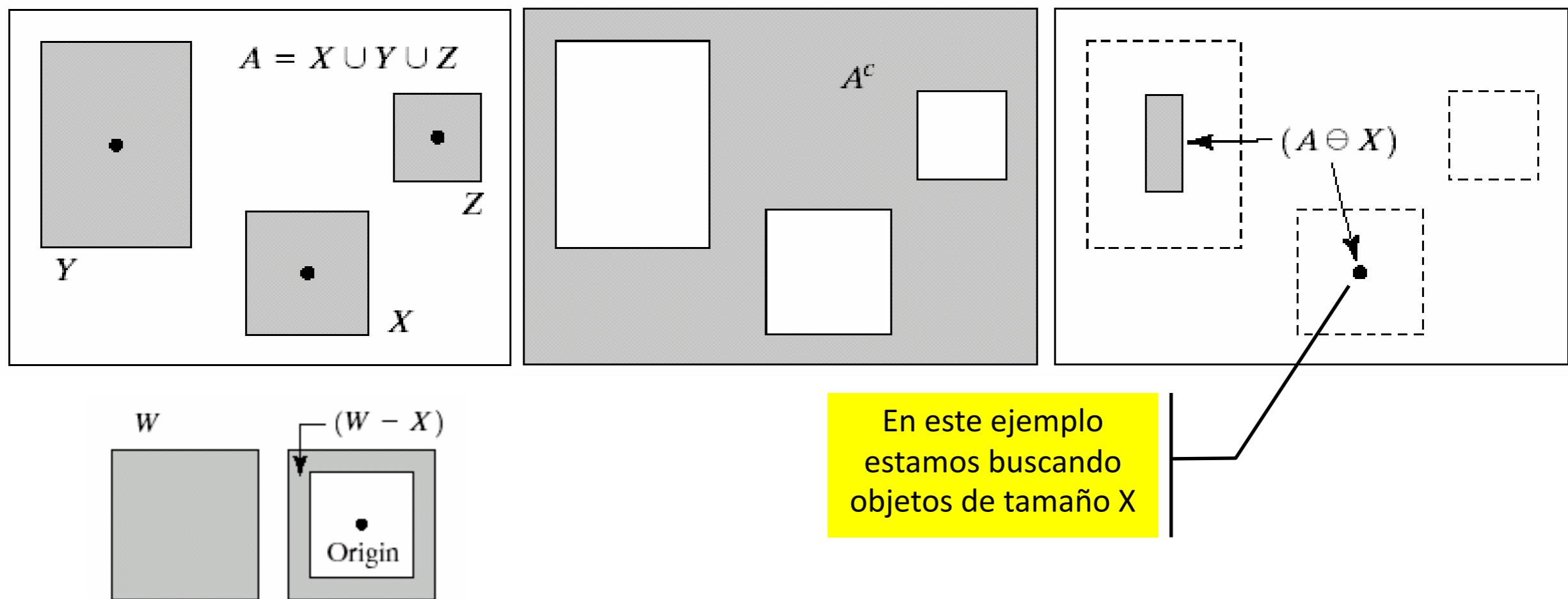
- Procesamiento morfológico de imágenes
  - Teoría de conjuntos y operadores lógicos
  - Dilatación y erosión
  - Apertura y clausura
  - Algoritmos básicos
    - Detector de bordes
    - Relleno de regiones
    - Componentes conectadas
    - Transformación Hit-or-Miss



## ▶ Transformación Hit-or-Miss

- Nos permite detectar las regiones según el tamaño que nosotros definamos.

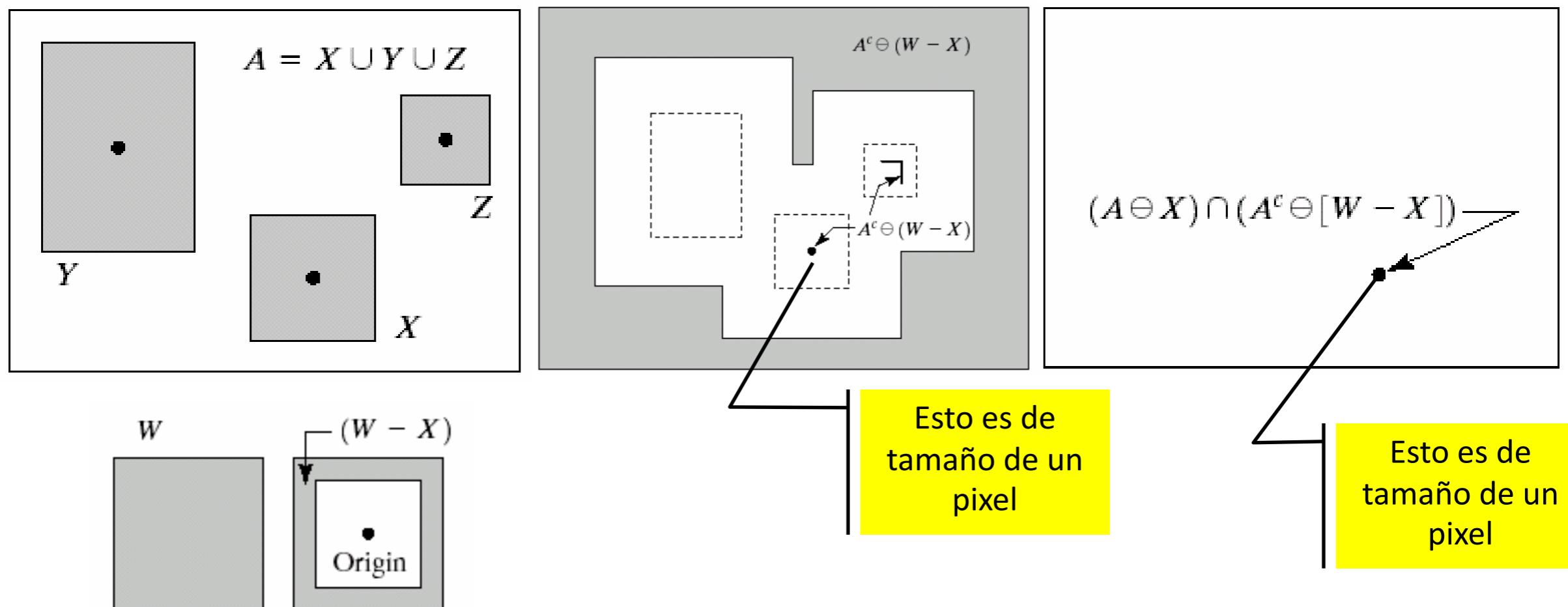
$$P = (A \ominus X) \cap (A^c \ominus (W - X))$$



## ▶ Transformación Hit-or-Miss

- Nos permite detectar las regiones según el tamaño que nosotros definamos.

$$P = (A \ominus X) \cap (A^c \ominus (W - X))$$



## ▶ Transformación Hit-or-Miss

- Nos permite detectar las regiones según el tamaño que nosotros definamos.

$$P = (A \ominus X) \cap (A^c \ominus (W - X))$$

bw1

bw2

Buscamos  
objetos de este  
tamaño

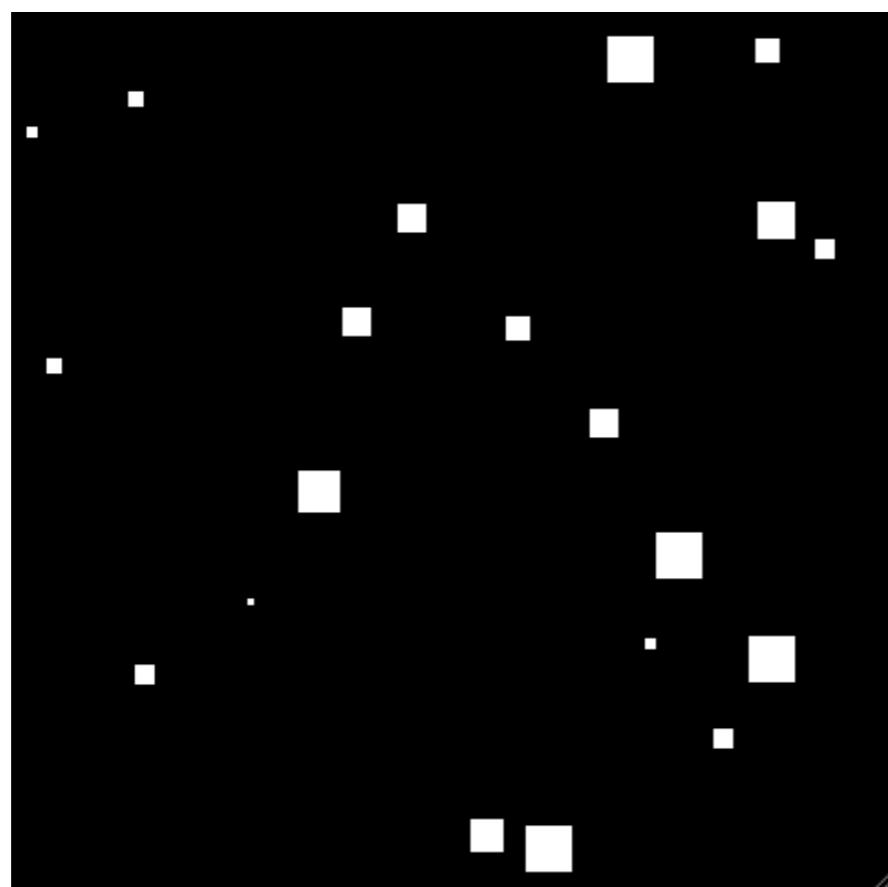
```
A = puntos_aleatorios(400, 80, 10).astype('uint8')  
  
N=9  
X = cv2.getStructuringElement(cv2.MORPH_RECT, (N-2,N-2))  
Z = cv2.getStructuringElement(cv2.MORPH_RECT, (N,N))  
Z[1:N-1, 1:N-1]=0
```

```
#Erosion  
bw1 = cv2.erode(A,X)  
bw2 = cv2.erode(cv2.bitwise_not(A),Z)  
  
HM =cv2.bitwise_and(bw1, bw2)  
cv2.imshow('Original',A)  
cv2.imshow('Hit Miss',HM)  
cv2.waitKey(0)
```

Aca obtenemos  
el resultado

- ▶ Transformación Hit-or-Miss
  - Nos permite detectar las regiones según el tamaño que nosotros definamos.

$$A \otimes B = (A \ominus X) \cap (A^c \ominus (W - X))$$



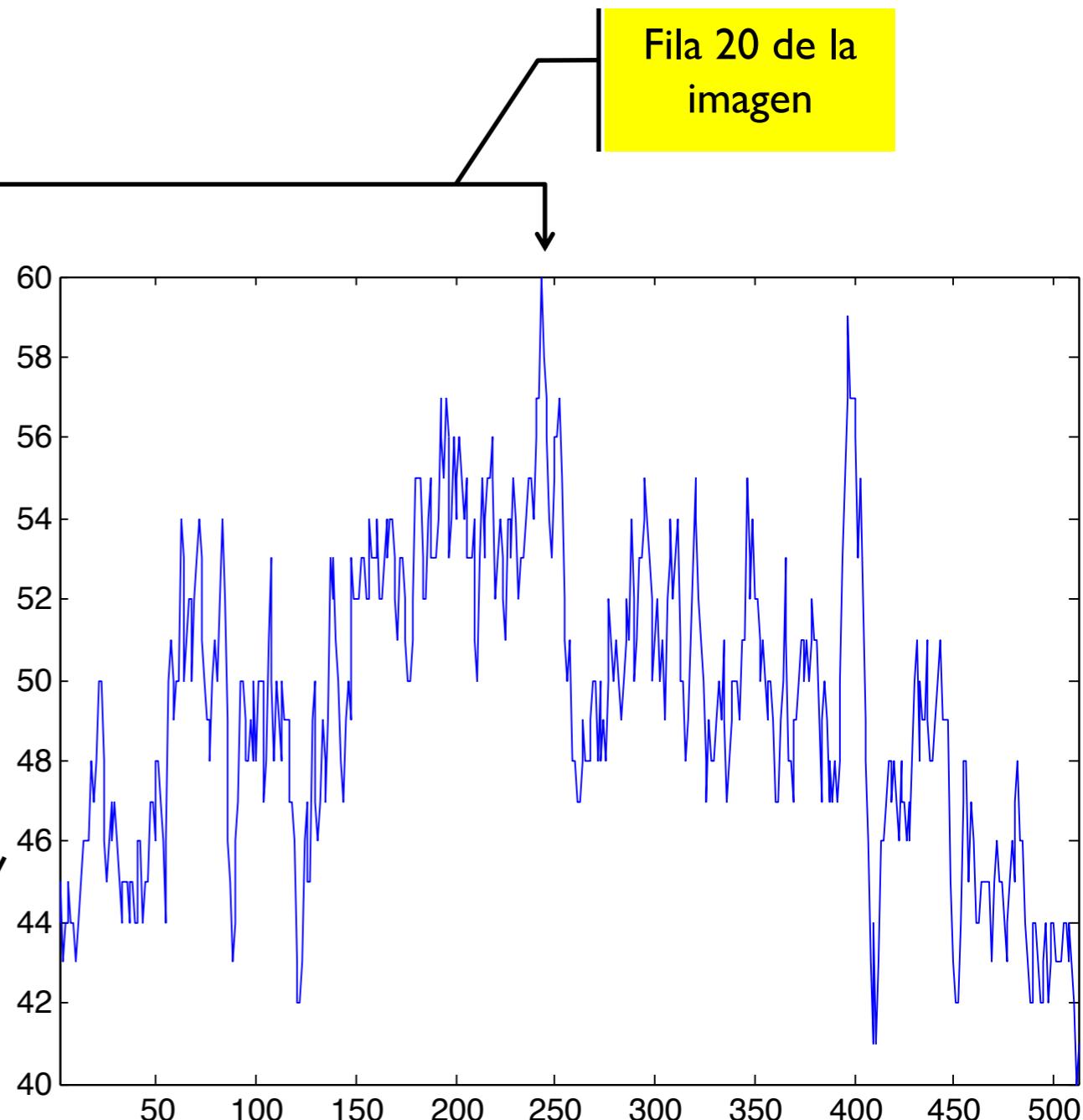
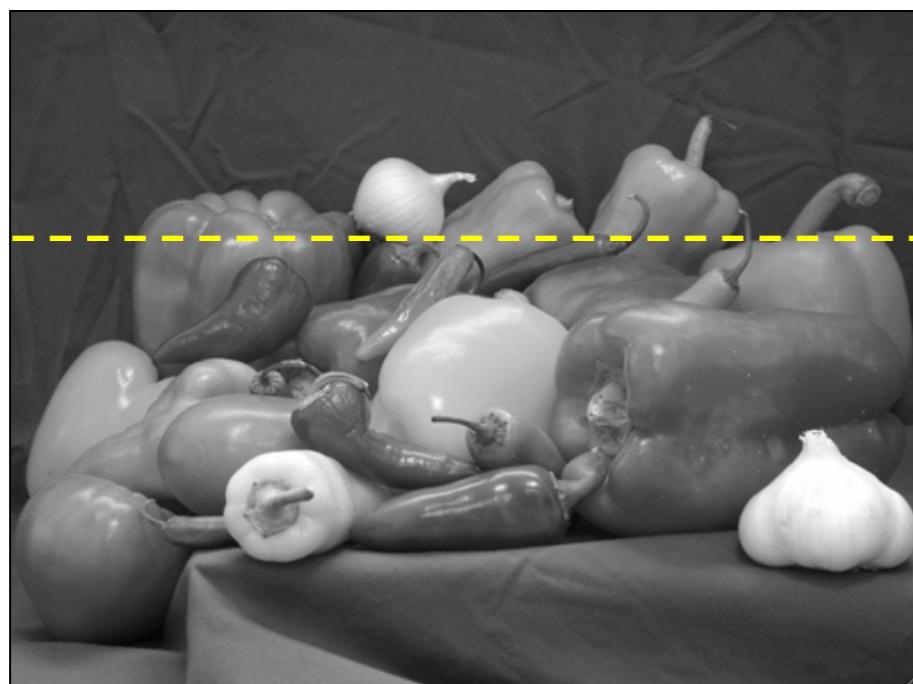
*A*



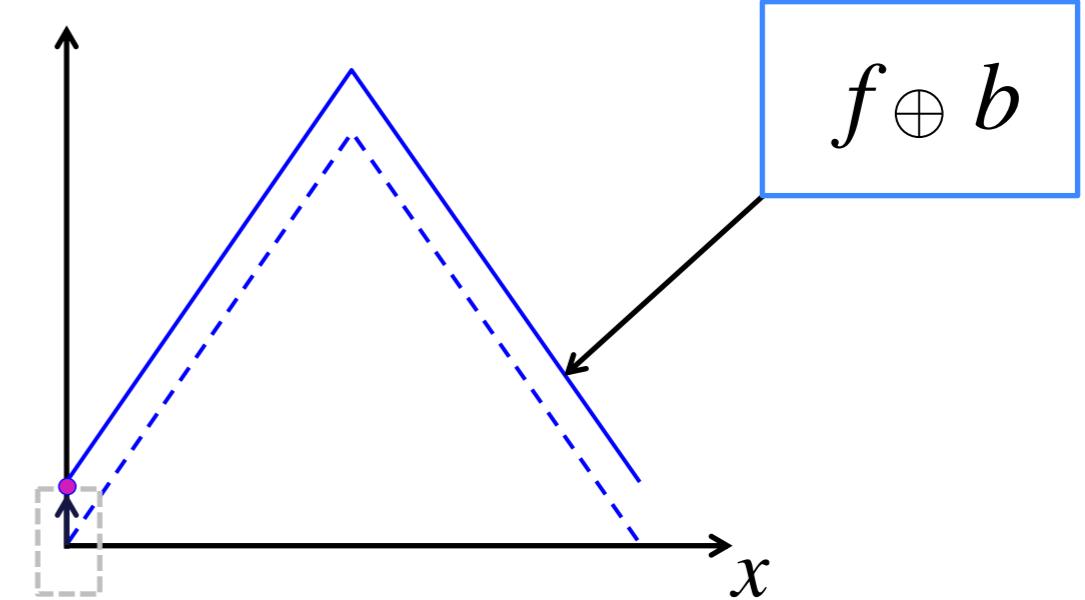
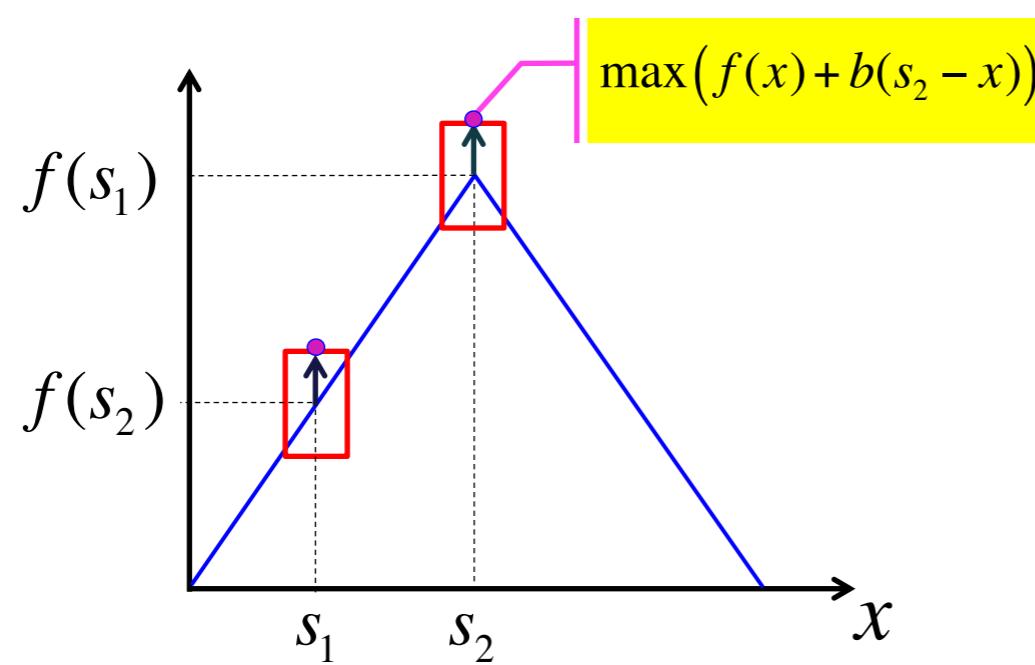
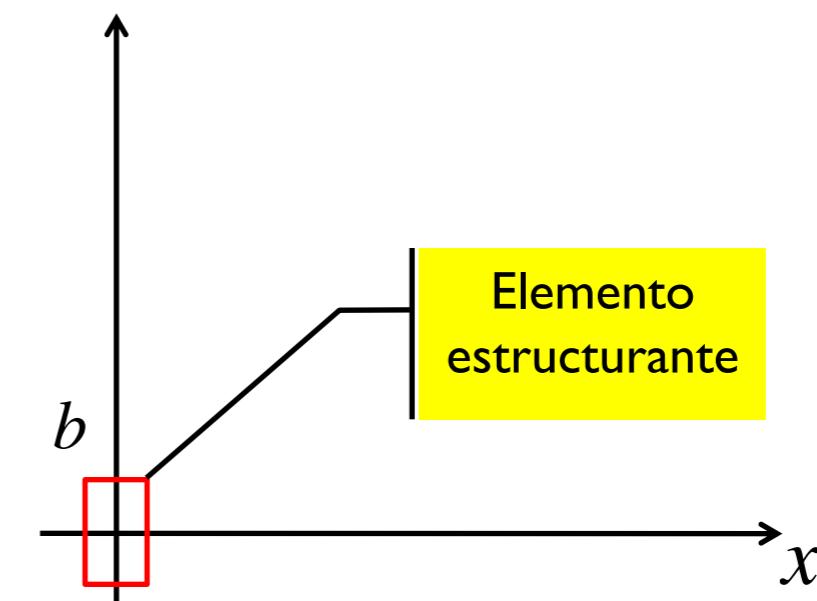
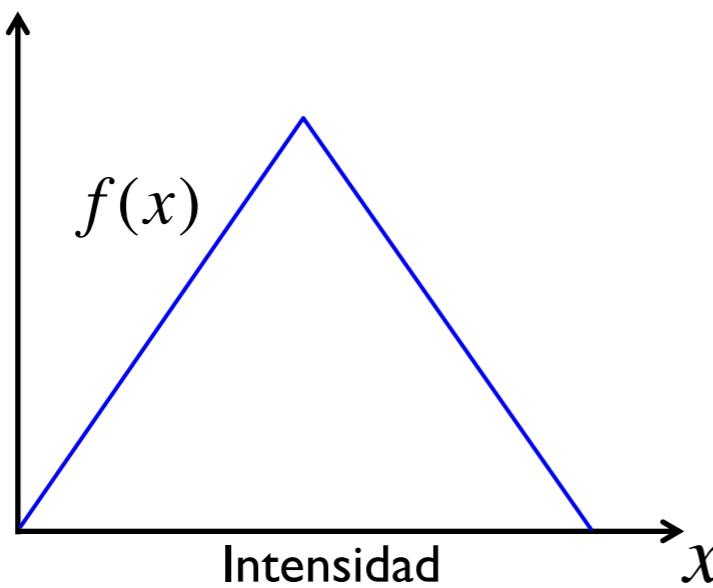
*P*

- Procesamiento morfológico de imágenes
  - Dilatación en escala de grises
  - Erosión en escala de grises
  - Comparación
  - Apertura y clausura
  - Suavización y gradiente morfológico
  - Transformación Top-Hat

- ▶ Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **dilatación** el proceso es el siguiente.



- Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **dilatación** el proceso es el siguiente.



- ▶ Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **dilatación** el proceso es el siguiente.

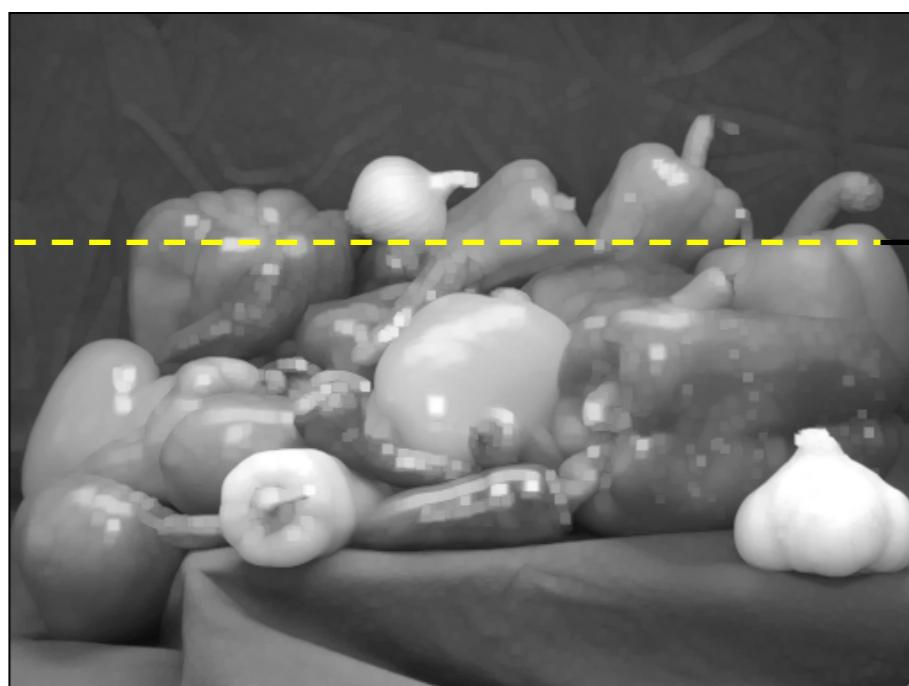
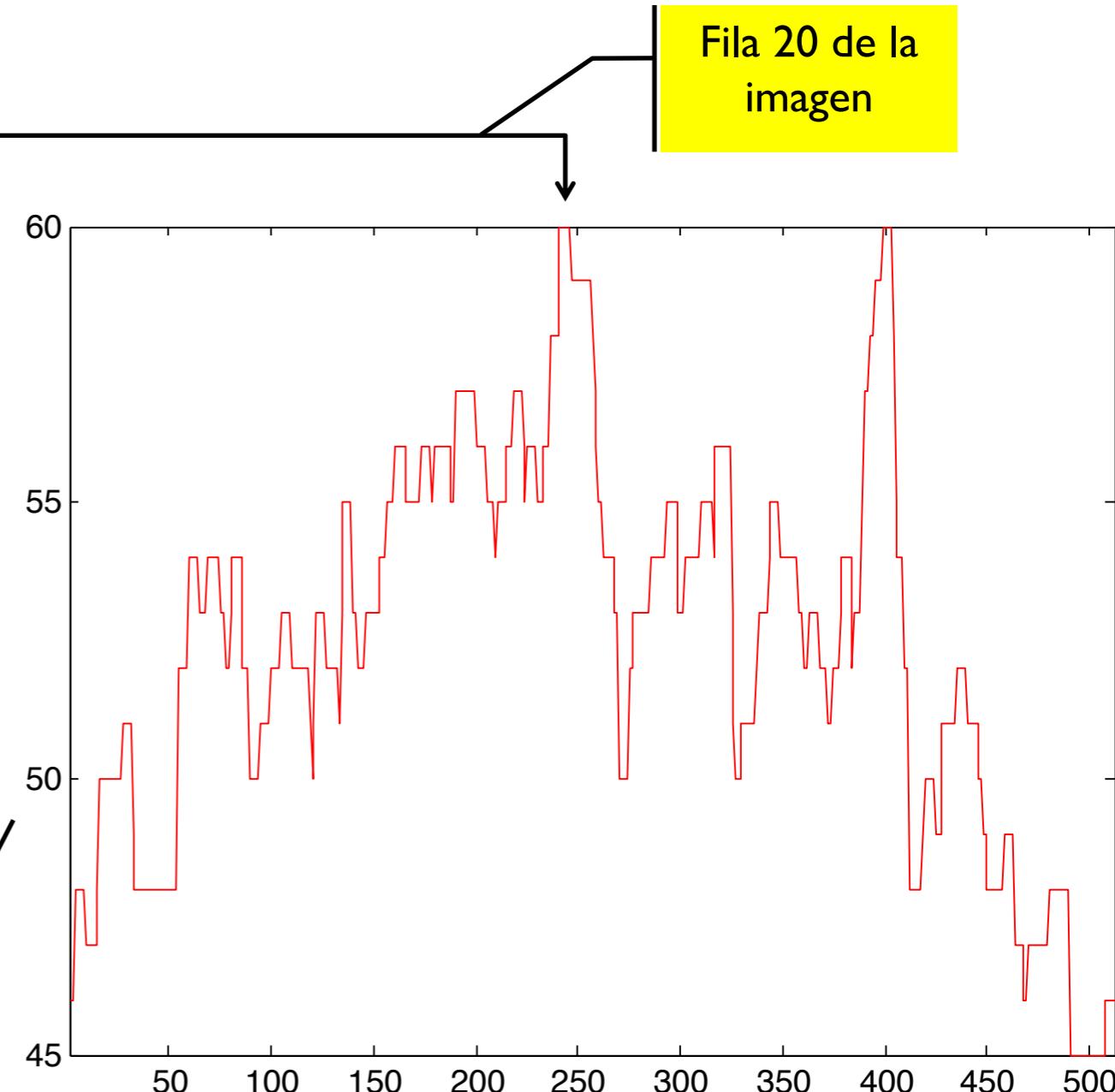


Imagen dilatada



- ▶ Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **dilatación** el proceso es el siguiente.



Imagen original

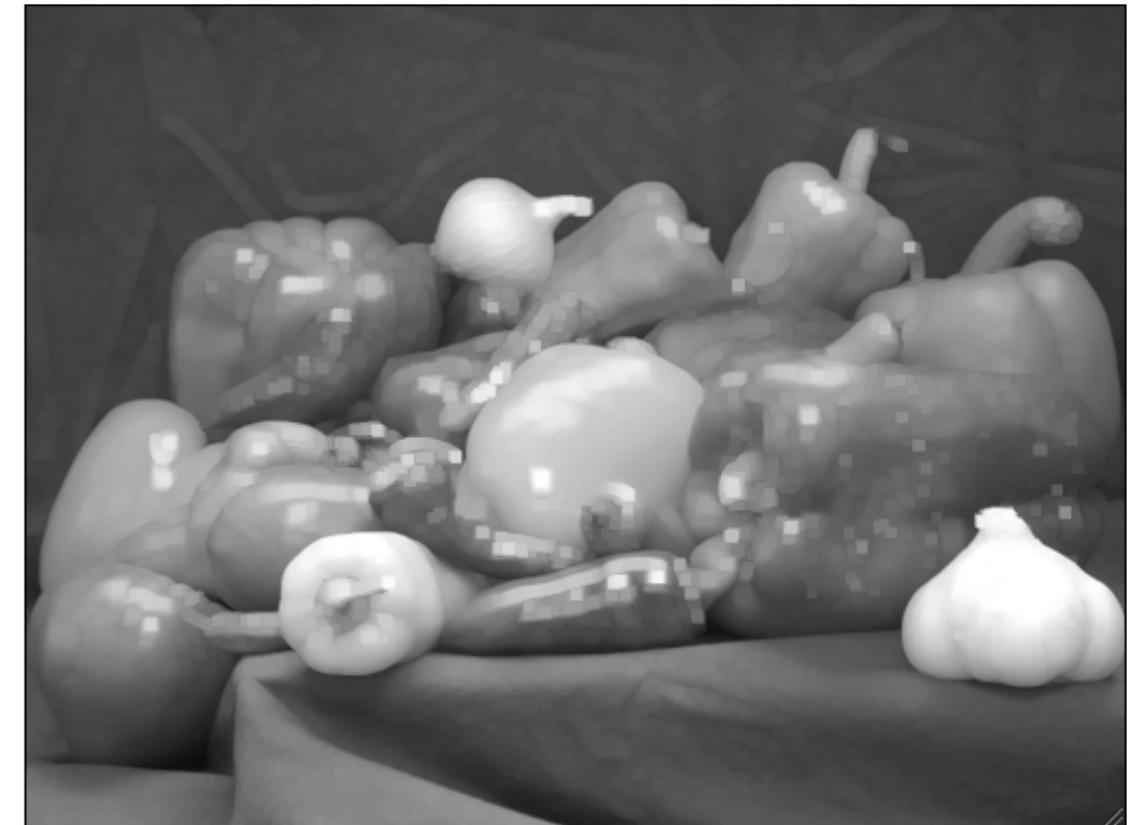
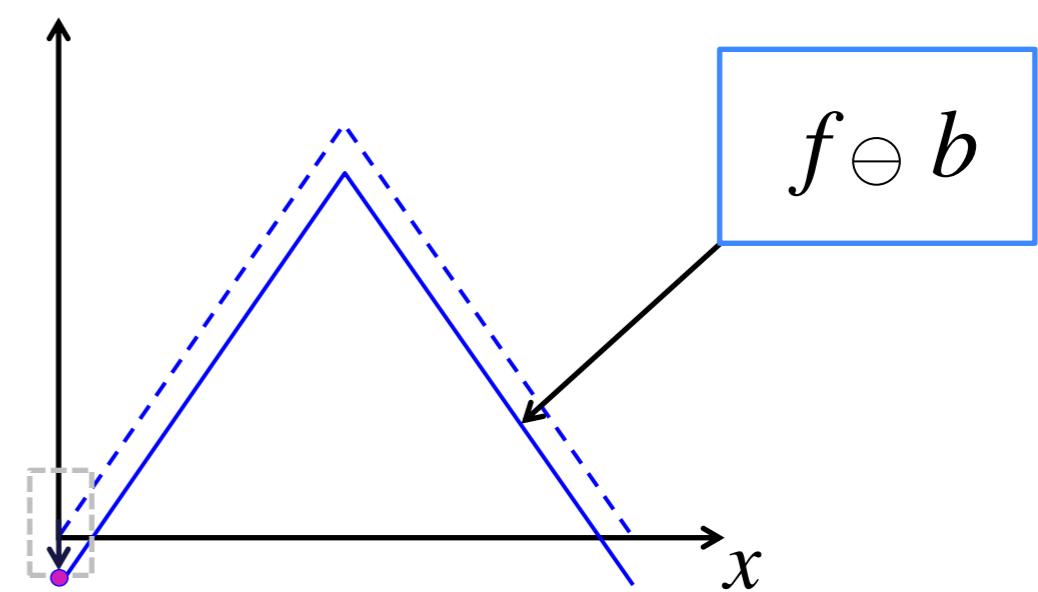
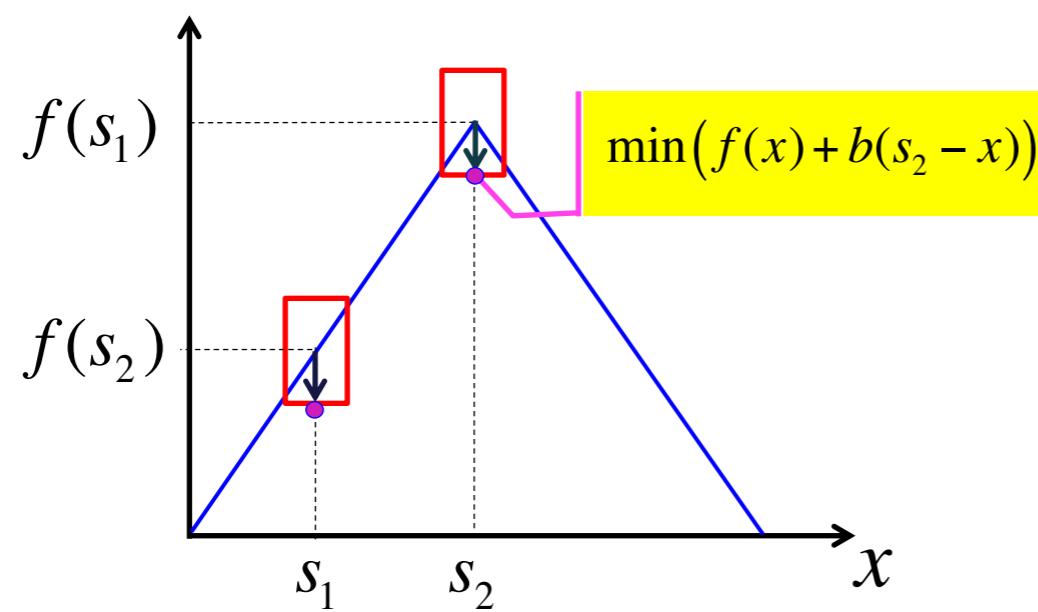
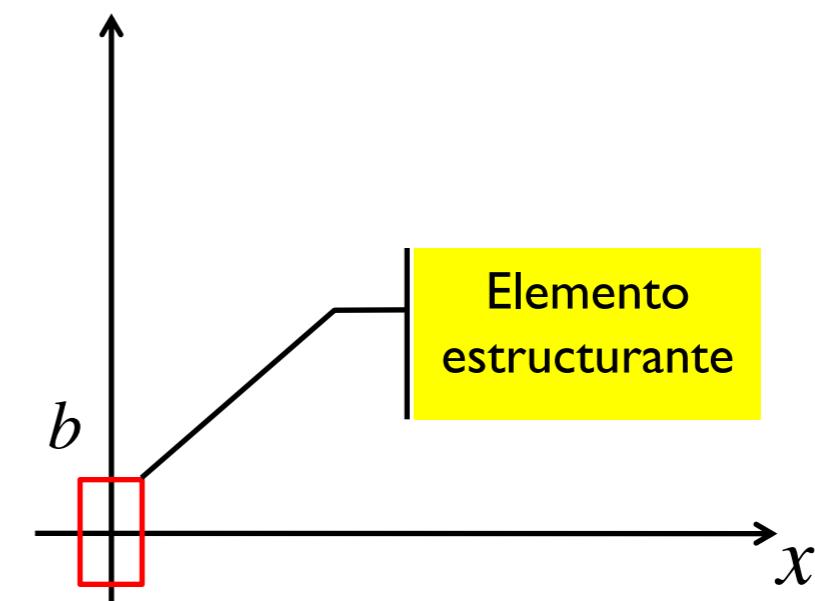
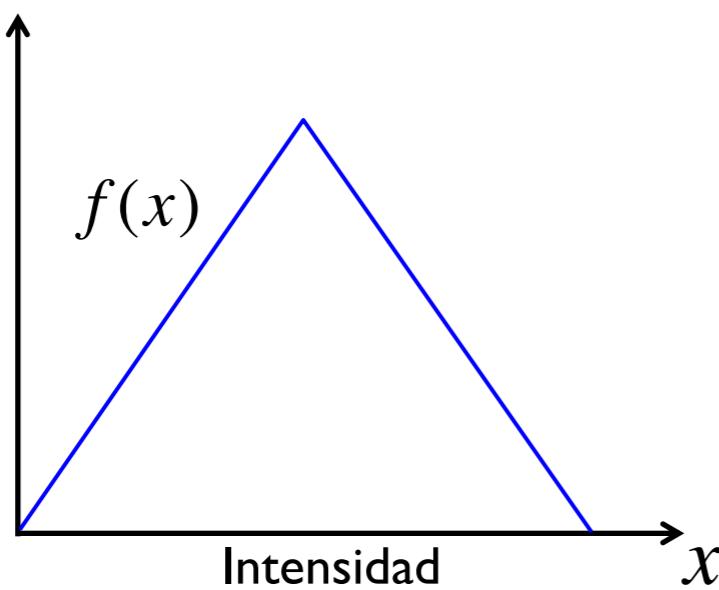


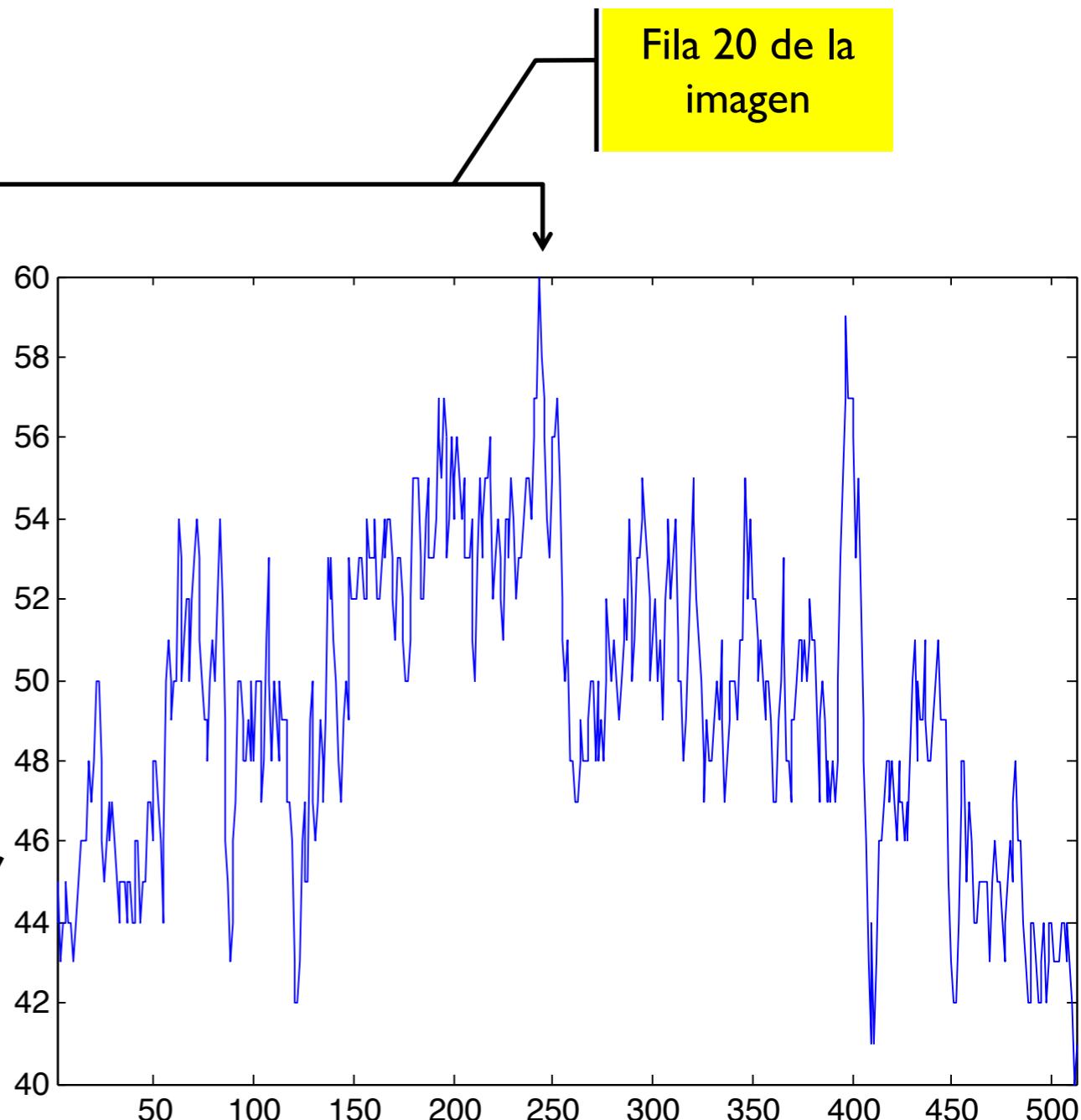
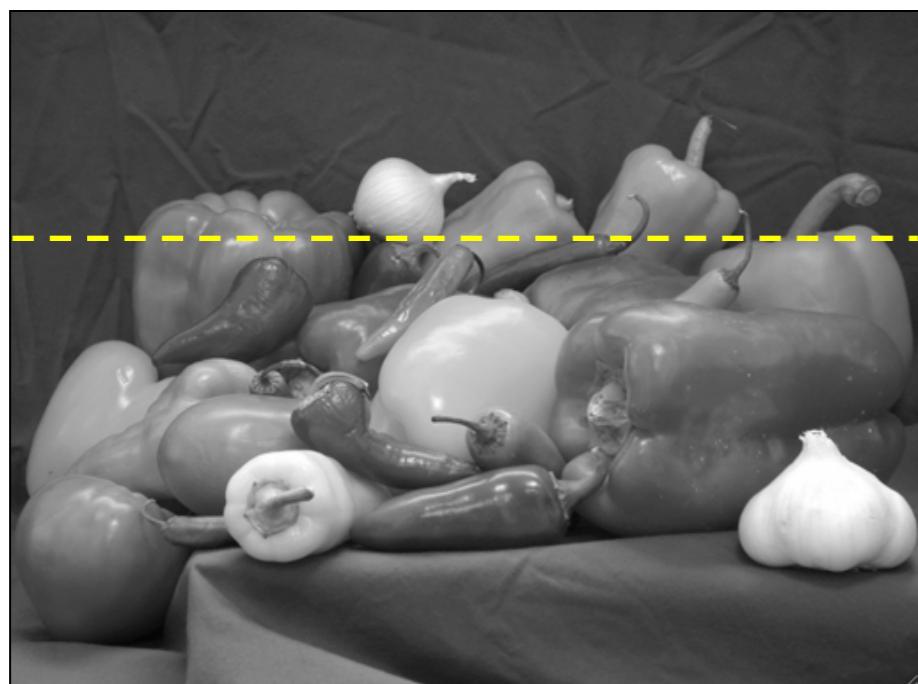
Imagen dilatada

- Procesamiento morfológico de imágenes
  - Dilación en escala de grises
  - Erosión en escala de grises
  - Comparación
  - Apertura y clausura
  - Suavización y gradiente morfológico
  - Transformación Top-Hat

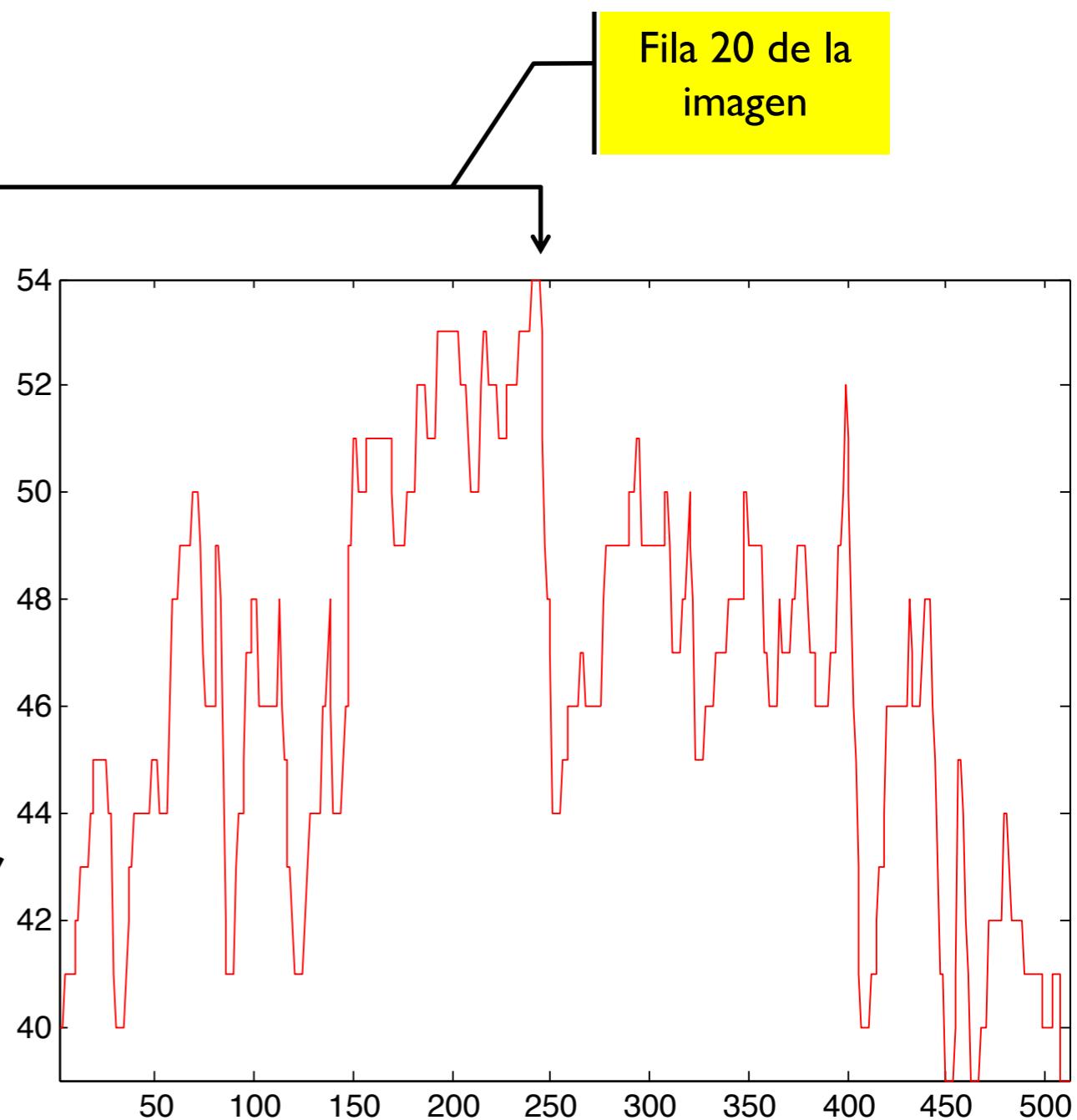
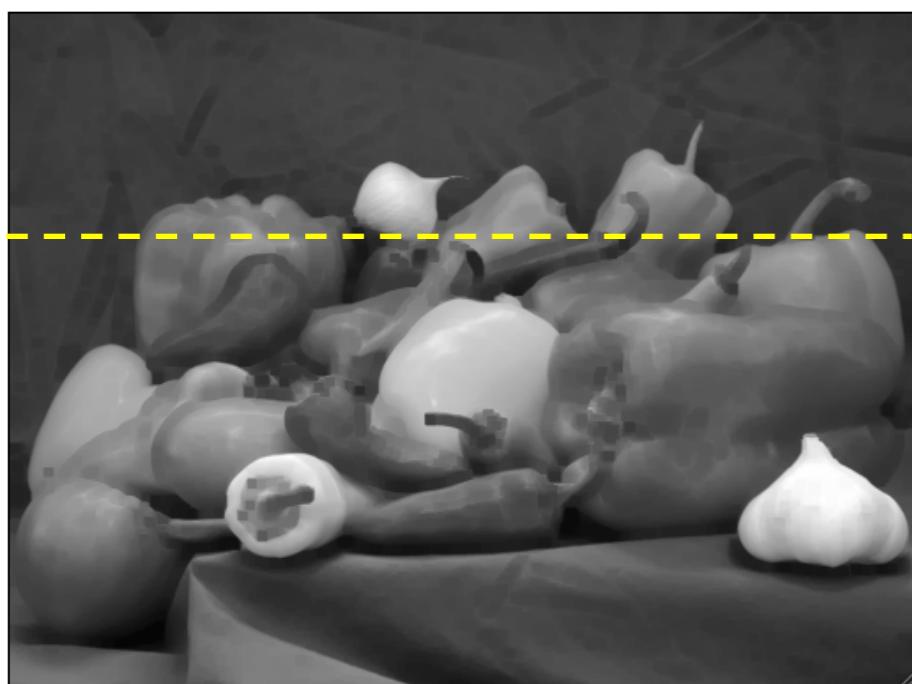
- Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **erosión** el proceso es el siguiente.



- ▶ Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **erosión** el proceso es el siguiente.



- ▶ Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **erosión** el proceso es el siguiente.



- ▶ Los operadores morfológicos pueden ser empleados en imágenes en escala de grises. En el caso de la **erosión** el proceso es el siguiente.

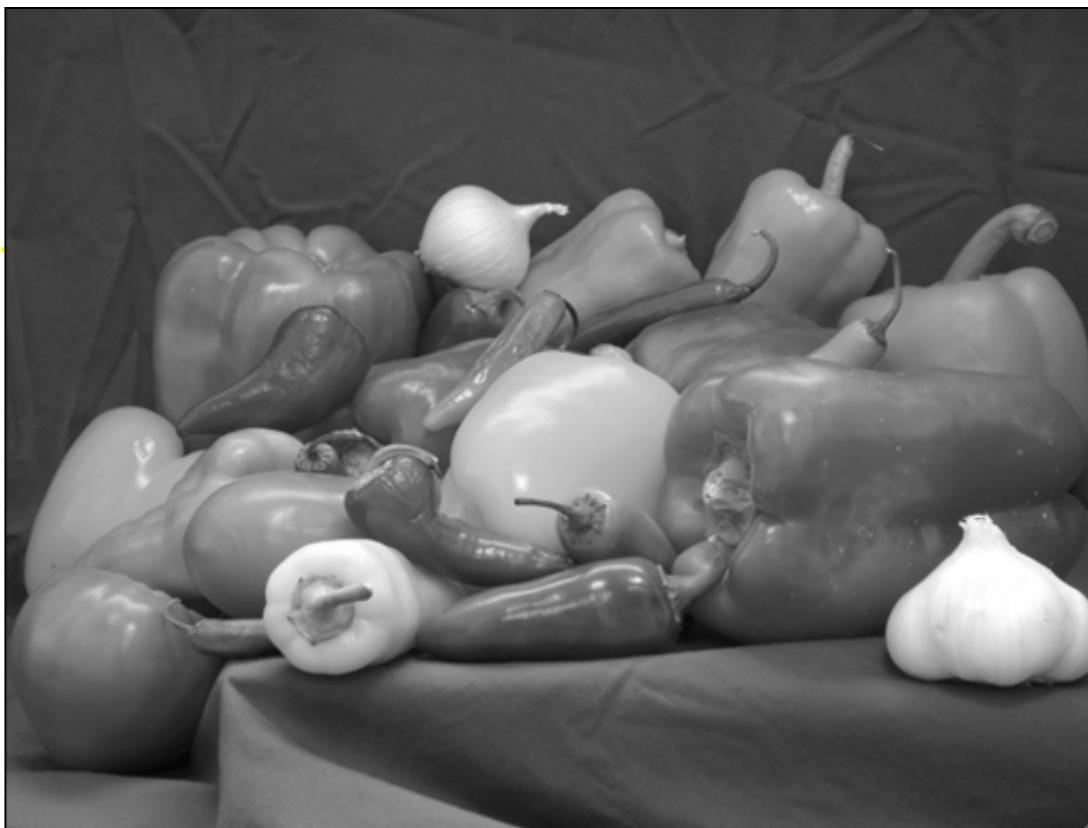


Imagen original

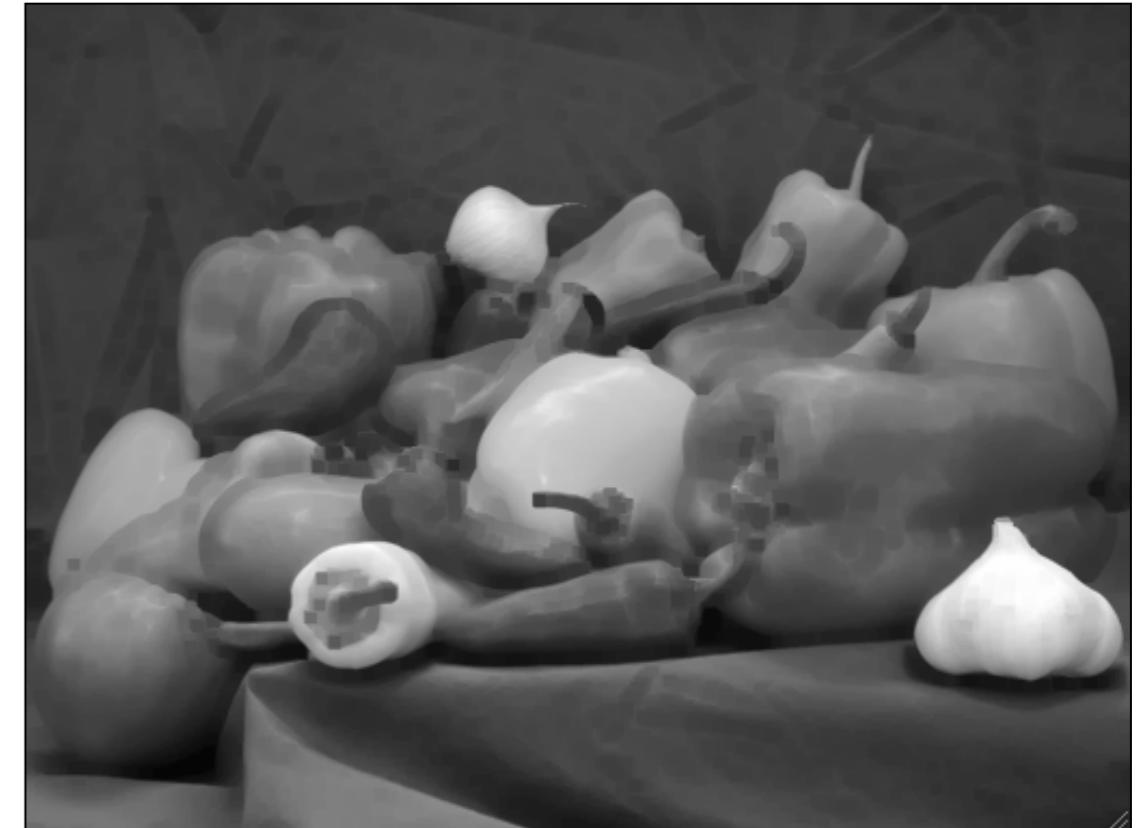
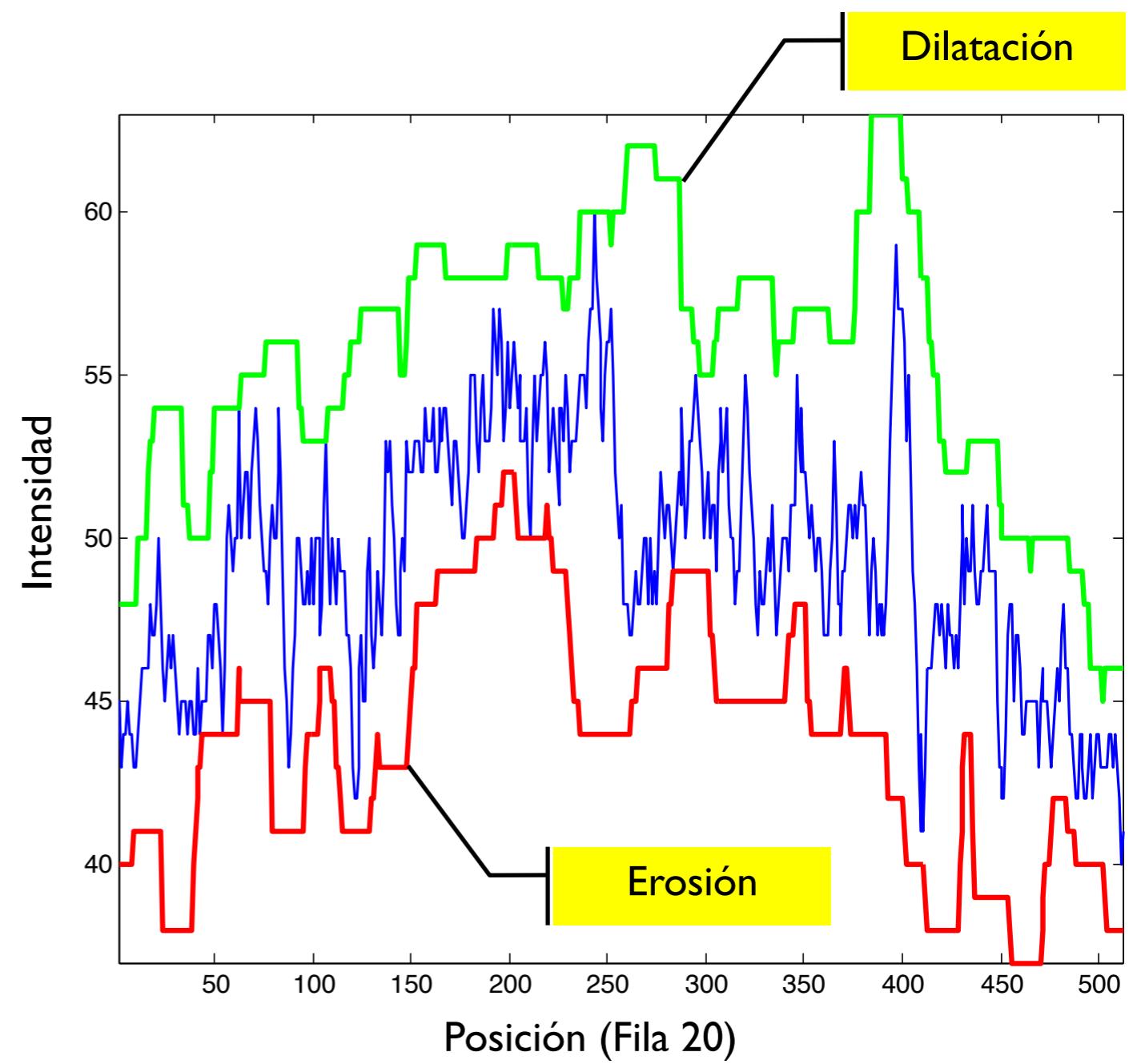
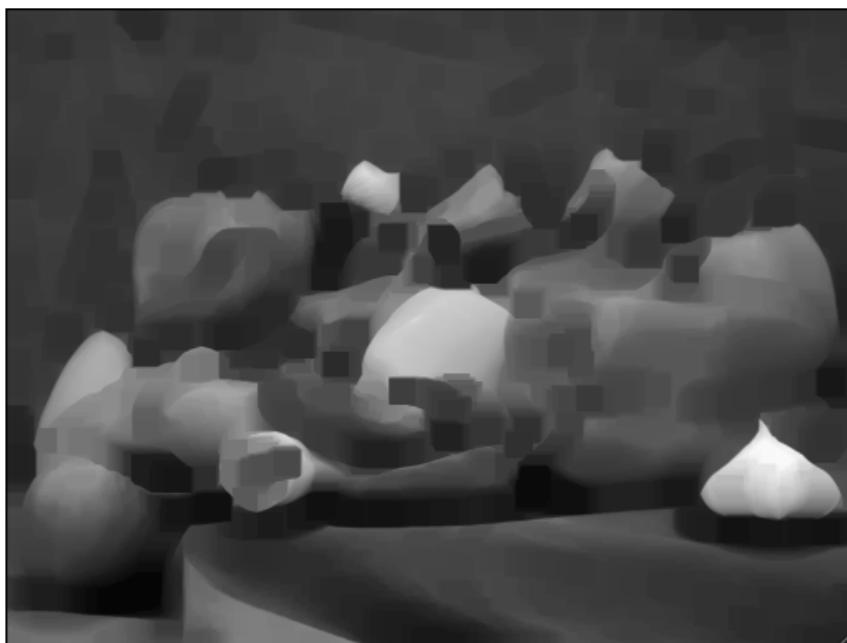


Imagen erosionada

- Procesamiento morfológico de imágenes
  - Dilación en escala de grises
  - Erosión en escala de grises
  - Comparación
  - Apertura y clausura
  - Suavización y gradiente morfológico
  - Transformación Top-Hat



- ▶ La dilatación oscurece la imagen y la erosión las aclara. Esto se debe a la operación que modifica las intensidades en los niveles de gris.



- Procesamiento morfológico de imágenes
  - Dilación en escala de grises
  - Erosión en escala de grises
  - Comparación
  - Apertura y clausura
  - Suavización y gradiente morfológico
  - Transformación Top-Hat

## ► Apertura

- La **apertura** es una función que tiene dos pasos. Primero, aplicamos erosión y luego a dicho resultado una dilatación. Como recordaremos, su principal utilidad es la reducción del ruido. Veamos el resultado aplicado a imágenes de grises.

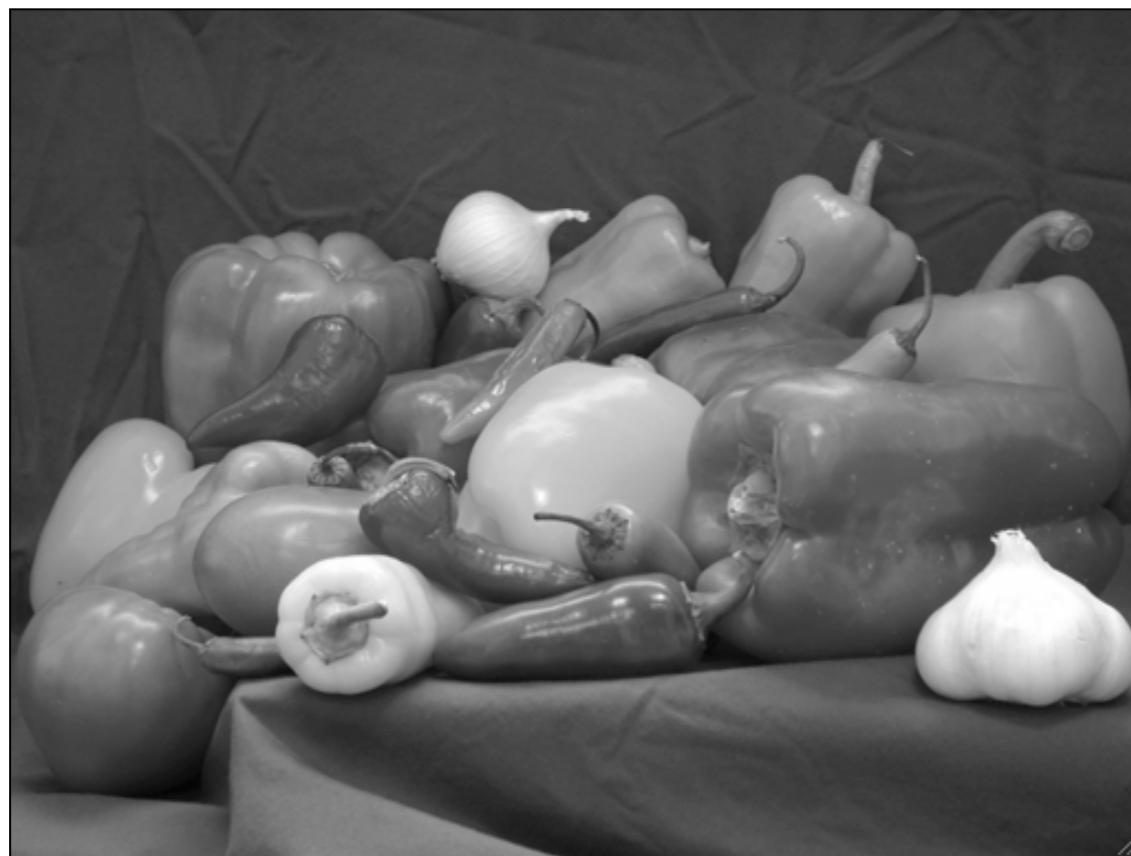
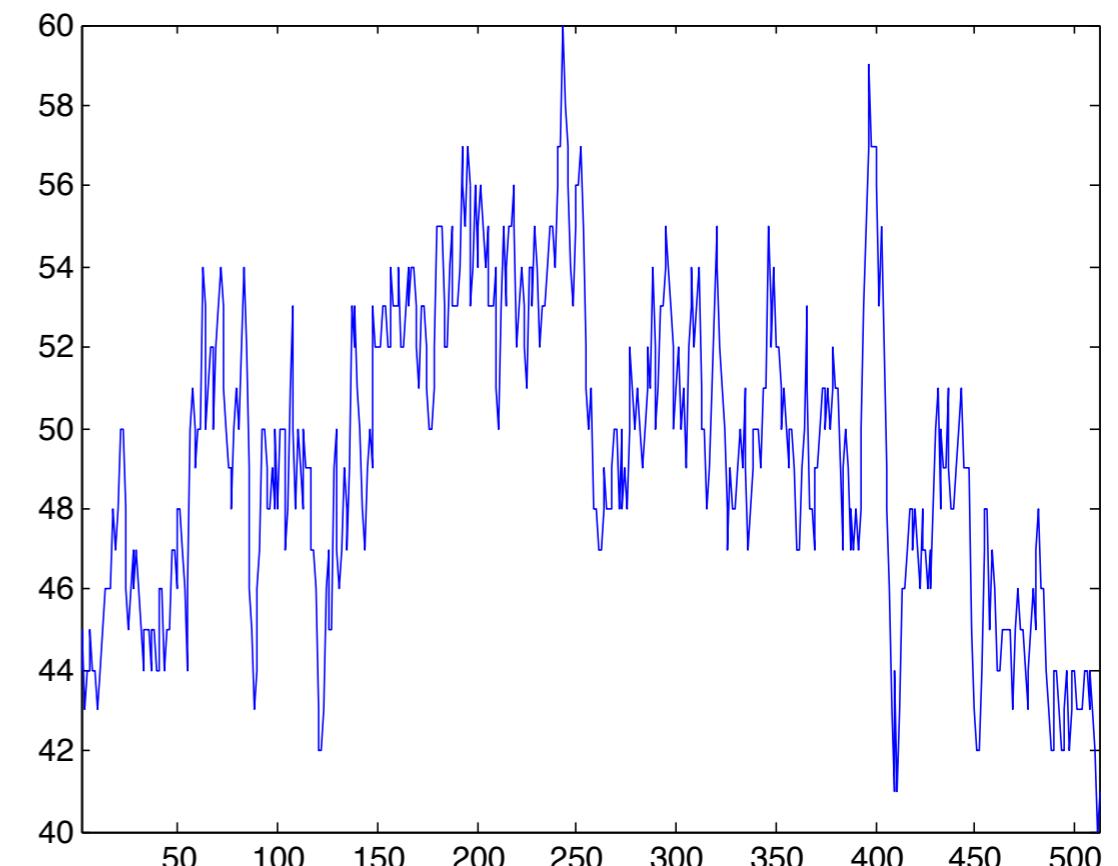
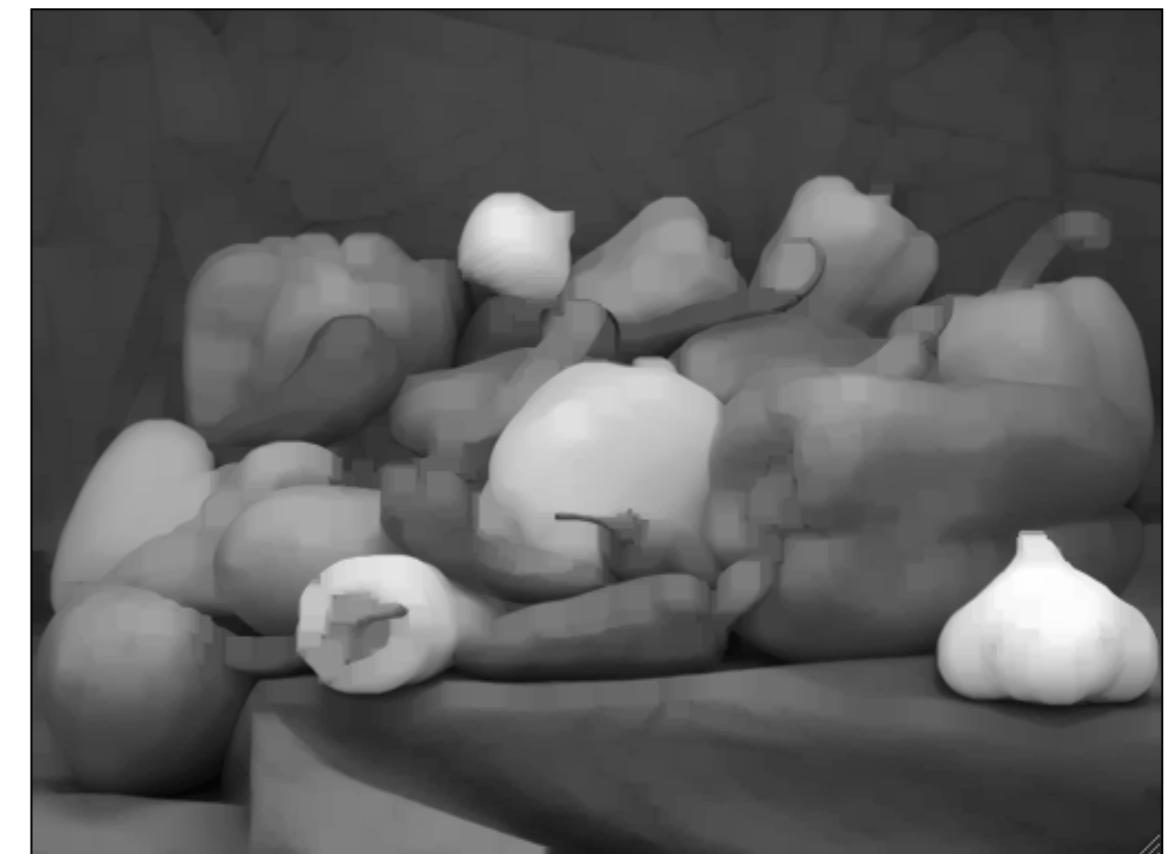
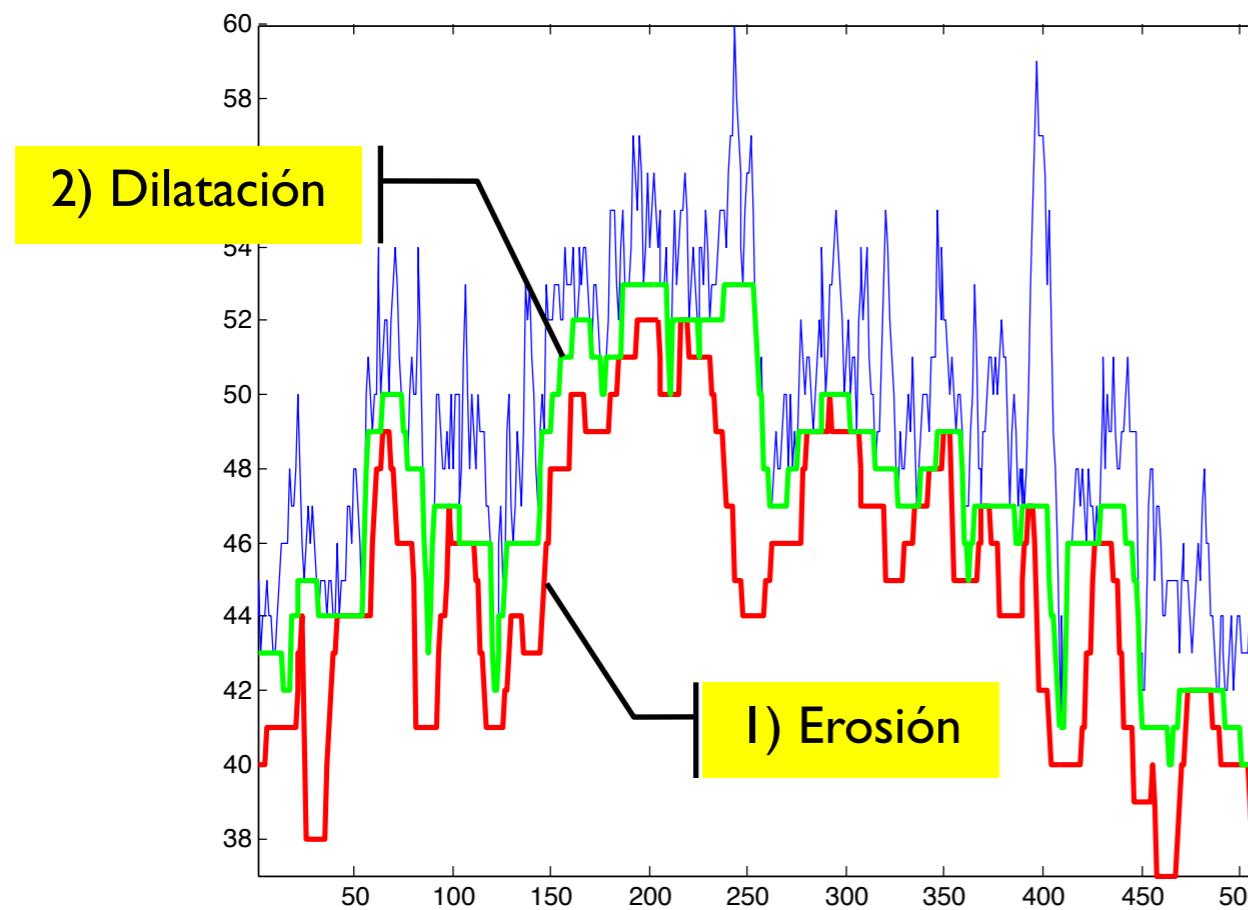


Imagen original



## ► Apertura

- La **apertura** es una función que tiene dos pasos. Primero, aplicamos erosión y luego a dicho resultado una dilatación. Como recordaremos, su principal utilidad es la reducción del ruido. Veamos el resultado aplicado a imágenes de grises.



Resultado de apertura

## ► Apertura

- La **apertura** es una función que tiene dos pasos. Primero, aplicamos erosión y luego a dicho resultado una dilatación. Como recordaremos, su principal utilidad es la reducción del ruido. Veamos el resultado aplicado a imágenes de grises.

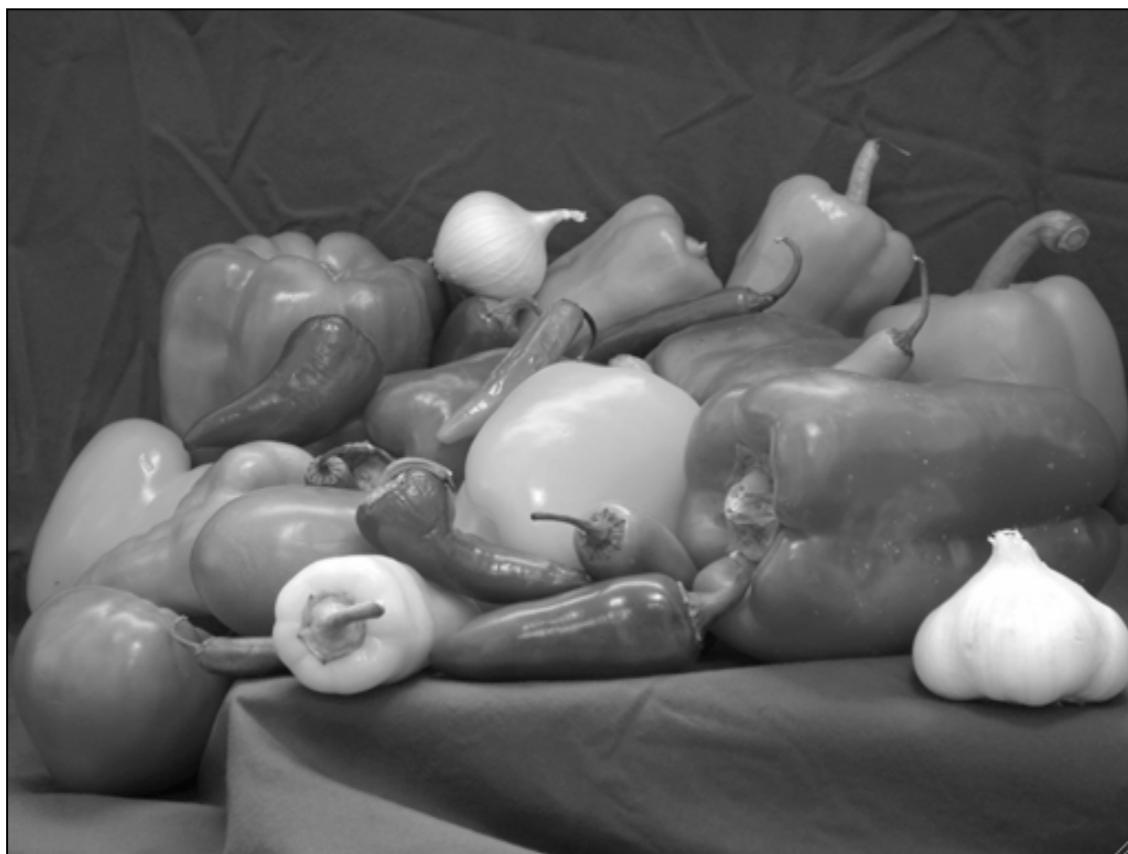
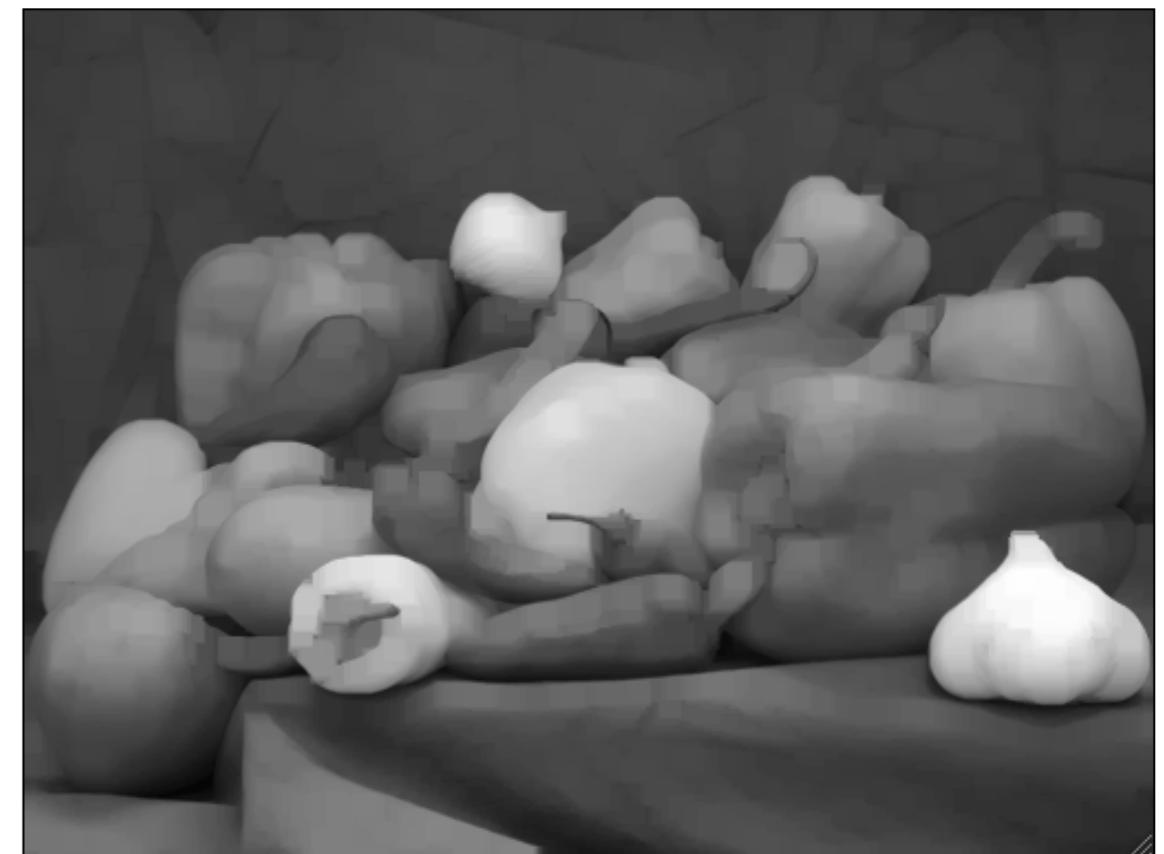


Imagen original



Resultado de apertura

## ▶ Clausura

- La **clausura** es una función que tiene dos pasos. Primero, aplicamos dilatación y luego a dicho resultado una erosión. Como recordaremos, su principal utilidad es unir regiones. Veamos el resultado aplicado a imágenes de grises.

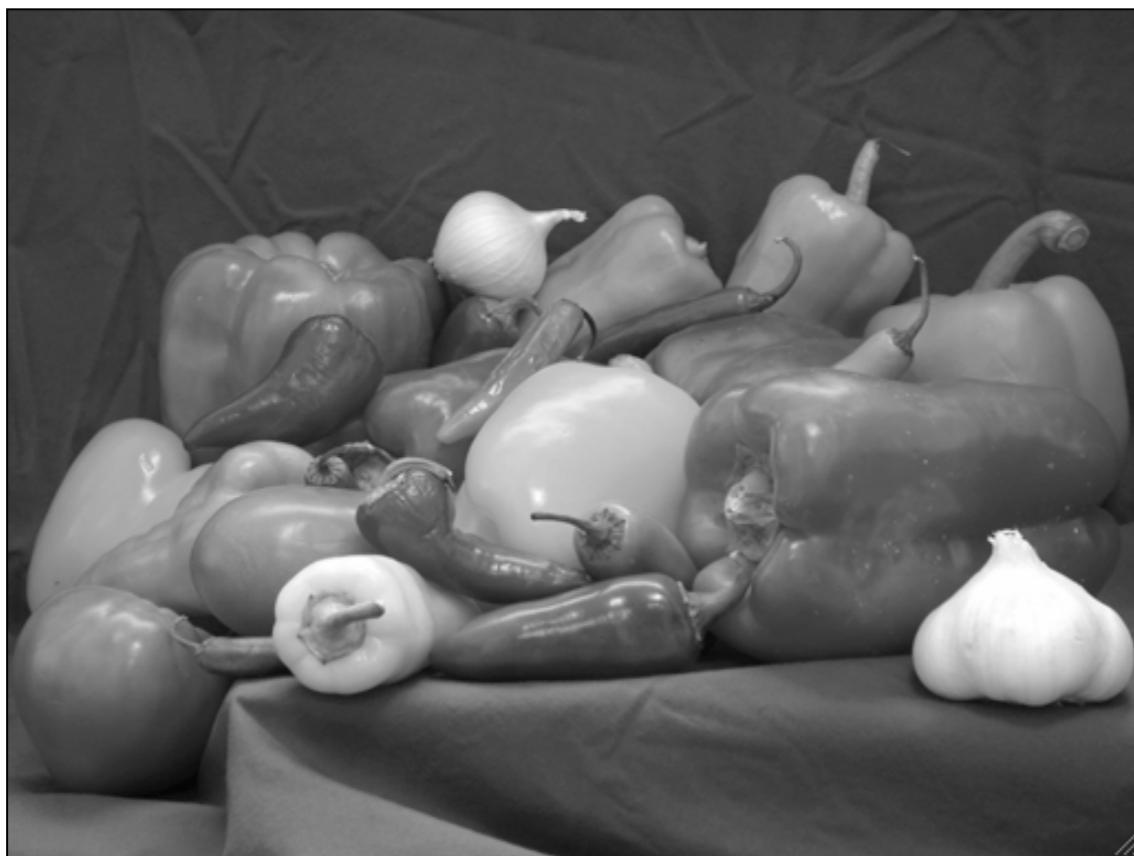
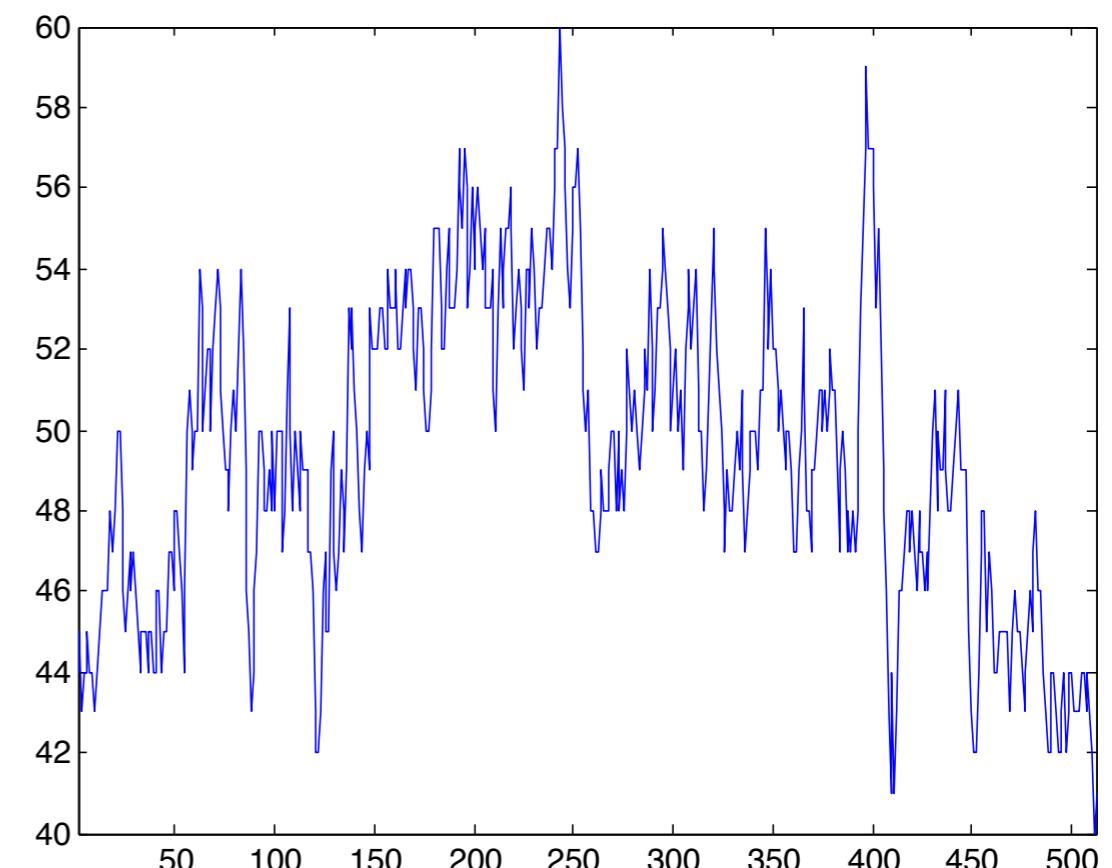
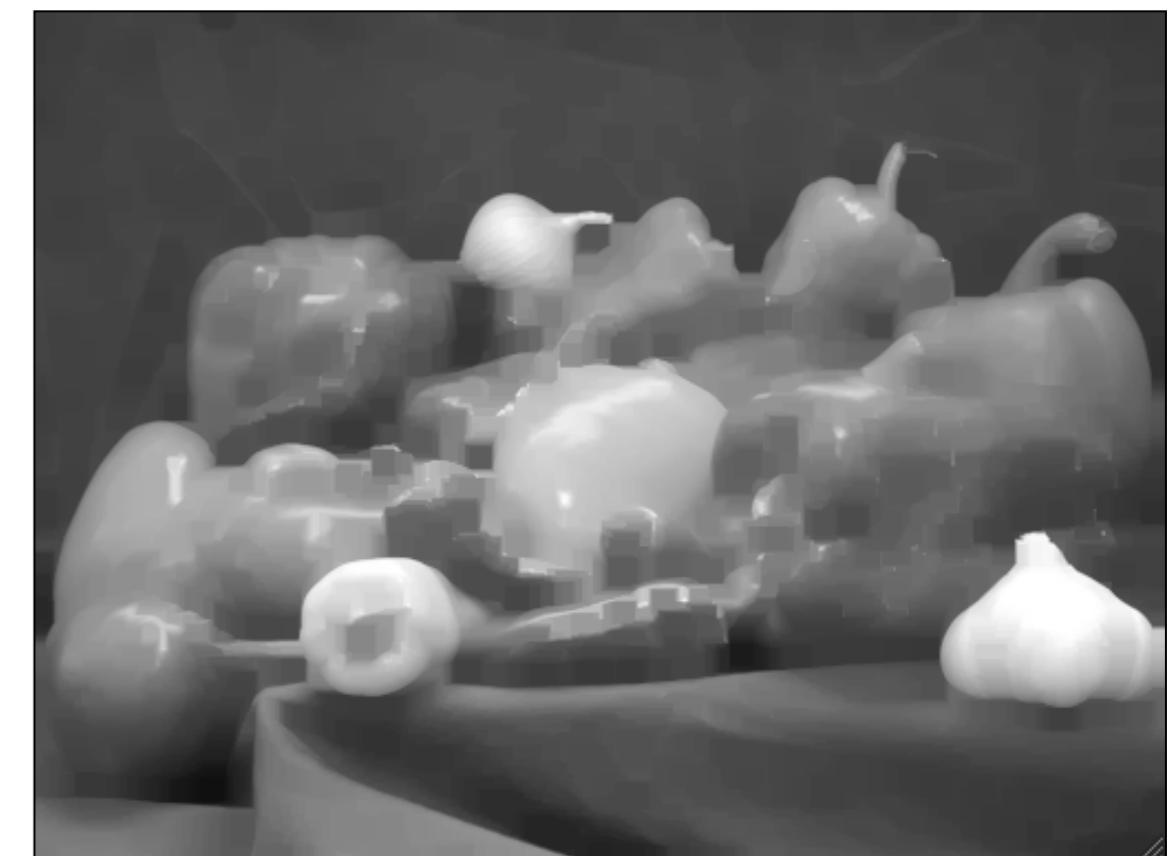
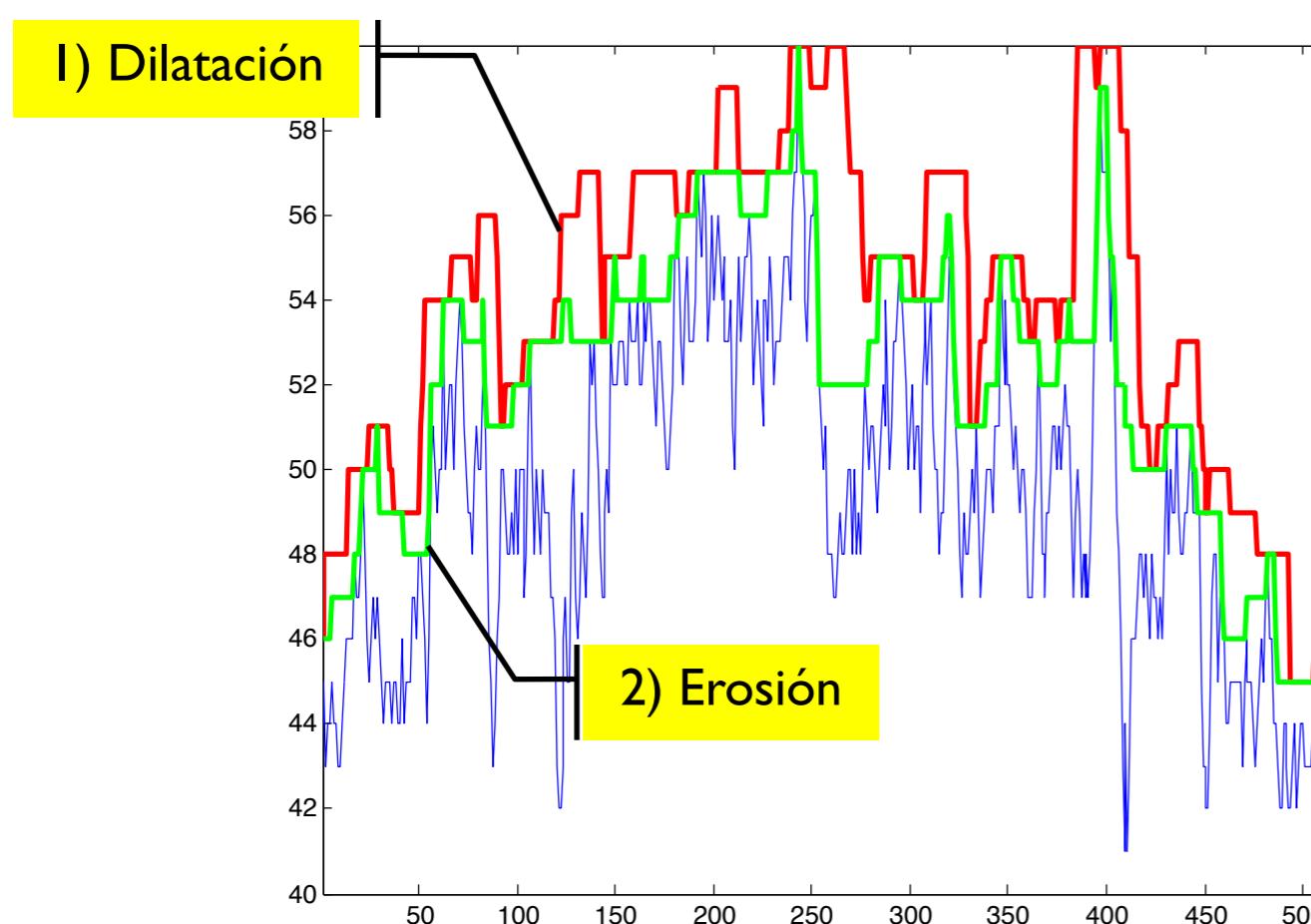


Imagen original



## Clausura

- La **clausura** es una función que tiene dos pasos. Primero, aplicamos dilatación y luego a dicho resultado una erosión. Como recordaremos, su principal utilidad es unir regiones. Veamos el resultado aplicado a imágenes de grises.



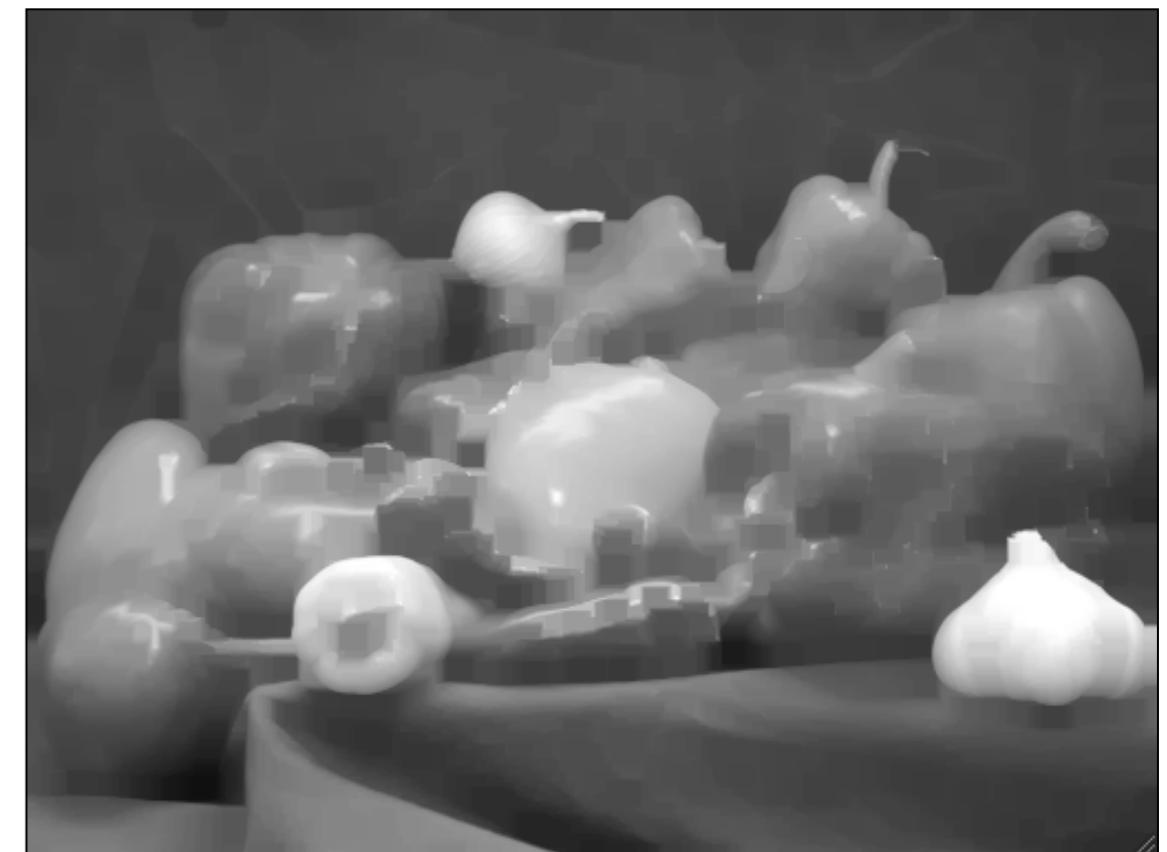
Resultado de clausura

## ▶ Clausura

- La **clausura** es una función que tiene dos pasos. Primero, aplicamos dilatación y luego a dicho resultado una erosión. Como recordaremos, su principal utilidad es unir regiones. Veamos el resultado aplicado a imágenes de grises.



Imagen original



Resultado de clausura

- Procesamiento morfológico de imágenes
  - Dilación en escala de grises
  - Erosión en escala de grises
  - Comparación
  - Apertura y clausura
  - Suavización y gradiente morfológico
  - Transformación Top-Hat



- ▶ La suavización es el resultado de aplicar una **apertura** y a dicho resultado una **clausura**. Recordemos que la apertura consiste en aplicar erosión y luego dilatación. Y la clausura consiste en aplicar dilatación y luego erosión.

$$(A \circ B) \bullet (A \circ B)$$



Imagen original

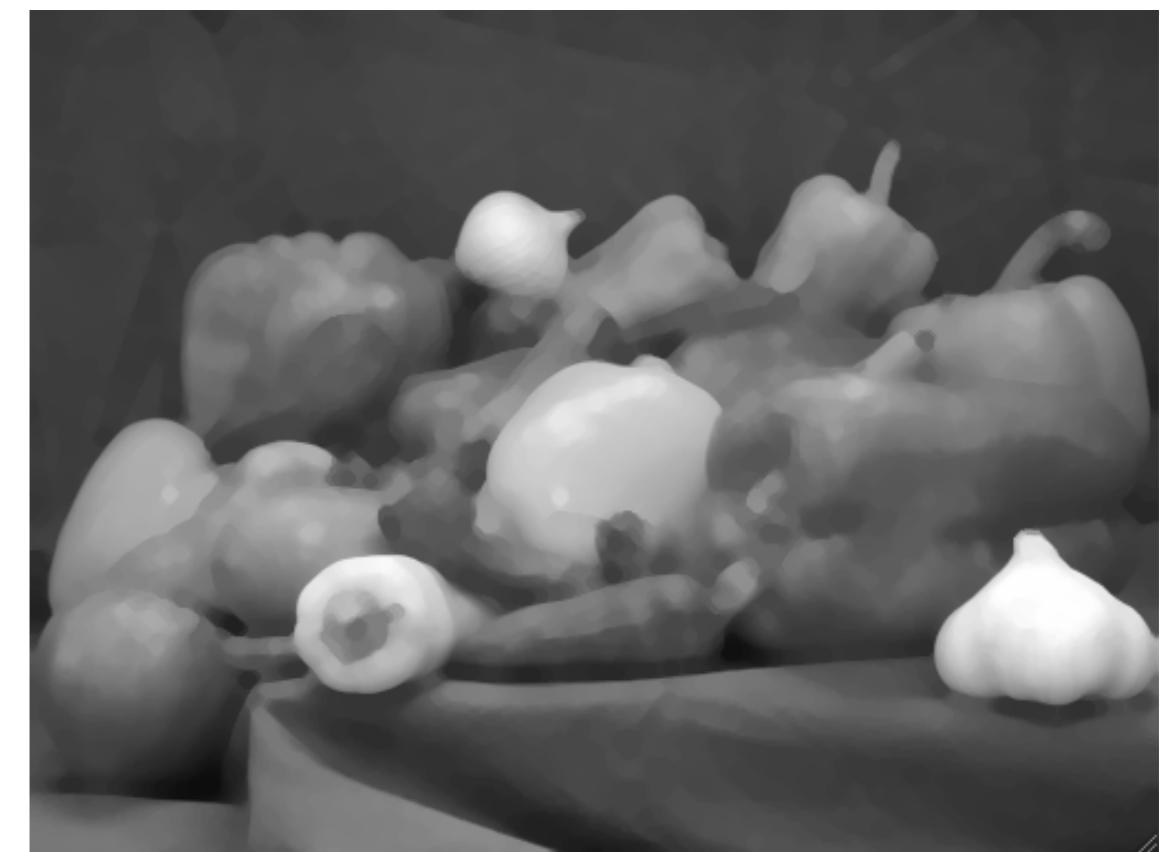


Imagen suavizada

- ▶ El gradiente morfológico consiste en restar la **dilatación** con una **erosión**. Matemáticamente consiste en:

$$R = (A \oplus B) - (A \ominus B)$$

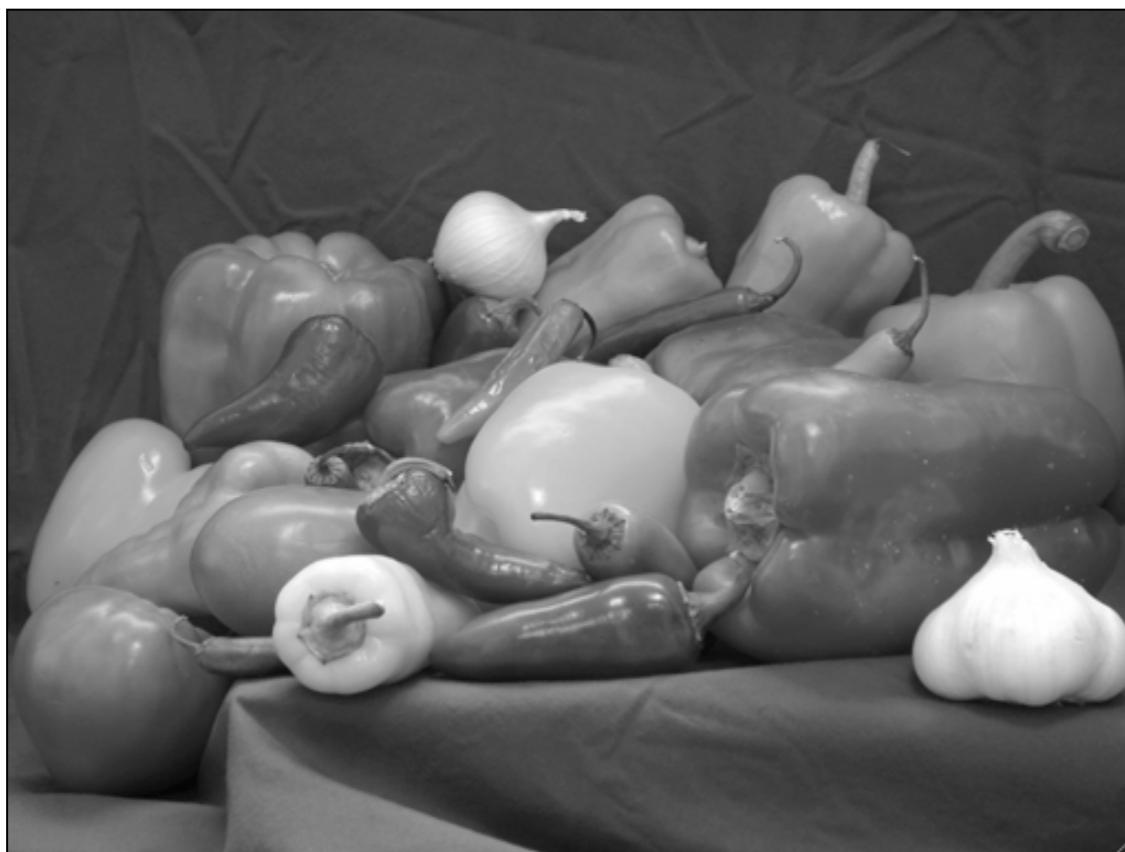


Imagen original

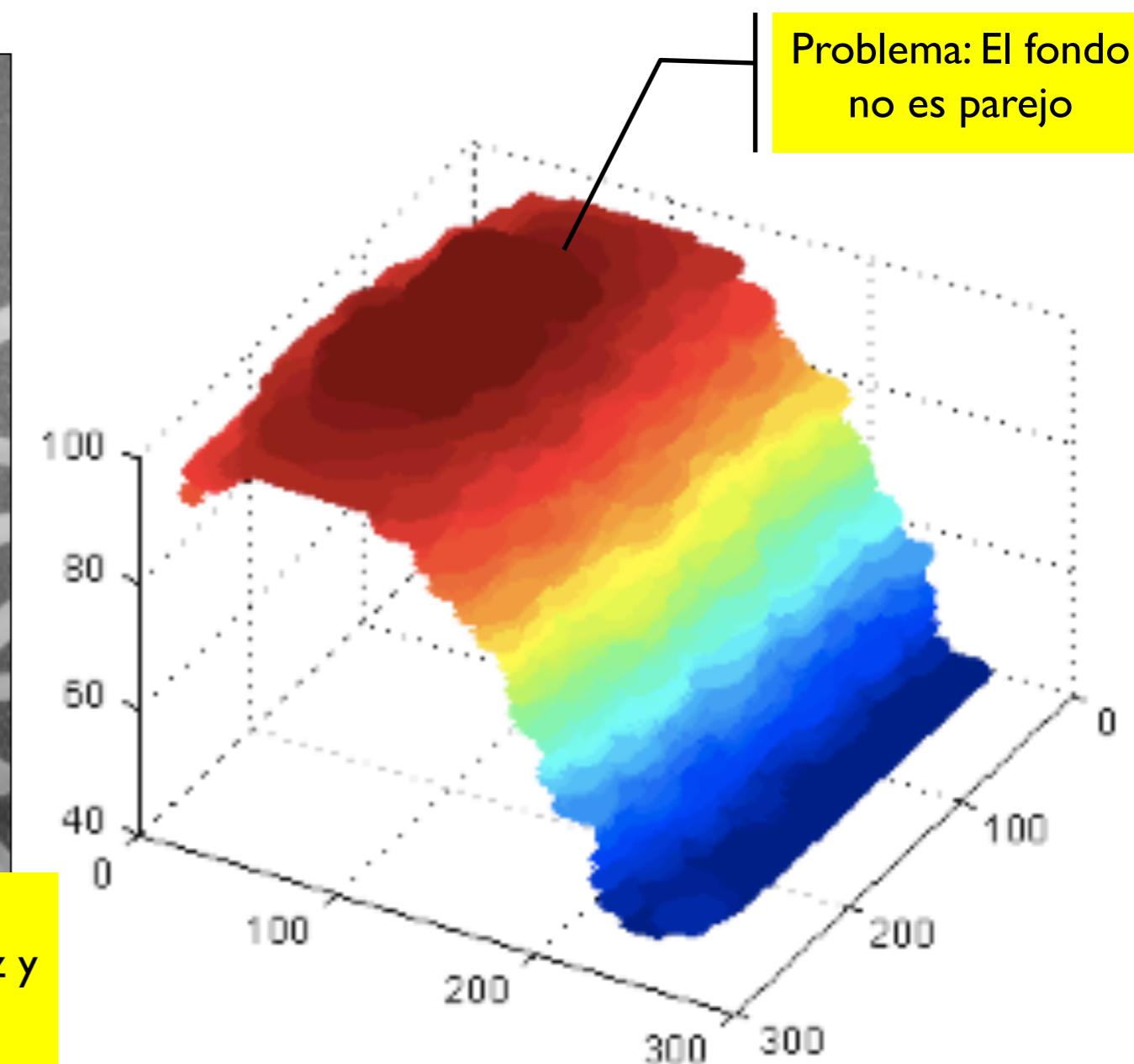
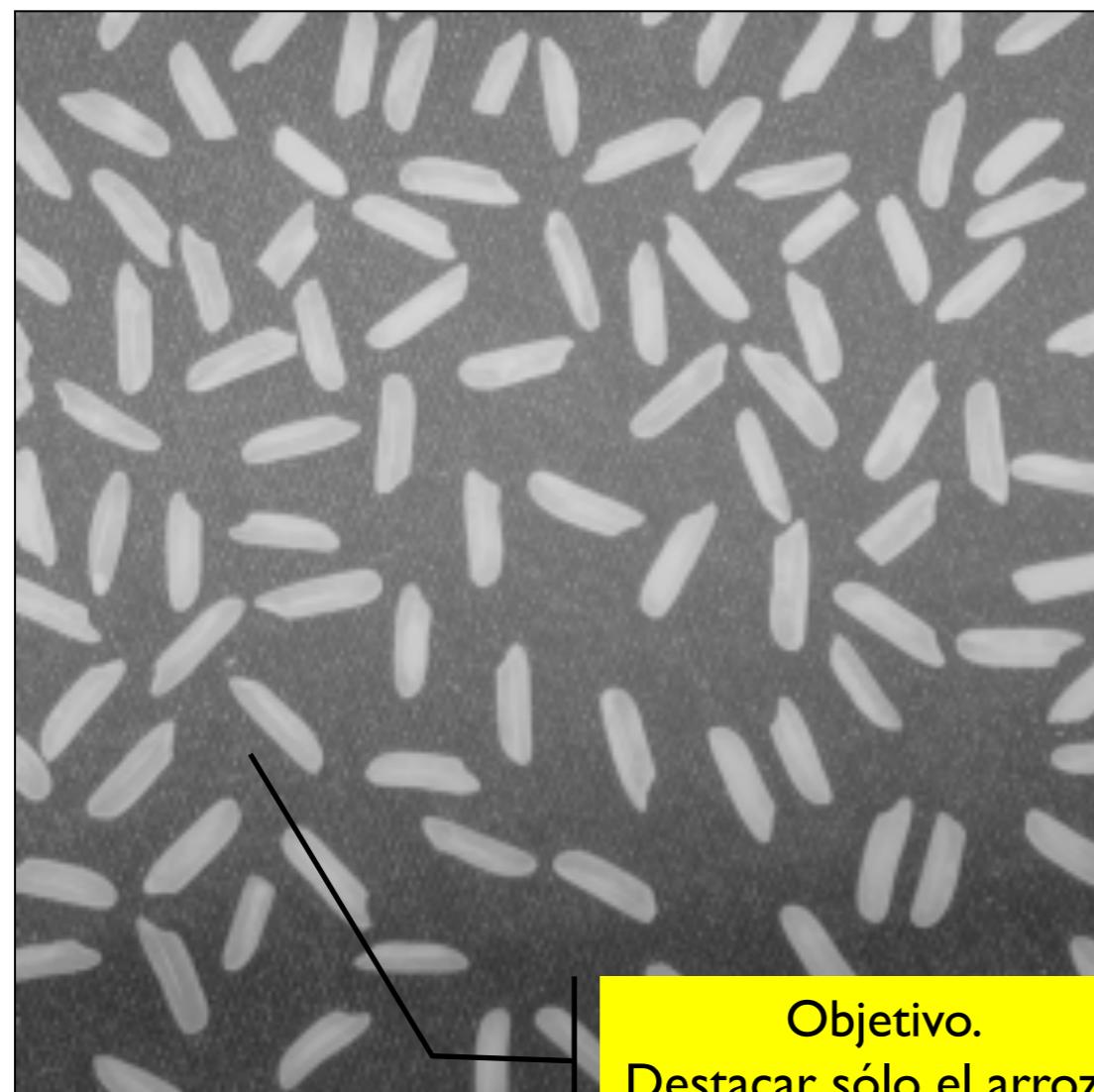


Gradiente morfológico

- Procesamiento morfológico de imágenes
  - Dilación en escala de grises
  - Erosión en escala de grises
  - Comparación
  - Apertura y clausura
  - Suavización y gradiente morfológico
  - Transformación Top-Hat

# Transformación Top-Hat

- ▶ Esta transformación es especialmente útil cuando queremos segmentar regiones. Veamos el siguiente ejemplo para ilustrar el problema.



## ▶ Filtro Top-Hat

- **El primer paso** de esta transformación es aplicar el operador apertura a la imagen original con un elemento estructurante de gran tamaño.

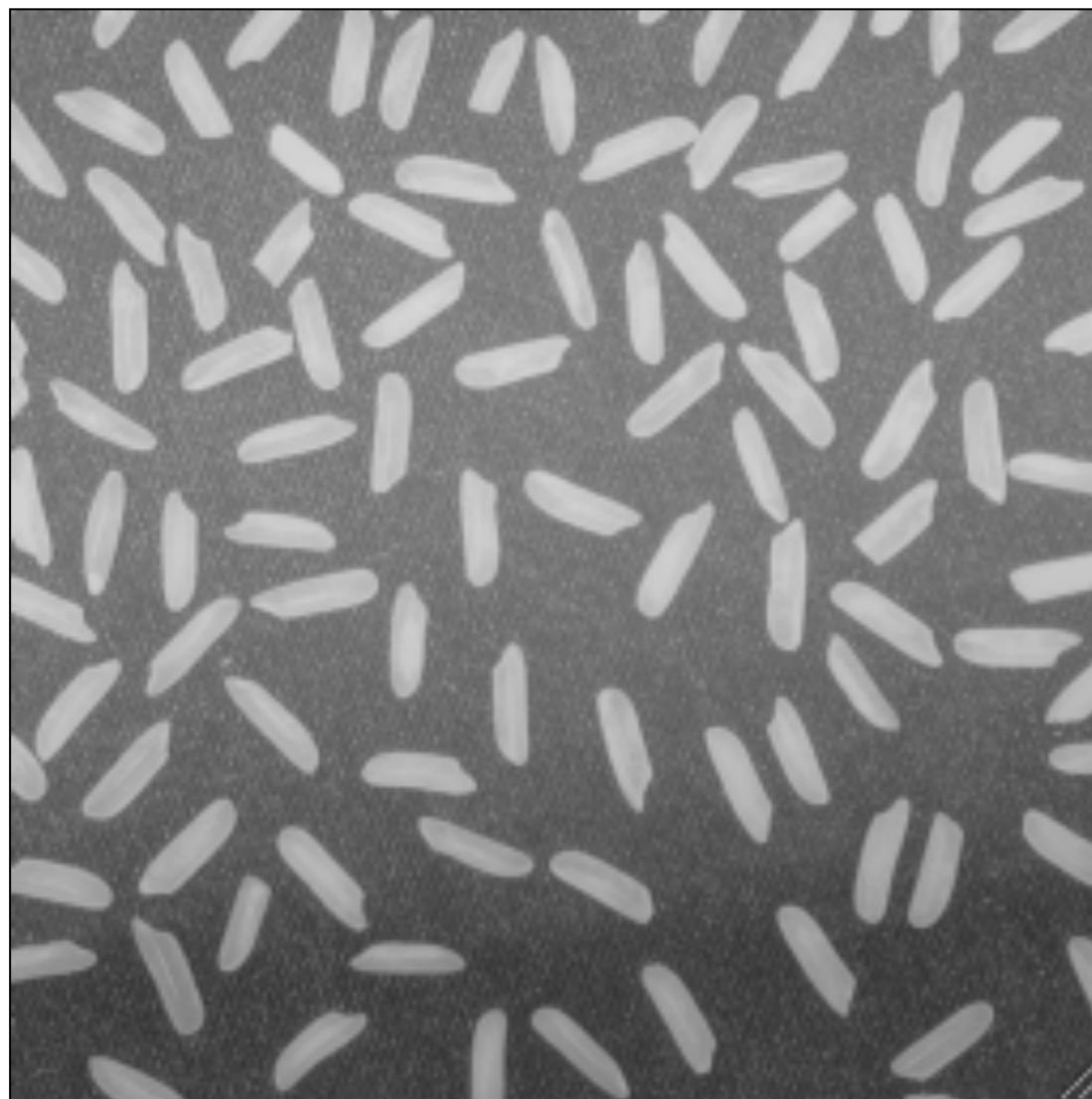


Imagen original



Resultado de la apertura

## ▶ Filtro Top-Hat

- **El segundo paso** de esta transformación es restar la imagen original con el resultado de la apertura.

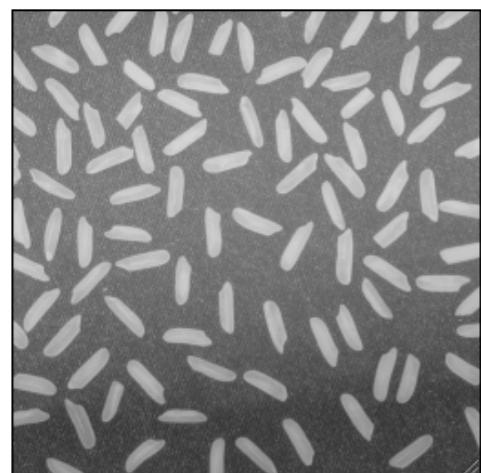
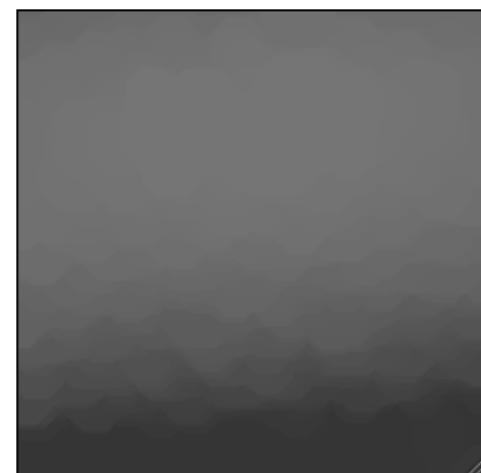


Imagen original (A)



Apertura

$$A - A \circ B = A - (A \ominus B) \oplus B$$

Gracias al filtro Top-Hat  
podemos eliminar el  
fondo.

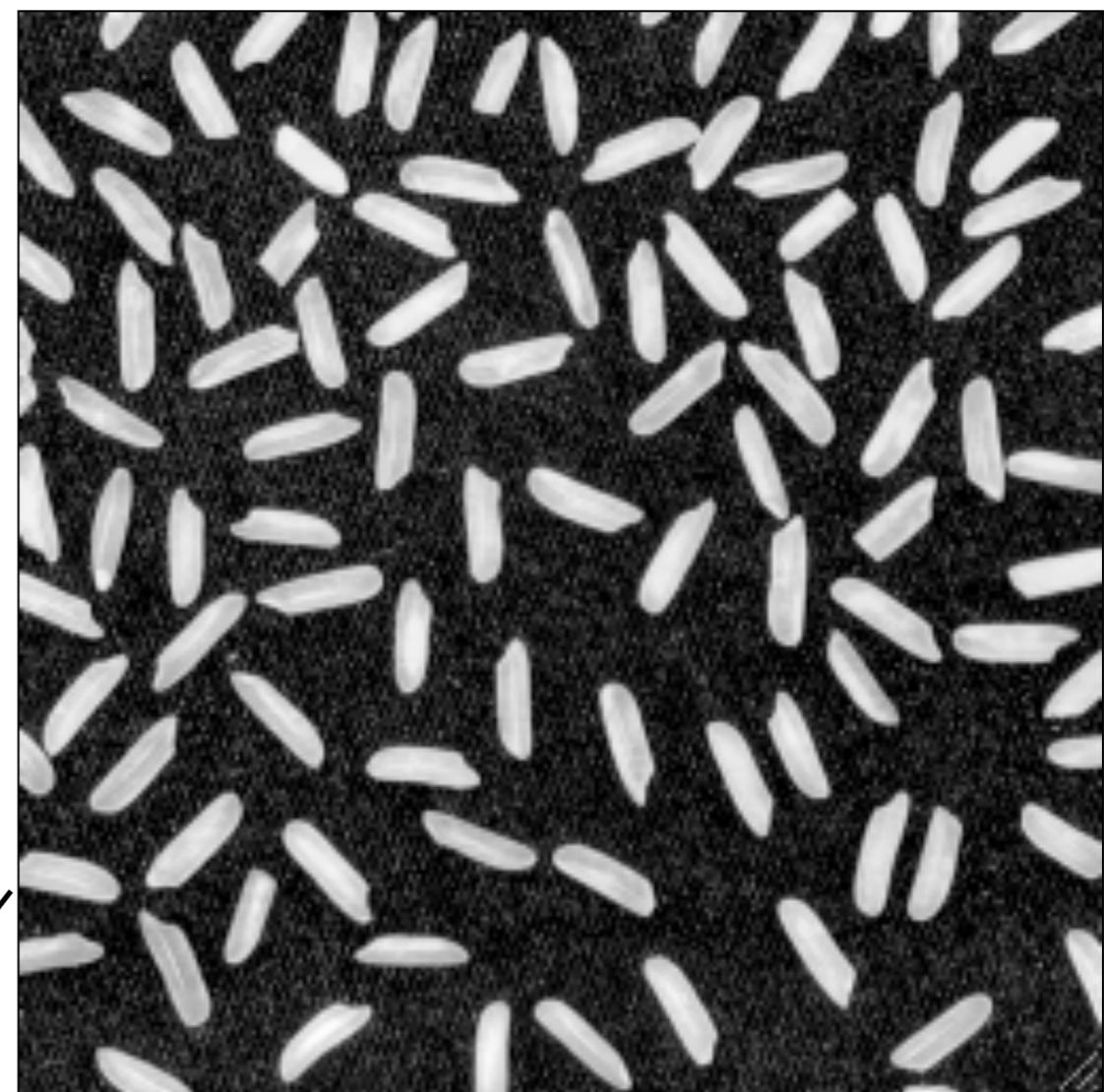


Imagen Top-Hat

## ▶ Filtro Top-Hat

- **El segundo paso** de esta transformación es restar la imagen original con el resultado de la apertura.

```
img = cv2.imread('rice.png')
N = 25
kernel = cv2.getStructuringElement(cv2.MORPH_CROSS,(N,N))

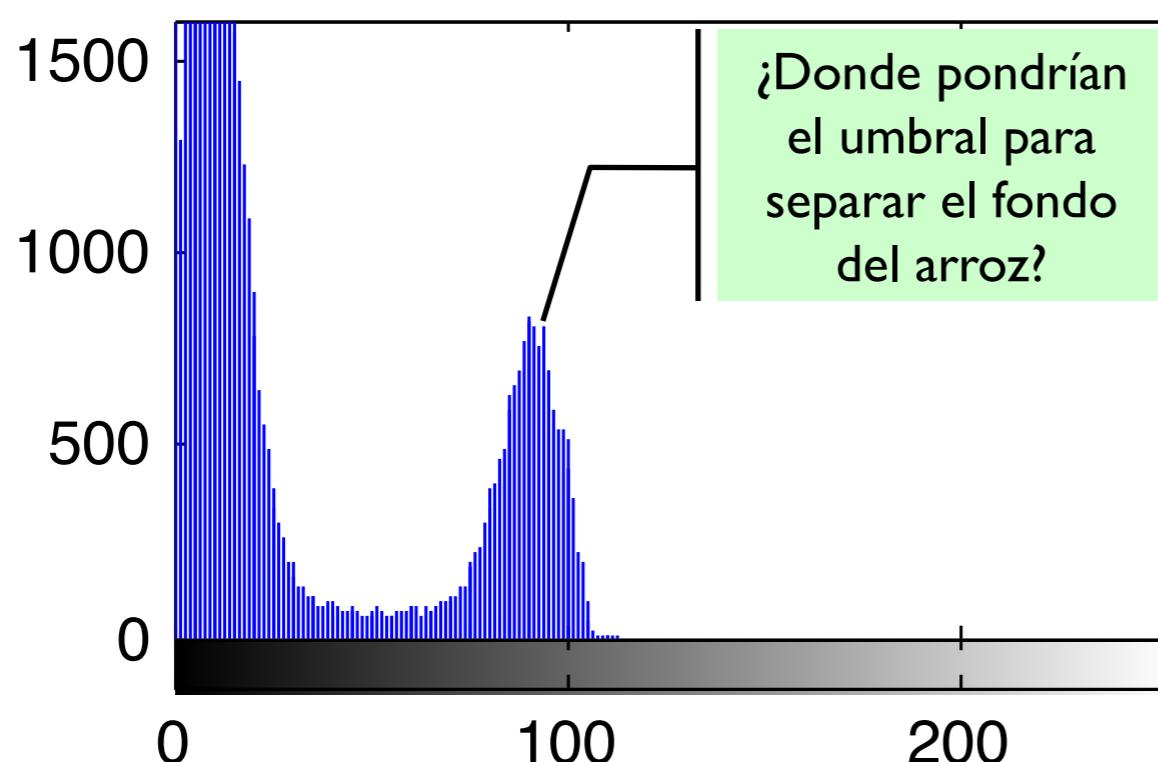
#apertura
tmp      = cv2.erode(img,kernel)
apertura = cv2.dilate(tmp,kernel)

resta    = cv2.subtract(img,apertura)

cv2.imshow('Arroz',resta)
cv2.waitKey(0)
```

## ▶ Aplicación

- Al eliminar el fondo, podemos analizar las características del arroz. Para ello debemos segmentarlo.



Al calcular el histograma podemos visualizar la distribución de los niveles de grises de la imagen

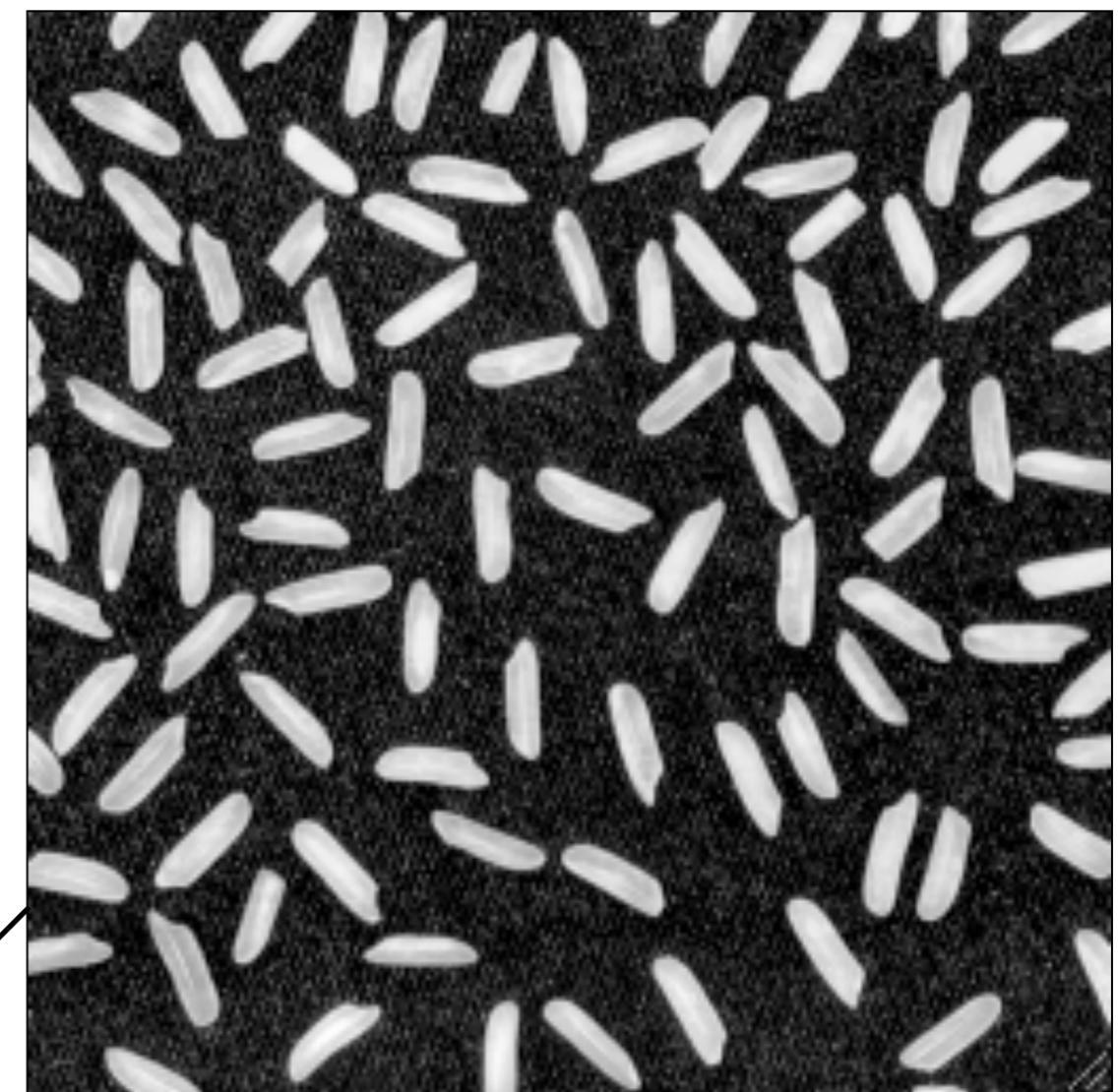
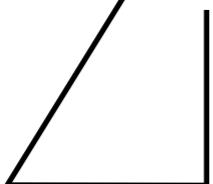


Imagen Top-Hat

## ▶ Aplicación

- Primero seleccionamos un umbral que permita separar el fondo de las estructuras.

```
from skimage.filters import threshold_otsu
from skimage import measure
gray = resta[:, :, 0]
# Buscamos umbral con otsu
thresh = threshold_otsu(gray)
binary = gray > thresh
```



Esta operación binaria es sumamente importante y simple ya que nos permite separar el fondo de las estructuras. (segmentar)

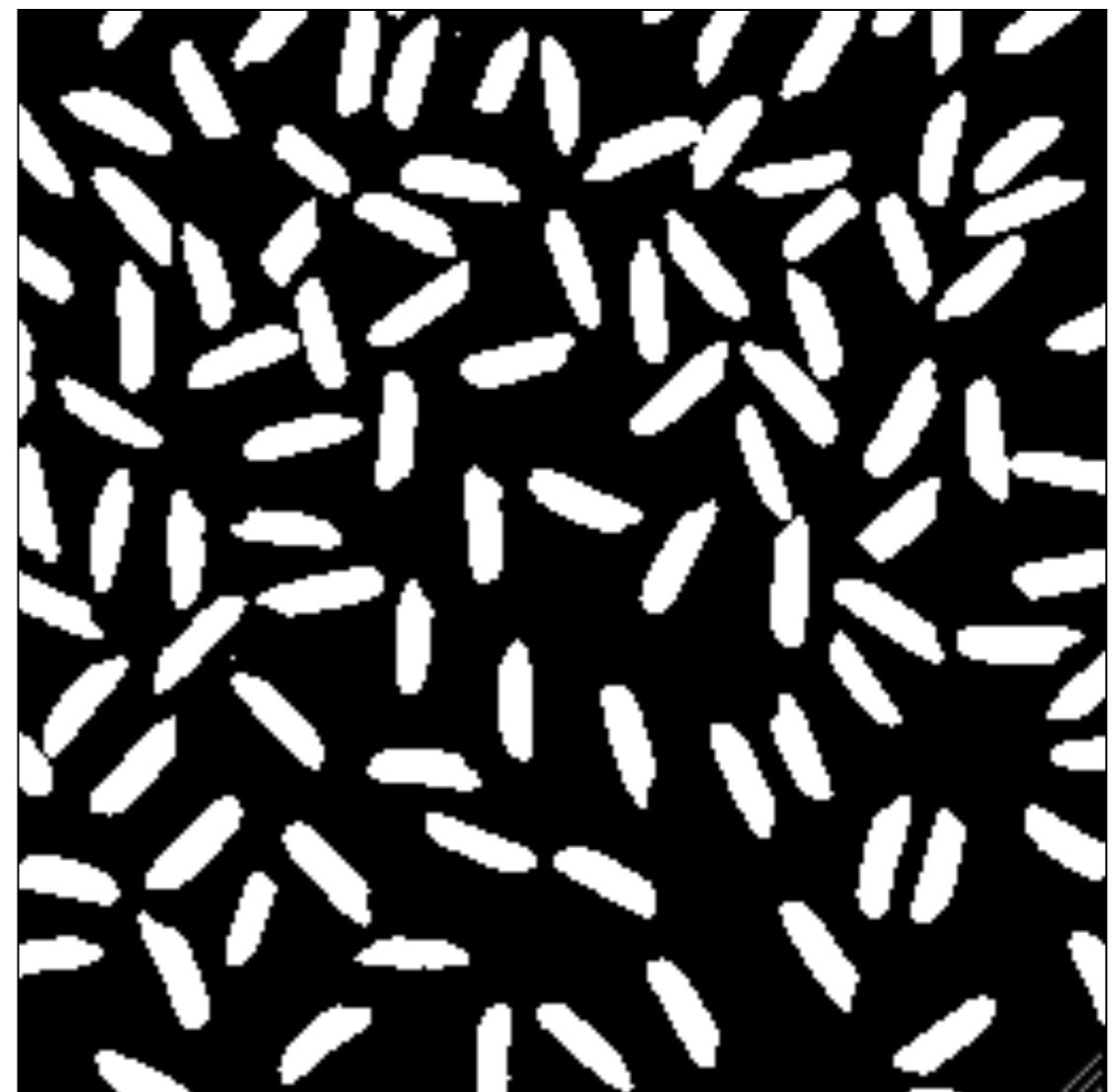
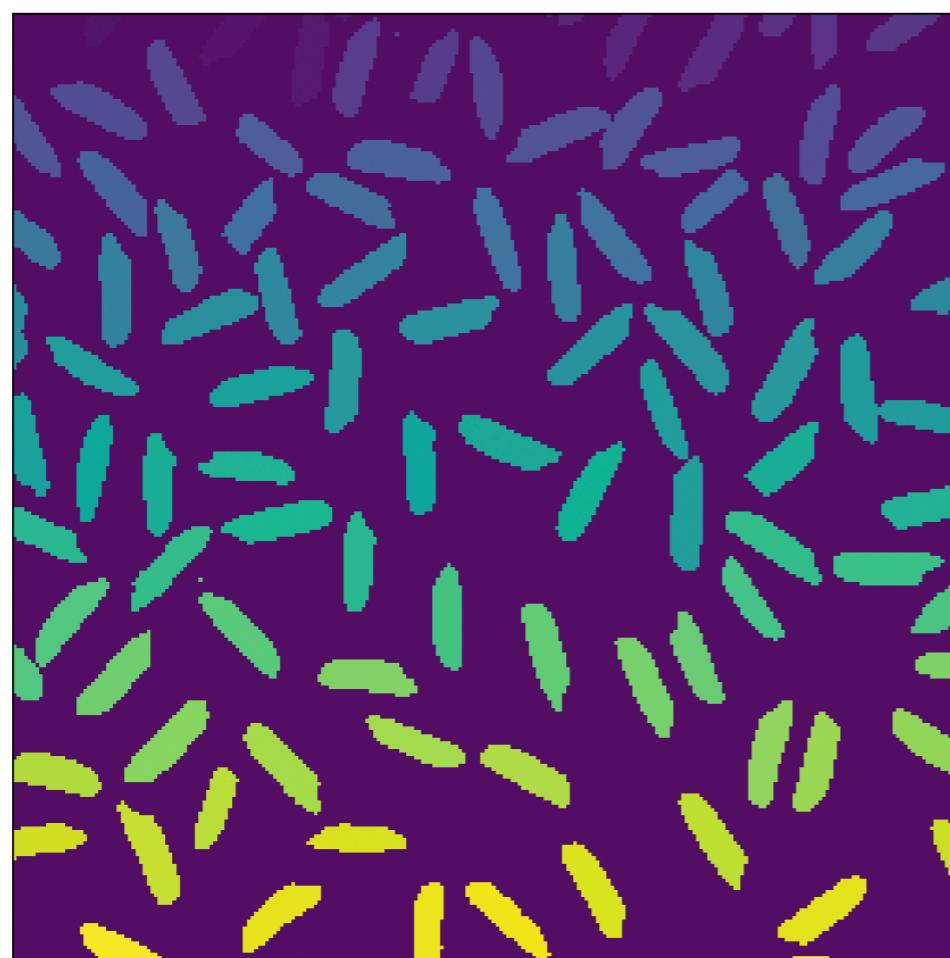


Imagen Binaria

## ▶ Aplicación

- Con la función `measure.label(imagen_binaria)` permite separar cada uno de los objetos binarios que se encuentren en una imagen

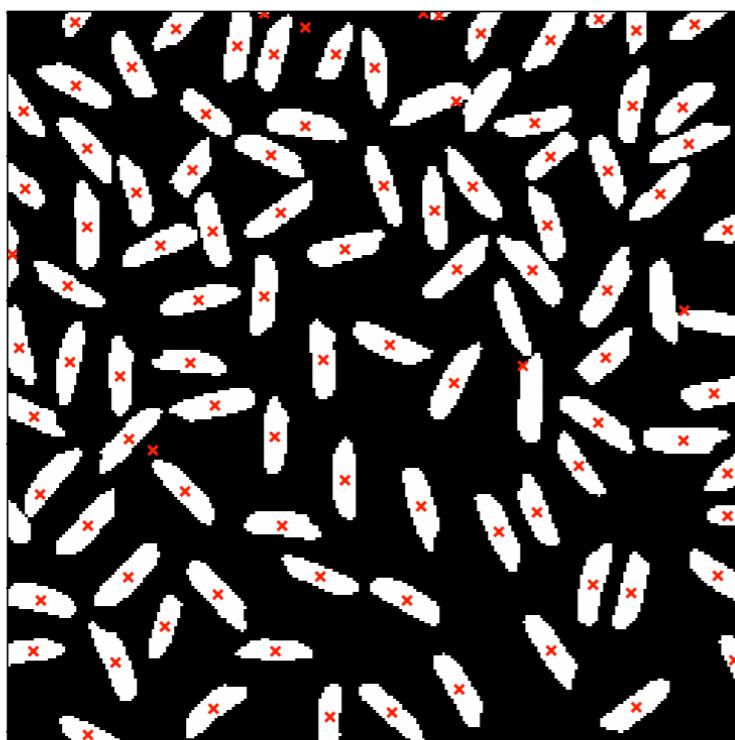
```
# componentes conectados
all_labels = measure.label(binary)
```



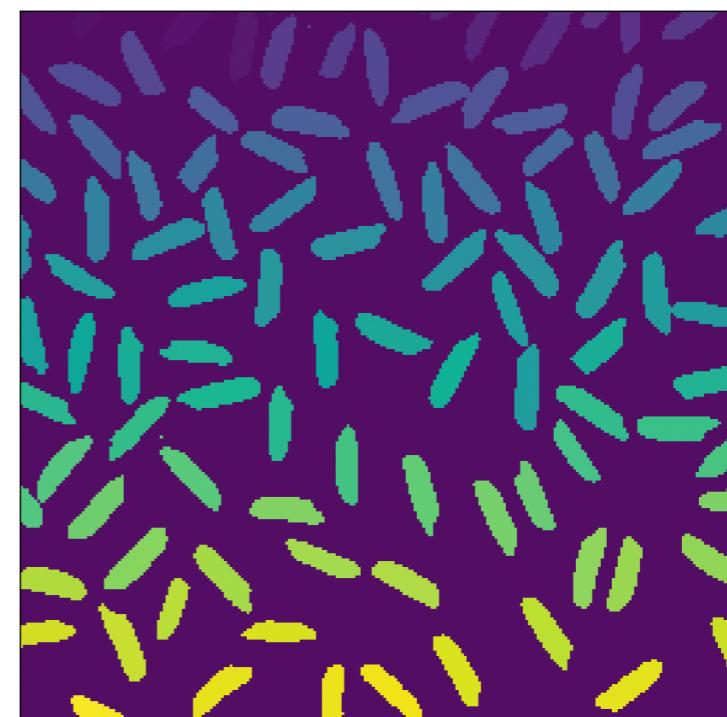
La función `measure` nos permite a través del método `label` enumerar todos los objetos

## ▶ Aplicación

- Con la función `regionprops` extraemos características de los objetos, tales como el área, su centro, perímetro, y muchas más..



regionprops



```
measure.regionprops(label_image=all_labels)
```

Con `regionprops` podemos obtener información estadística de cada región anteriormente etiquetada

## ► Aplicación

- Con la función `regionprops` extraemos características de los objetos, tales como el área, su centro, perímetro, y muchas más..

```
# Threshold data
centroids = []
areas = []
plt.imshow(binary, cmap="gray")

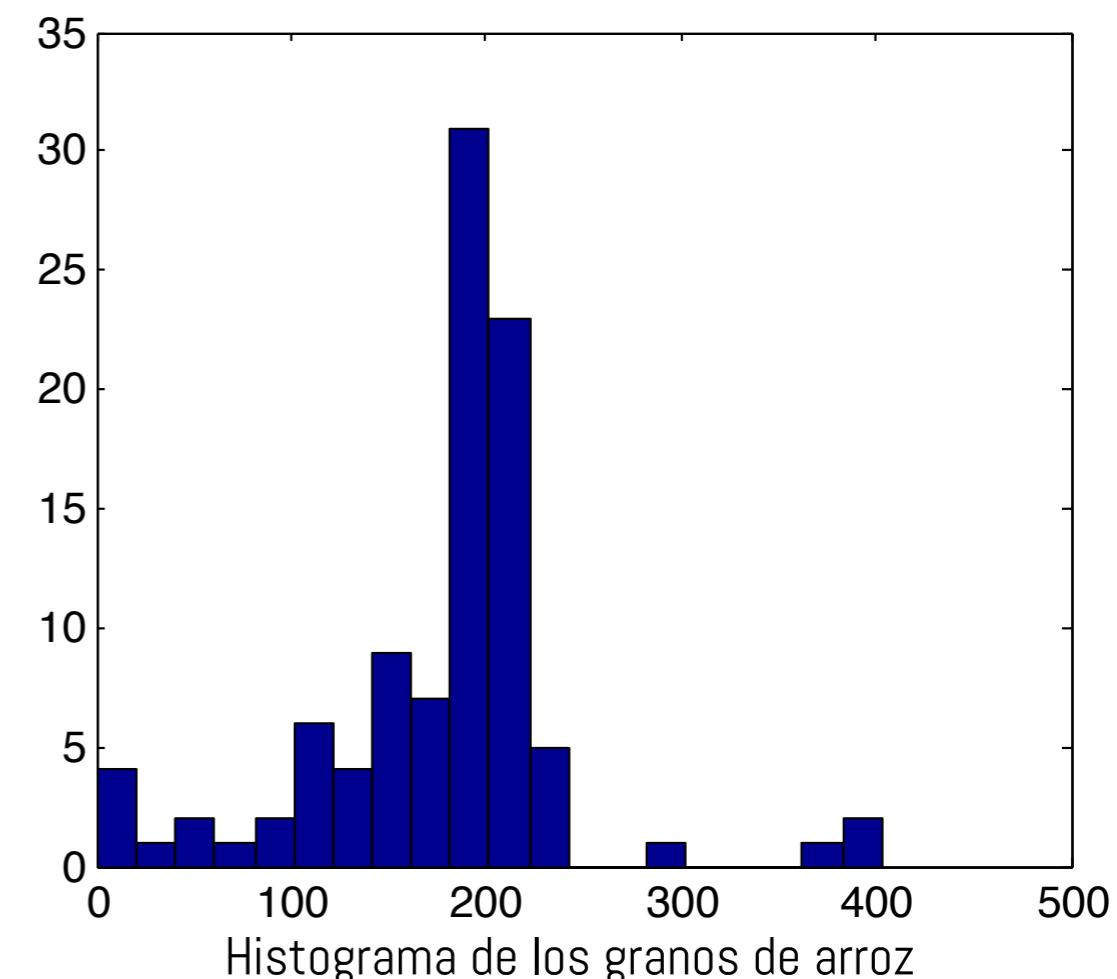
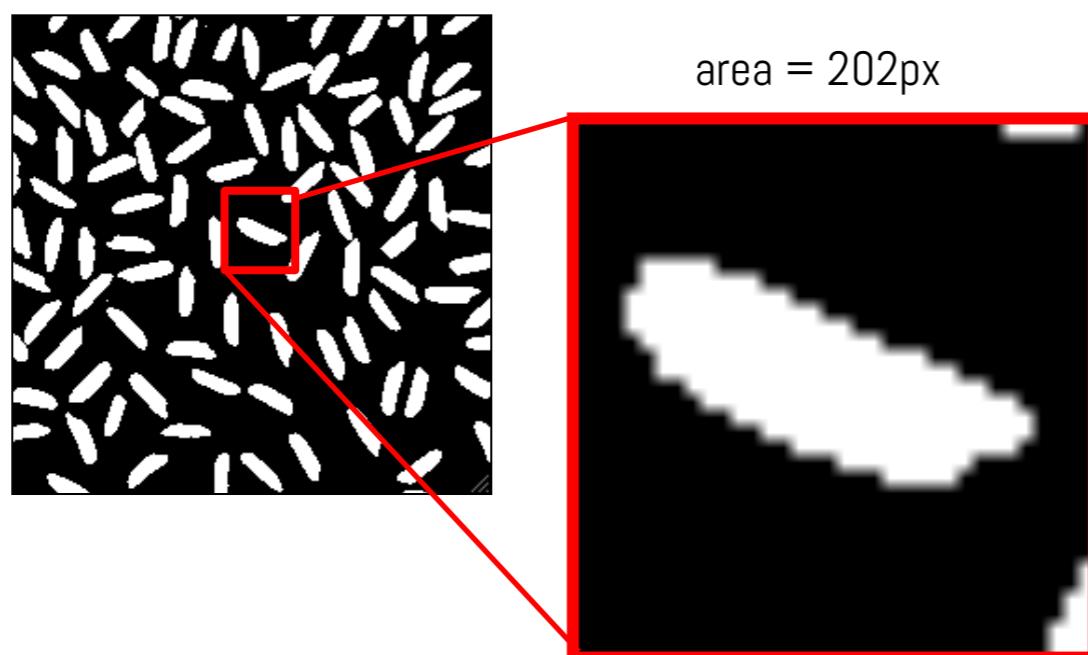
for region in measure.regionprops(label_image=all_labels):

    cx, cy = region.centroid[0], region.centroid[1]
    areas.append(region.area)
    centroids.append((cx, cy))
    plt.scatter(cy, cx, marker="x", color="red", s=20)

plt.show()
```

## ▶ Aplicación

- Al aplicar un histograma sobre el área, de cada grano de arroz podemos observar que existe una distribución entre las áreas segmentadas.



```
plt.figure()  
counts, bins = np.histogram(areas, bins=20)  
plt.hist(bins[:-1], bins, weights=counts)
```

En el arreglo `áreas` se almacena el área de cada arroz de la imagen.

- Resultados
  - Una simple aplicación permite obtener excelentes resultados para evaluar la calidad de un producto



Granos de arroz mayores  
a 200



Granos de arroz mayores a  
200 y menores a 220



Granos de arroz menores  
a 50