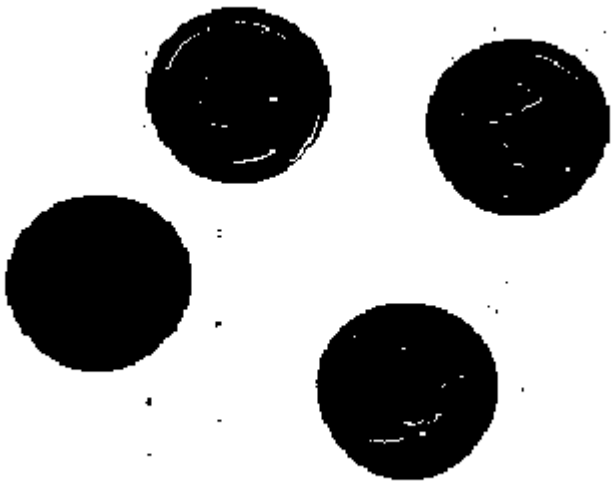
Sesión 1. Segmentación mediante umbralización

1. **Carga en Matlab las imágenes 'bacteria.tif' y 'eight.tif'. A partir de la imagen y ayudándote de impixelinfo, trata de obtener visualmente el mejor umbral para cada una de las imágenes. A continuación, muestra el histograma de cada imagen y determina el umbral a partir del histograma. Umbraliza las imágenes. ¿Has obtenido un umbral mejor que el obtenido visualmente? ¿De qué manera ayuda el histograma a obtener un mejor umbral?**

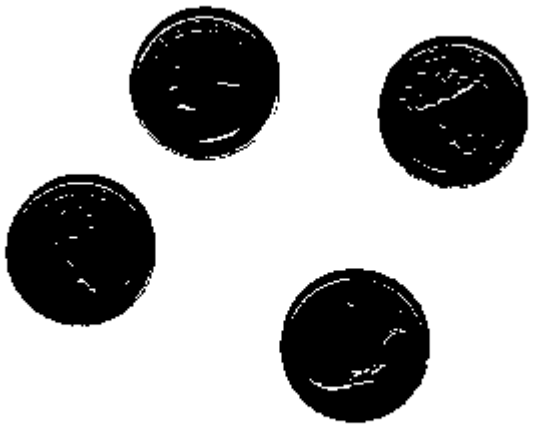
```
%Para obtener visualmente el umbral, tenemos que averiguar las intensidades
más oscuras en los fondos, y las intensidades más claras en las bacterias y
monedas, respectivamente. Luego, calcularemos el valor medio para realizar una
medición aproximada del umbral. Para el histograma, el cálculo es directo

%Bacteria
>> bacteria = imread('ivc_practica3_imagenes/bacteria.tif');
>> imshow(bacteria);
>> impixelinfo;
%Umbral fondo: 105
%Umbral bacteria: 97
%Por lo tanto, el umbral medio -> T = (105+97)/2 = 101
>> imhist(bacteria);
%Umbral histograma: 102

>> level1 = ((101*100)/255)/100;
>> level2 = ((102*100)/255)/100;
>> bw1 = im2bw(bacteria, level1);
>> bw2 = im2bw(bacteria, level2);
>> imshow(bw1);
```

```
>> imshow(bw2);
```

```
%Bacteria
>> eight = imread('ivc_practica3_imagenes/eight.tif');
>> imshow(eight);
>> impixelinfo;
%Umbral fondo: 222
%Umbral bacteria: 198
%Por lo tanto, el umbral medio -> T = (222+198)/2 = 210
>> imhist(eight);
%Umbral histograma: 171
%Ya que el intervalo de intendidades es muy extendido en el histograma,
tomamos un valor intermedio aproximado

>> level1 = ((210*100)/255)/100;
>> level2 = ((171*100)/255)/100;
>> bw1 = im2bw(bacteria, level1);
>> bw2 = im2bw(bacteria, level2);
>> imshow(bw1);
```



```
>> imshow(bw2);
```



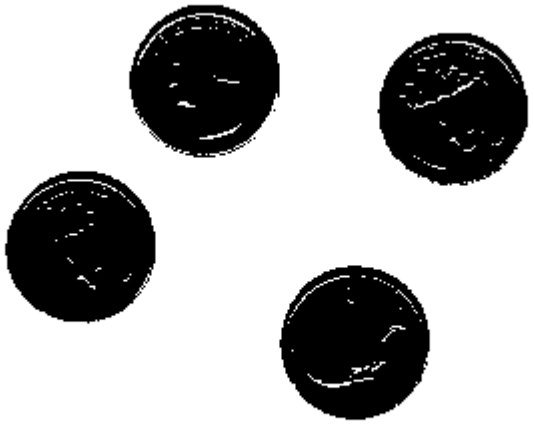
%Los resultados con el histograma han sido mejores, principalmente con las monedas, ya que permite visualizar el reparto de intensidades con los píxeles de la imagen y, por lo tanto, obtener un mejor umbral, para eliminar brillos, como en dicho caso.

2. Utiliza ahora el método de Otsu para obtener el umbral óptimo. ¿Coincide el umbral obtenido con el método de Otsu con el obtenido en el apartado anterior visualmente o mediante el histograma?

```
>> level1 = graythresh(bacteria);
>> bw1 = imbinarize(bacteria, level1);
>> level2 = graythresh(eight);
>> bw2 = imbinarize(eight, level2);
>> imshow(bw1);
```

%En el caso de la bacteria, se asemeja más a la imagen anterior cuyo umbral fue hallado visualmente, aunque presenta diferencias más o menos apreciables en ciertos píxeles.

```
>> imshow(bw2);
```

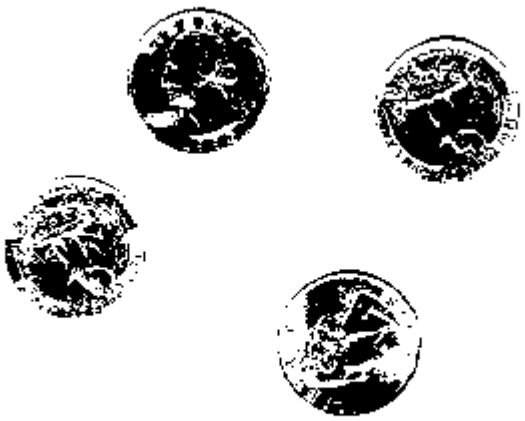


```
    %En el caso de la moneda se asemeja muchos más a la imagen del histograma que
a la imagen hallada visualmente.
```

3. **Escribe una función en Matlab llamada globalthresh que implemente el algoritmo básico de umbralización global. La función aceptará tres parámetros: la imagen original, el umbral inicial, T_0, y el error máximo de umbral, ΔT. La función devolverá la imagen umbralizada y el umbral final. Pruébalo luego sobre las siguientes imágenes: 'eight.tif' y 'bacteria.tif'. Prueba en la imagen 'bacteria.tif' con varios umbrales iniciales, ¿cómo afecta la elección del umbral inicial en la obtención del umbral final?**

```
>> img1 = imread('ivc_practica3_imagenes/bacteria.tif');
>> [imagen1,umbral1] = globalthresh(img1, 0.4, 0.9);
>> umbral1
>>

umbral1 =

      0.4000

>> imagen1
>>
```

```
>> img2 = imread('ivc_practica3_imagenes/eight.tif');
>> [imagen2,umbral2] = globalthresh(img2, 0.4, 0.9);
>> umbral2
>>

umbral2 =

      0.4000

>> imagen2
>>
```



Sesión 2. Descriptores para la detección de formas de regiones.

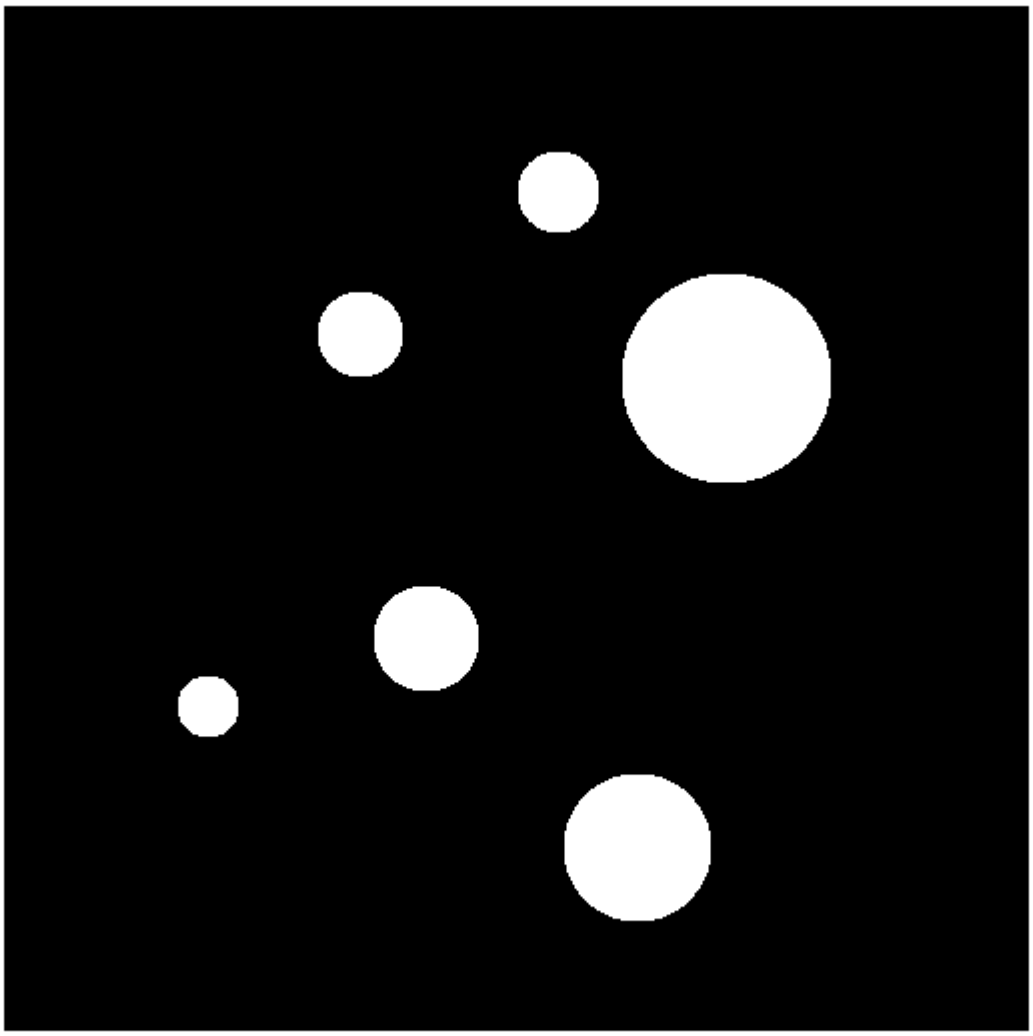
4. **Implementa una función en código Matlab llamada 'detectarForma.m' que produzca como resultado una imagen segmentada que sólo muestre los objetos con la forma que se indicará como parámetro. La estructura de la función será la siguiente:**

   function [numforma,imgd]= detectarForma(imgo,forma)

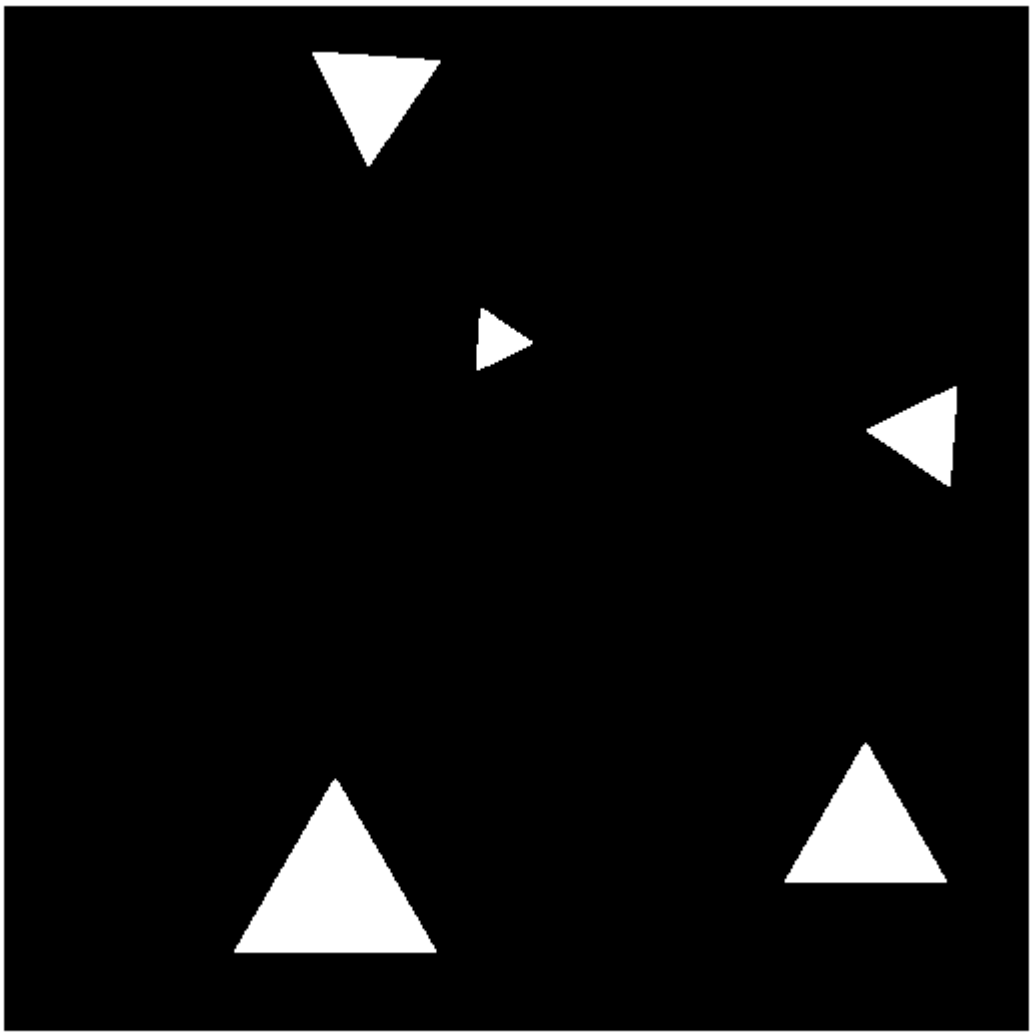
   donde 'imgo' es la imagen de entrada (ya leída en Matlab previamente con imread) y 'forma' identifica el tipo de forma que se quiere detectar como objeto ('1' para círculo, '2' para triángulos, '3' para cuadrados y '4' para rectángulos). Además, la función debe almacenar en la variable 'numforma', el número de objetos con dicha 'forma'. Esta variable será devuelta con la función junto con una imagen binaria 'imgd' en la que sólo aparezcan representados los objetos con dicha forma. Se permite suponer que la imagen de entrada a la función será una imagen en color como la imagen 'formas.tif' que se adjunta a este informe, con lo que primero habrá que transformarla a escala de grises para poder trabajar con ella.

   NOTA: para trabajar con los objetos segmentados en la imagen es conveniente que el fondo sea negro y los objetos blancos.

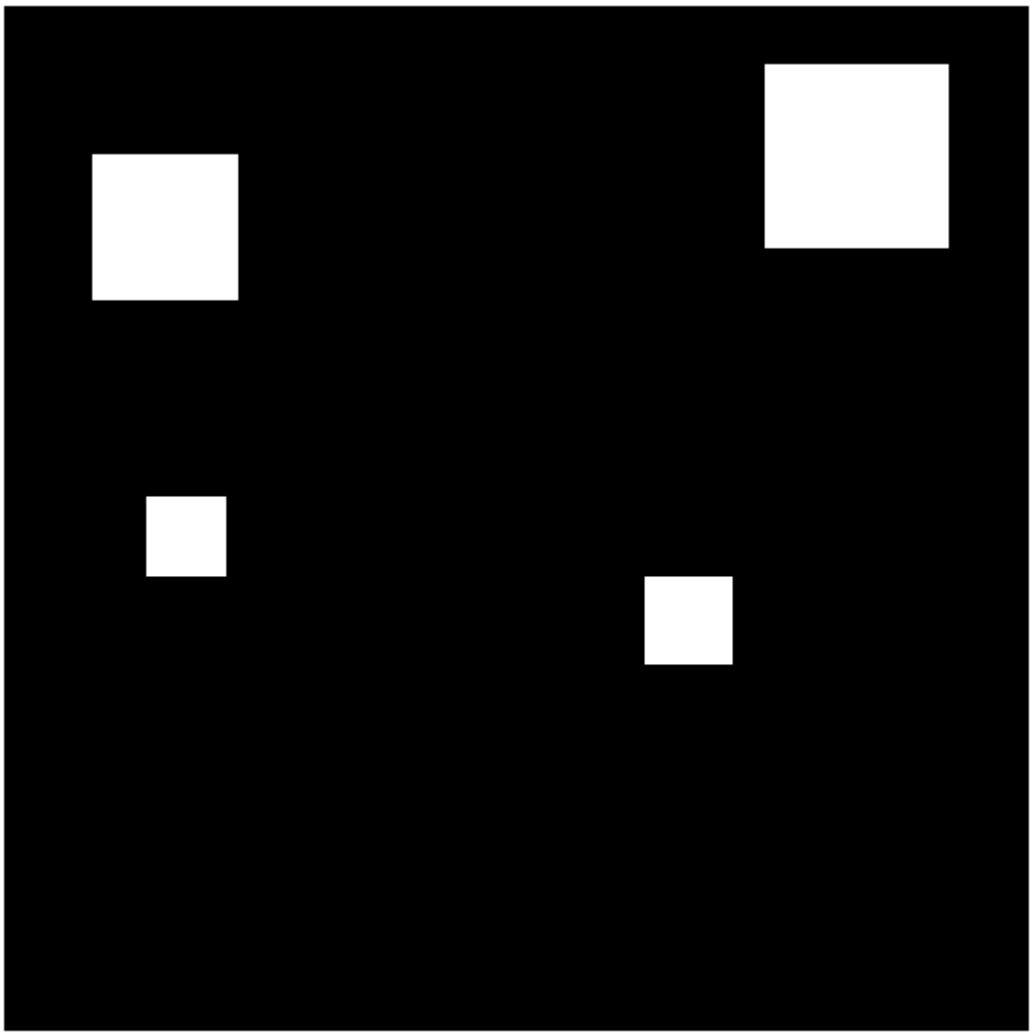
```
>> img = imread('ivc_practica3_imagenes/formas.tif');
>> [numforma,imgd] = detectarForma(img, 1);
>> numforma

numforma =

       6

>> imshow(imgd);
>>
```



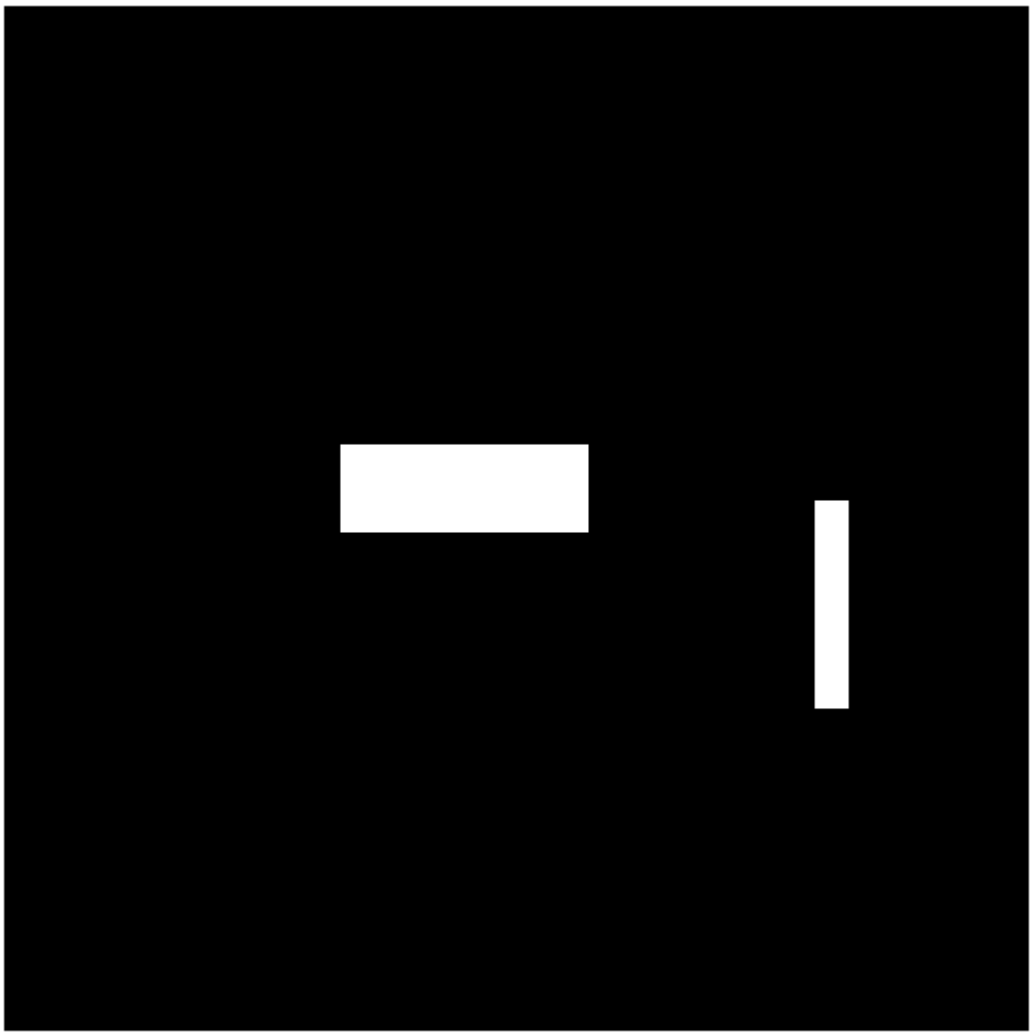
```
>> [numforma,imgd] = detectarForma(img, 2);
>> numforma

numforma =

      5

>> imshow(imgd);
>>
```



```
>> [numforma,imgd] = detectarForma(img, 3);
>> numforma

numforma =

       4

>> imshow(imgd);
>>
```

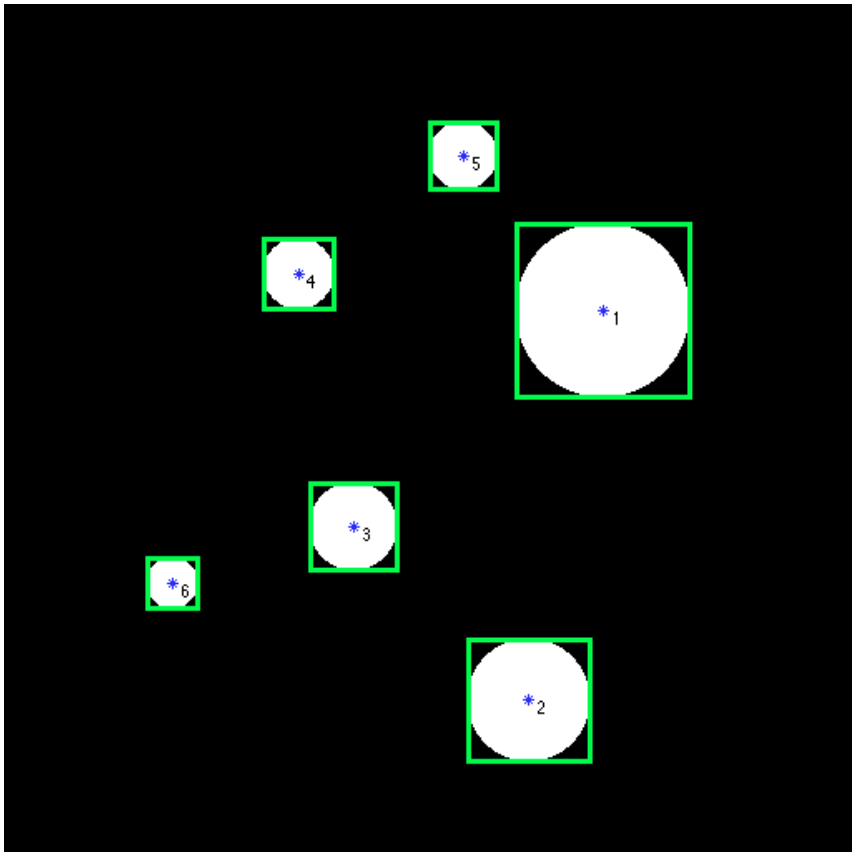


```
>> [numforma,imgd] = detectarForma(img, 4);
>> numforma

numforma =

      2

>> imshow(imgd);
>>
```

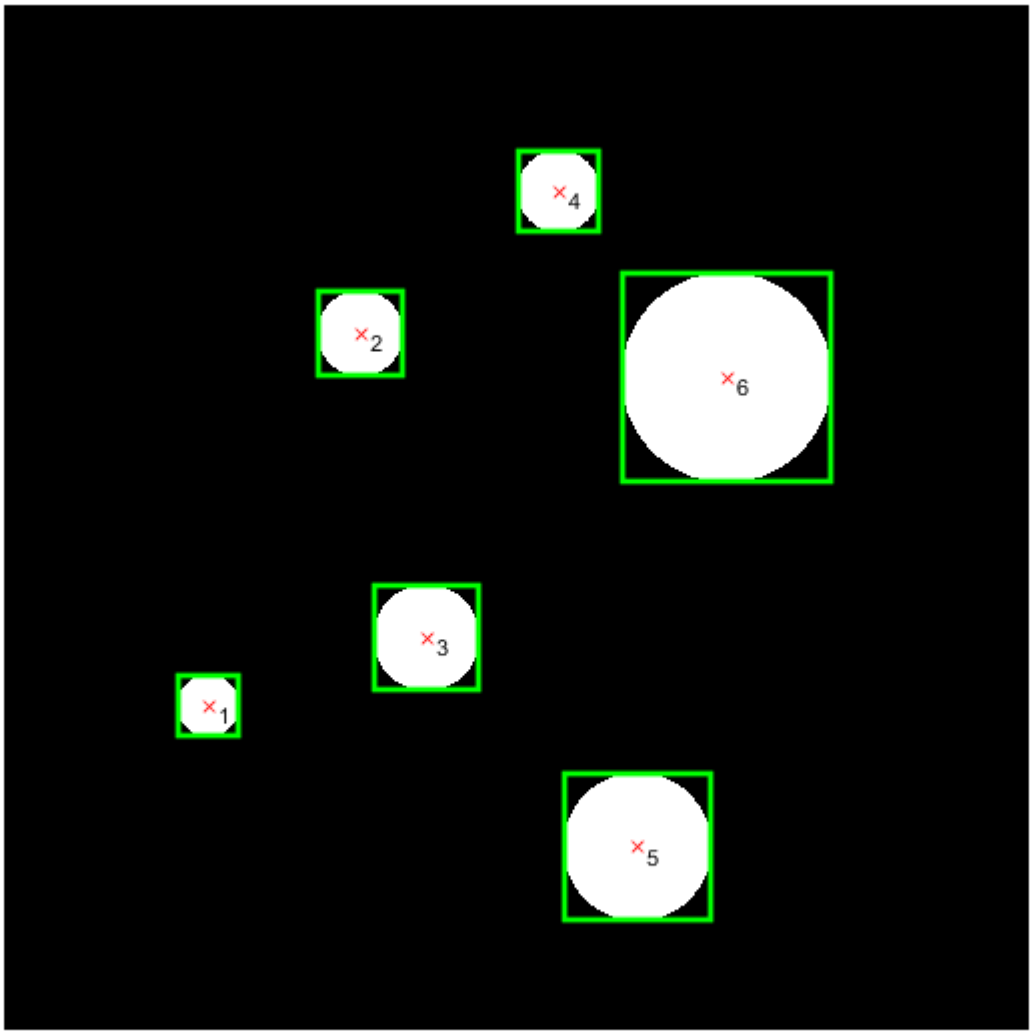


5. **A partir de la función implementada en el apartado anterior, utilizad la función regionprops para obtener la posición del centro de gravedad de cada objeto encontrado. Mostrad una cruz roja en cada centro de gravedad de los objetos encontrados en la imagen utilizando la función plot. Representad mediante un rectángulo verde la caja que encierra cada uno de los objetos encontrados en la imagen (utilizad la función rectangle). Escribid sobre cada objeto el orden en función del área, de forma que el objeto con la forma indicada en la función del ejercicio 1 de mayor tamaño tenga el valor 1, el siguiente en tamaño el 2 y así sucesivamente.**

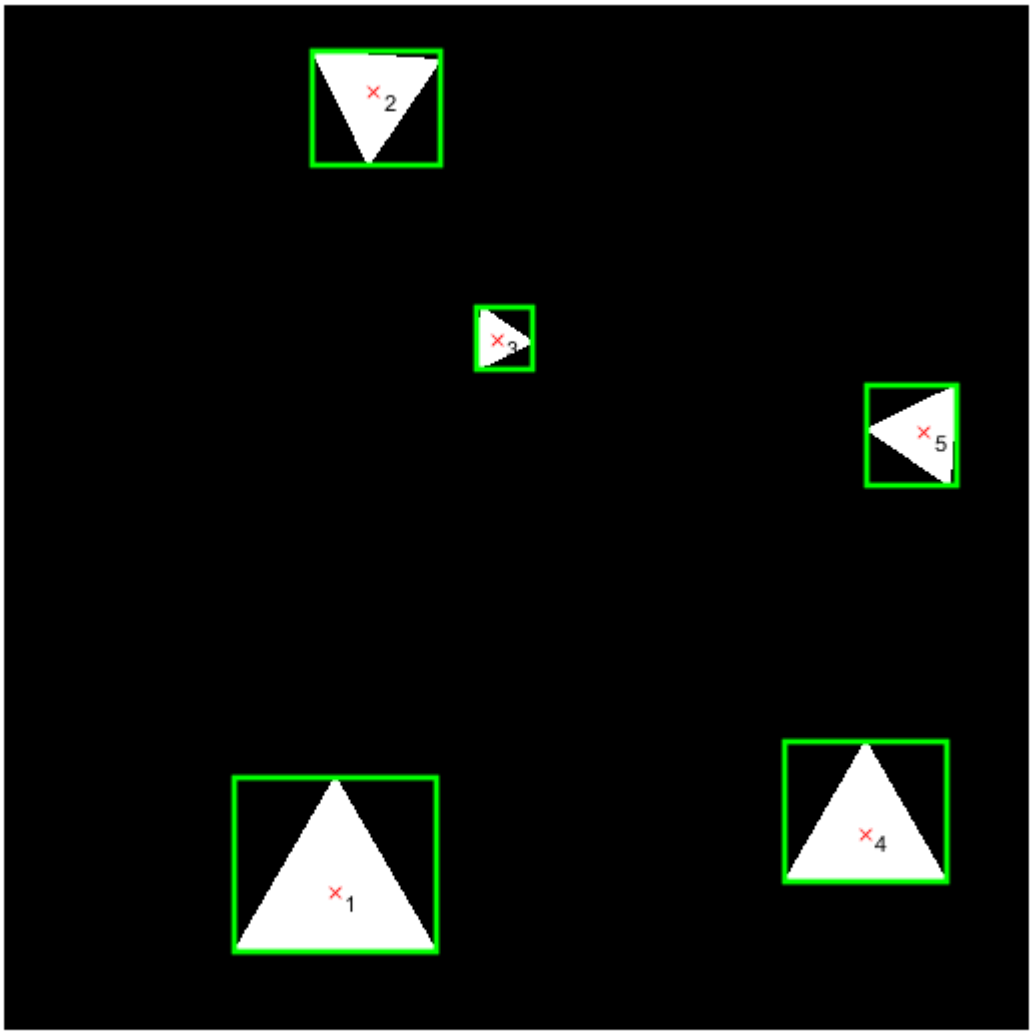
   NOTA: Utilizad la función text para escribir sobre la imagen después de haber ejecutado 'hold on'. Con int2str se puede pasar una variable entera a string para poder escribirla con la función text. Una buena opción para escribir el texto es hacerlo en el centro de gravedad más 5 pixels, tal como se muestra en la siguiente figura:



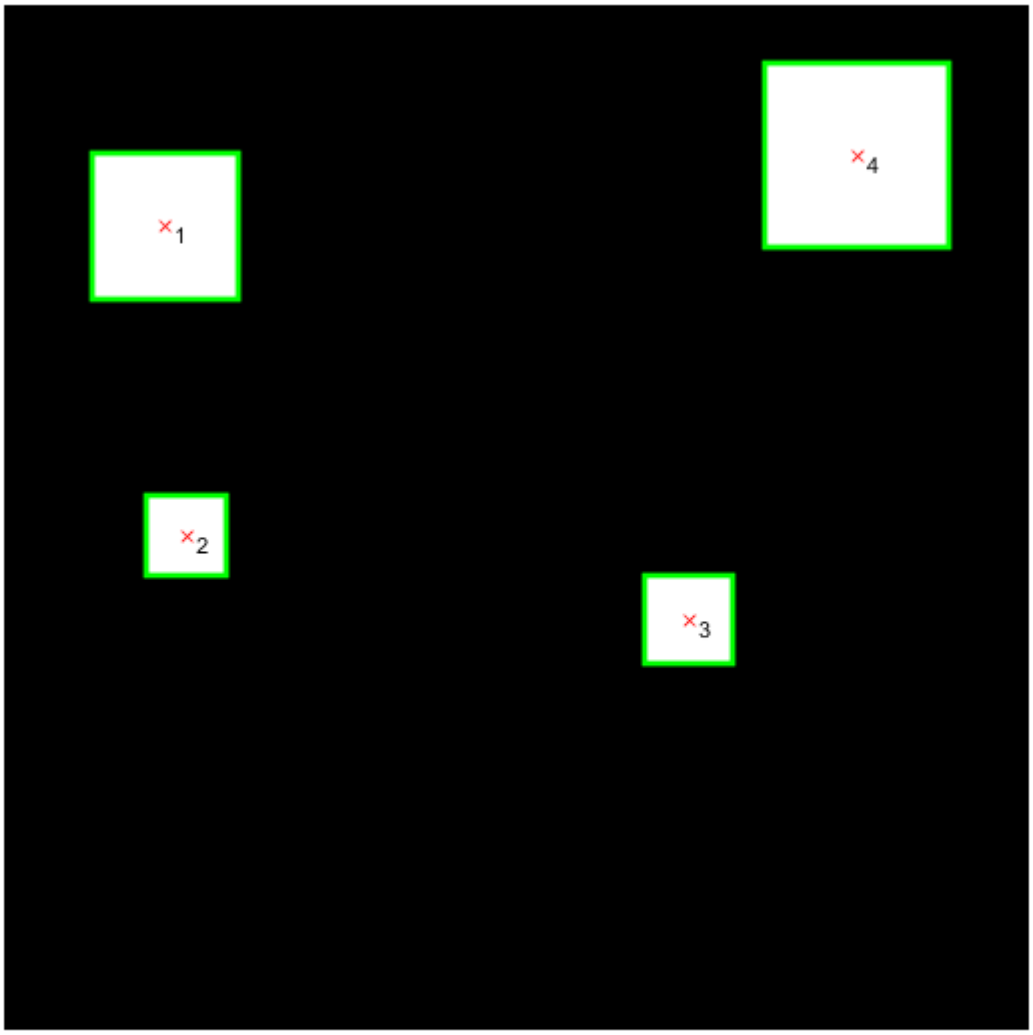

```
>> detectarForma2(img, 1);
```



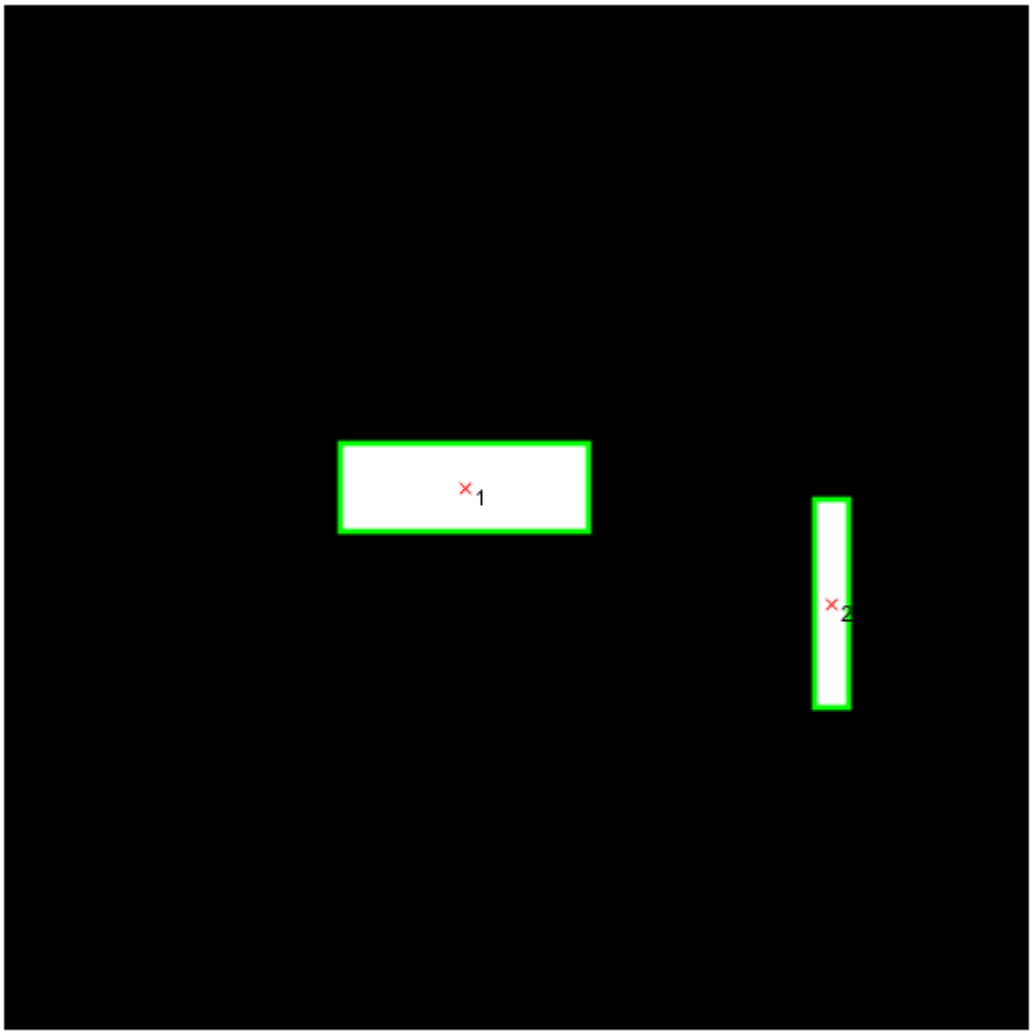

```
>> detectarForma2(img, 2);
```




```
>> detectarForma2(img, 3);
```




```
>> detectarForma2(img, 4);
```


1
2