Universidad de Oviedo

INGENIERÍA DE SISTEMAS Y AUTOMÁTICA

Algoritmos de Visión Artificial con Matlab

Sesión 3. Detección de bordes.

Operaciones binarias

Ignacio Alvarez García

Rafael C. González de los Reyes

Indice

- Estructura del curso
- Detección de bordes
- Binarización por umbral
- Operaciones con imágenes binarizadas
- Ejemplos con Matlab

Indice

- **Estructura del curso**
- Detección de bordes
- Binarización por umbral
- Operaciones con imágenes binarizadas
- Ejemplos con Matlab

Estructura del curso

Sesión 1 (3h)

- Sesión 1.1
  - Introducción a la visión por computador.
  - Elementos de un sistema de visión por computador.
  - Etapas del procesamiento de imágenes.
  - Formatos de almacenamiento de imágenes en memoria y disco.
  - Funcionalidades básicas de Matlab para la manipulación de imágenes.
- Sesión 1.2
  - Preprocesamiento de imágenes: introducción.
  - Mejora de contraste.
  - Zoom e interpolación.
  - Filtrado de ruidos.
  - Ejemplos con Matlab.
- Sesión 1.3
  - Resaltado de bordes.
  - Binarización y segmentación.
  - Operaciones con imágenes binarizadas.
  - Ejemplos con Matlab.

Sesión 2 (2h)

- Sesión 2.1
  - Búsqueda y ajuste de rectas.
  - Obtención de regiones.
  - Descriptores de regiones.
  - Uso de los descriptores.
  - Ejemplos con Matlab.
- Sesión 2.2
  - Calibración de cámaras.
  - Obtención de información 3D
- Sesión 2.3
  - Programación C/C++ con OpenCV
  - Inteligencia Artificial
  - Conclusiones

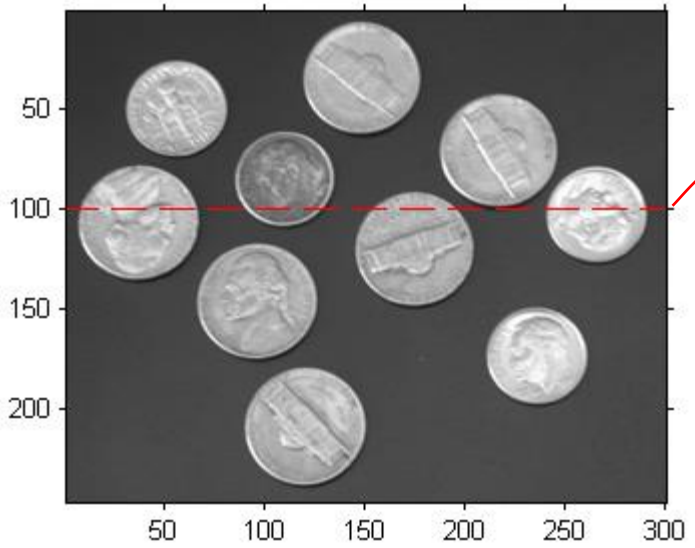
Indice

- Estructura del curso
- **Detección de bordes**
- Binarización por umbral
- Operaciones con imágenes binarizadas
- Ejemplos con Matlab

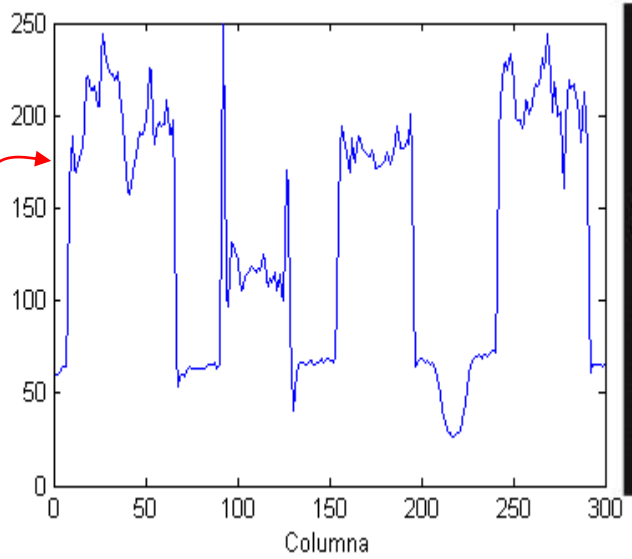
Detección de bordes

- Operación de preprocesamiento que resalta los bordes de los objetos.
- Borde: región donde la luminosidad sufre una variación brusca.
- Resaltado de bordes: generación de una imagen con máximos en las zonas de borde

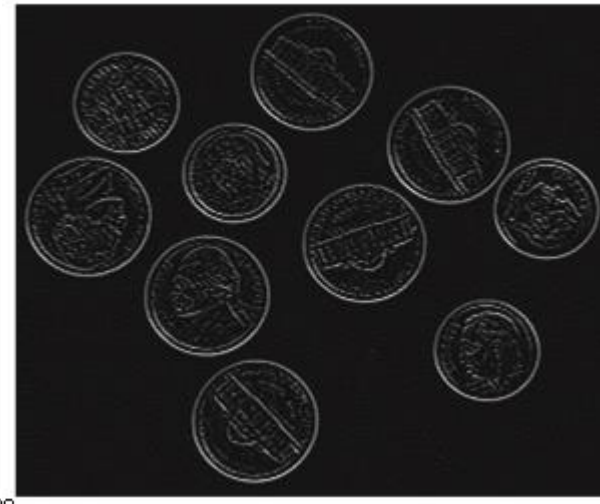
Imagen fuente



Fila 100



Bordes resaltados



Detección de bordes

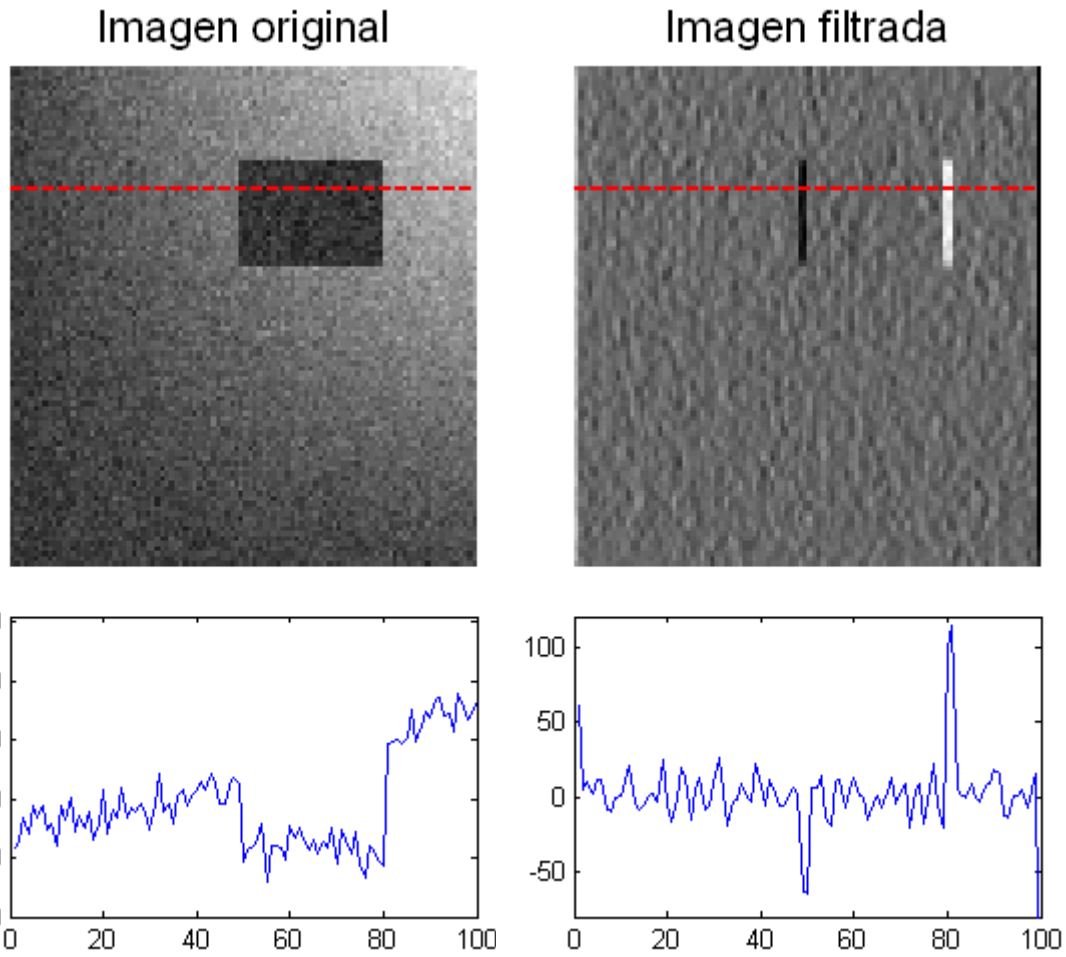
- Los bordes se resaltan mediante aplicación de filtros de convolución pasa-altos:
  - Se produce efecto de resaltado de borde siempre que:
    - Los pesos del kernel son positivos y negativos (a cada lado del punto central)

      Y
    - La suma de todos los pesos sea igual a 0
  - En estos casos, el kernel se comporta como un filtro paso-alto:
    - Elimina las bajas frecuencias (variaciones suaves).
    - Destaca las altas frecuencias (variaciones bruscas y ruido).
  - ¡ OJO ! Esto afecta tanto al ruido (indeseado) como a las características deseadas (ej. los bordes).

Detección de bordes

- Ejemplo: detección de bordes verticales

$$\text{kernel} = \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$



Detección de bordes

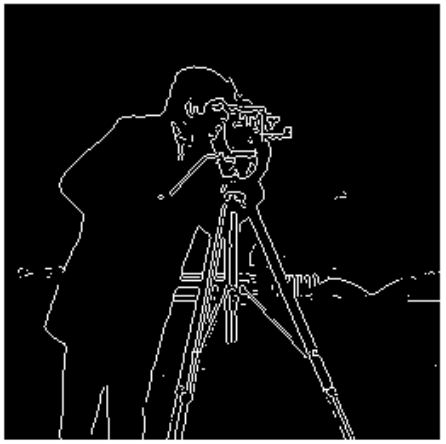
- Kernels básicos para detección de bordes:
  - Roberts: $\text{kernels} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$
  - Prewitt: $\text{kernels} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
  - Sobel: $\text{kernels} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
  - Laplacian of gaussian (log): (detección de pasos por cero) $\text{kernel (3x3)} \begin{bmatrix} 0.4038 & 0.8021 & 0.4038 \\ 0.8021 & -4.8233 & 0.8021 \\ 0.4038 & 0.8021 & 0.4038 \end{bmatrix}$

Detección de bordes

- En la literatura de visión (y en Matlab) se definen conjuntos de operaciones para detección de bordes:
  - (filtrado) + filtro paso-alto + selección automática de umbral
- El método conjunto más utilizado es el de Canny:
  - Filtro de gaussiana (suavizado)
  - Filtro pasa alto (Sobel, Prewitt o Roberts)
  - Supresión de no máximos (en función de la intensidad del resultado y el ángulo del gradiente en cada punto)

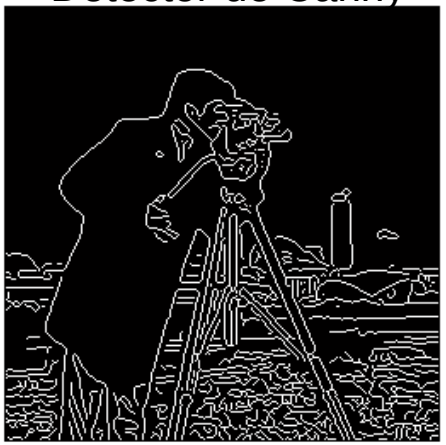
Imagen original

Detector de Sobel



Laplaciana de gaussiana

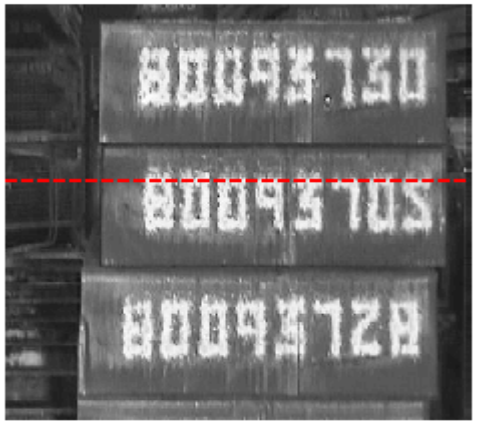
Detector de Canny

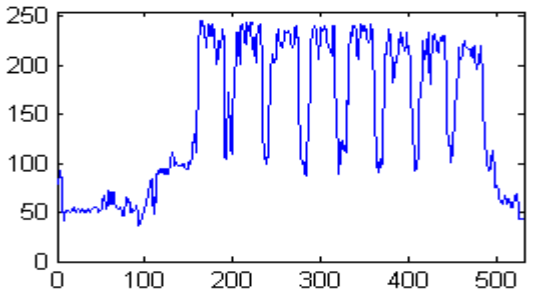


Detección de bordes

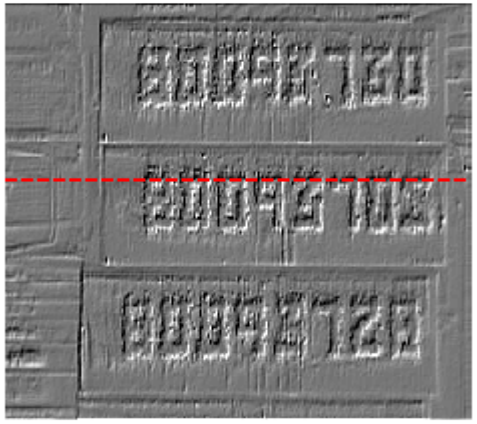
- Imágenes con bordes ‘suavizados’
  - Los bordes no abruptos (suavizados) se producen cuando la transición ocupa más de 2 ó 3 pixels.

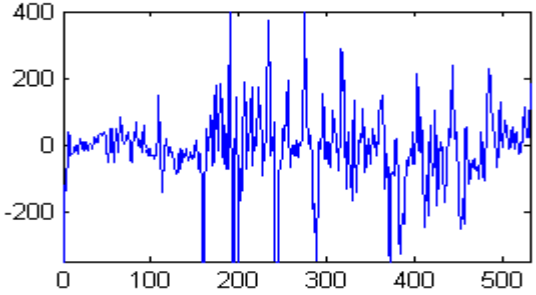
Imagen original





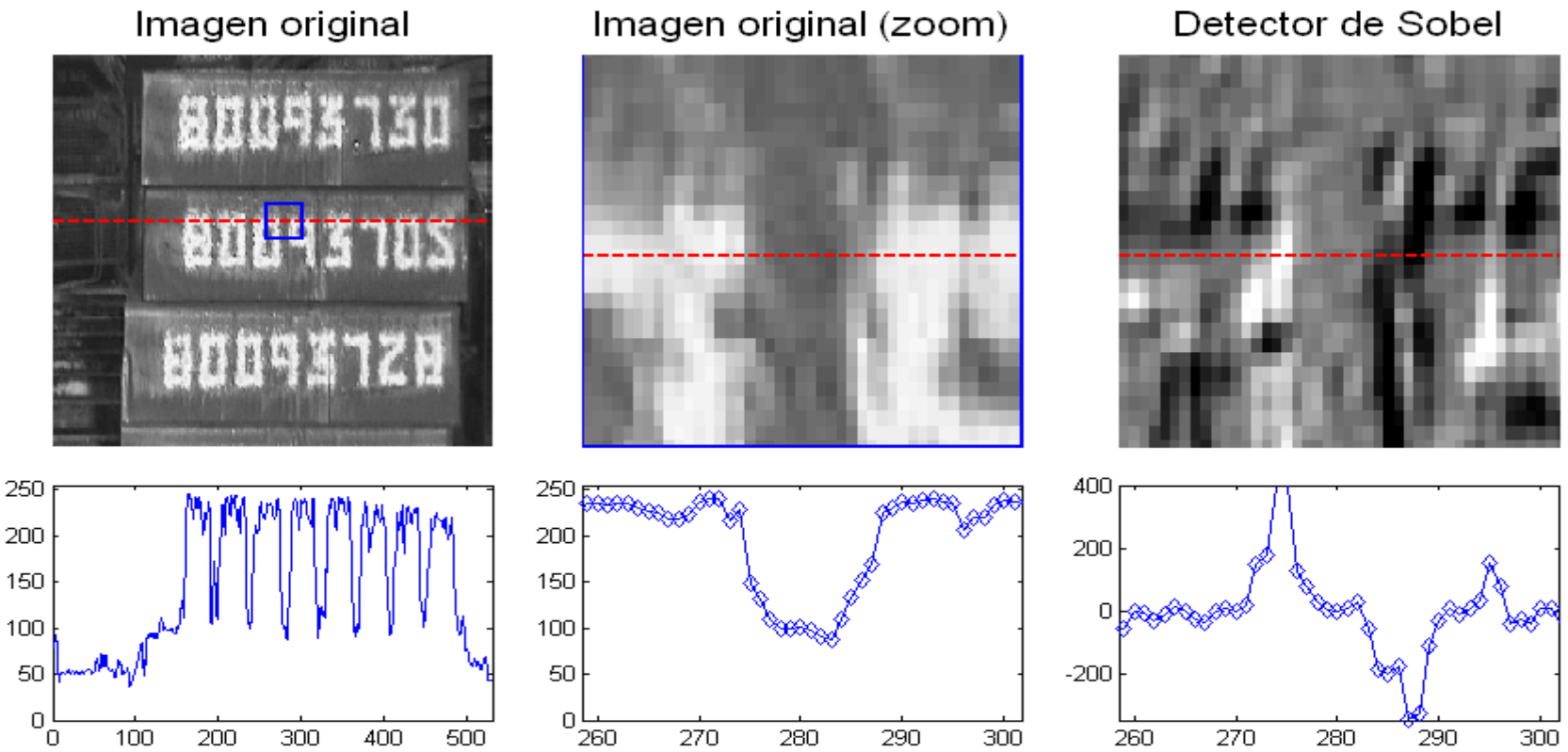
Detector de Sobel





Detección de bordes

- Imágenes con bordes ‘suavizados’
  - El borde detectado puede ser del mismo nivel que el ruido.

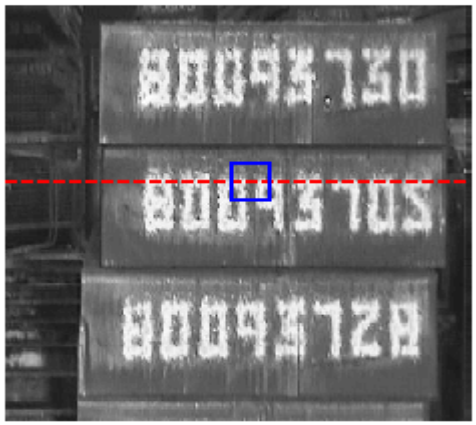


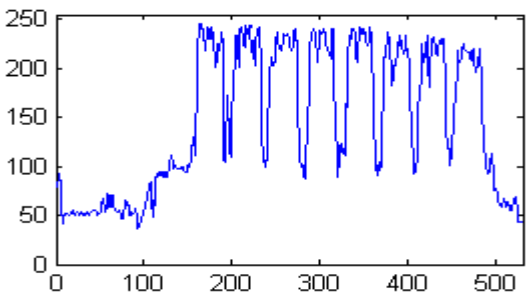
Detección de bordes

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

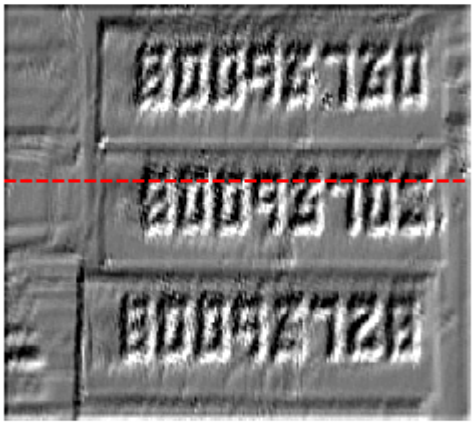
- Imágenes con bordes ‘suavizados’
  - Para resaltar bordes suavizados, se debe incrementar el tamaño del kernel

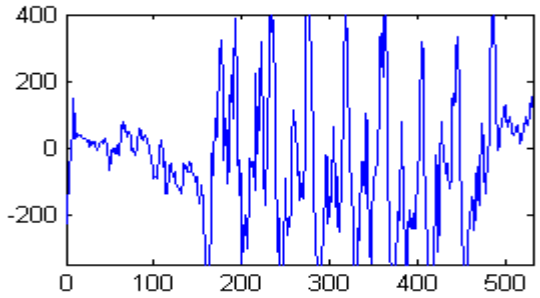
Imagen original



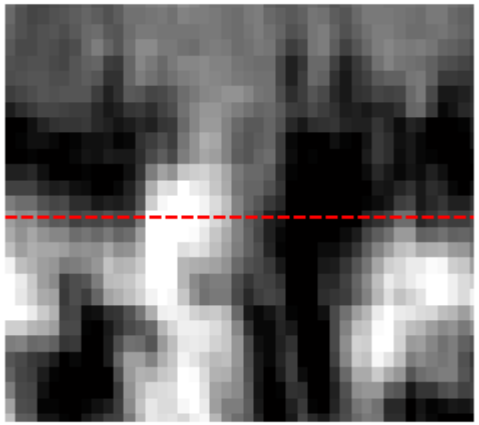


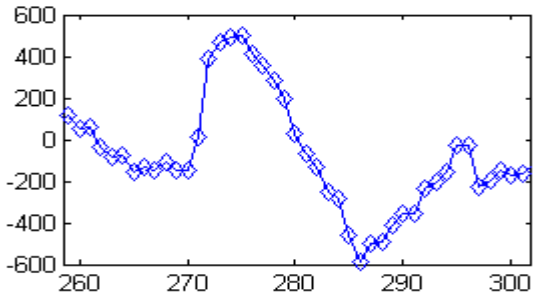
Kernel 11x11





Kernel 11x11





Detección de bordes

- Imágenes con bordes ‘suavizados’
  - Para resaltar bordes suavizados, se debe incrementar el tamaño del kernel

  pero …
  - Se pierden los detalles de menor tamaño.
  - El borde resultante al umbralizar es de mayor grosor.
  - La detección del borde se debe hacer con filtros detectores de máximos y mínimos.

Indice

- Estructura del curso
- Detección de bordes
- **Binarización por umbral**
- Operaciones con imágenes binarizadas
- Ejemplos con Matlab

Binarización por umbral

- Tras aplicar los algoritmos de pre-procesamiento:
  - Binarización de la imagen preprocesada para quedarse sólo con los elementos de interés:
    - Regiones (objetos)
    - Bordes
    - Zonas con máximos o mínimos
    - Zonas con colores o tonos determinados
  - Algoritmos de binarización:
    - Umbralización (pixel a pixel) global
    - Umbralización (pixel a pixel) por regiones
    - Detección de máximos locales (para bordes)
  - Resultado: imagen binaria
    - Pixels a 1: pertenecen a elementos de interés
    - Pixels a 0: no pertenecen a elementos de interés

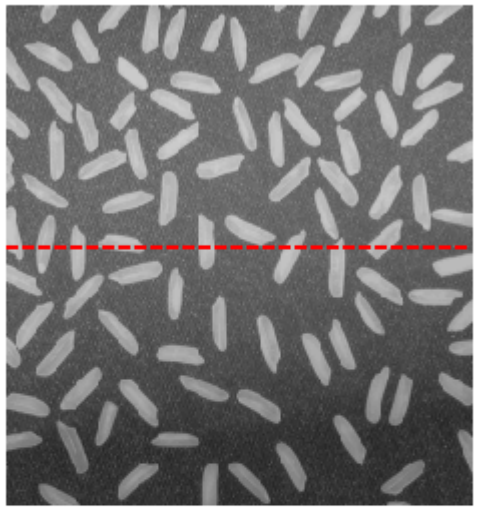
Binarización por umbral

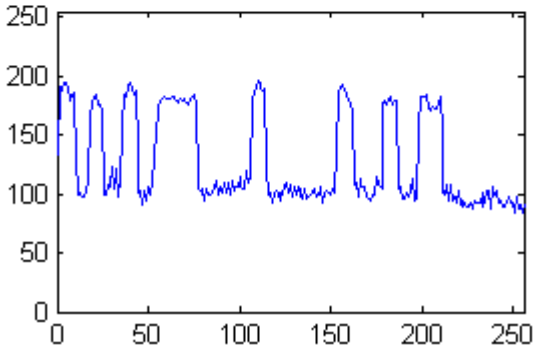
- Binarización por umbral:
  - Selección de los pixels que cumplen una determinada condición (> >= < <= = <>).
  - Configuraciones de selección:
    - Valor fijo conocido a priori.
    - Valor fijo a partir del histograma.
    - Valor variable por regiones.
  - Número de imágenes:
    - Uso de una única imagen preprocesada.
    - Combinación de varias imágenes preprocesadas.
  - Los detectores de contorno de Matlab (edge) devuelven directamente una imagen umbralizada.

Binarización por umbral

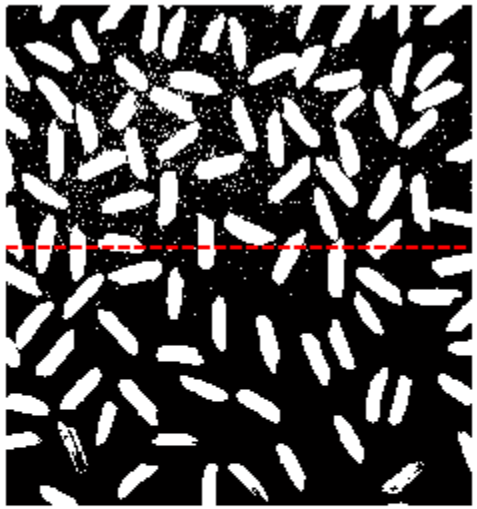
- Ejemplo: binarización de imagen original

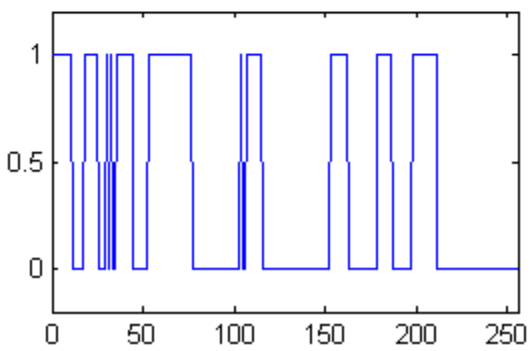
Imagen original



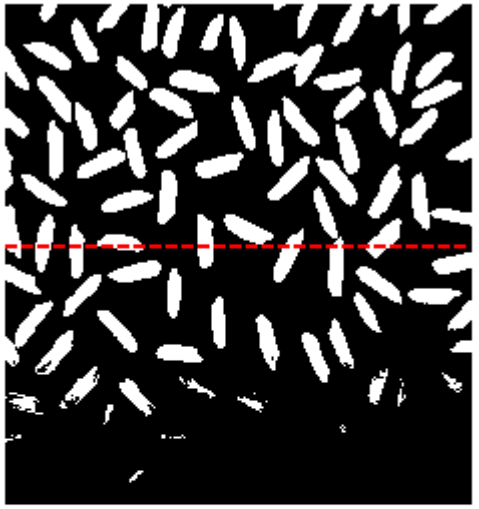


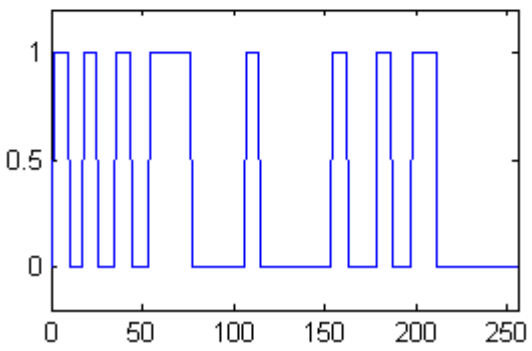
Umbral = 120





Umbral = 150

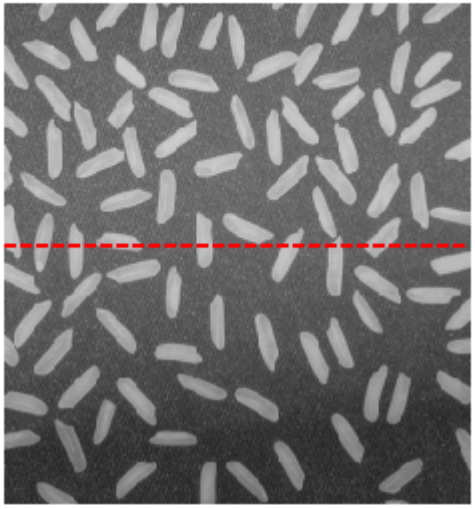


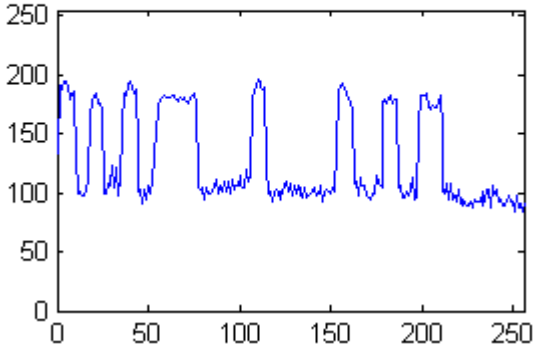


Binarización por umbral

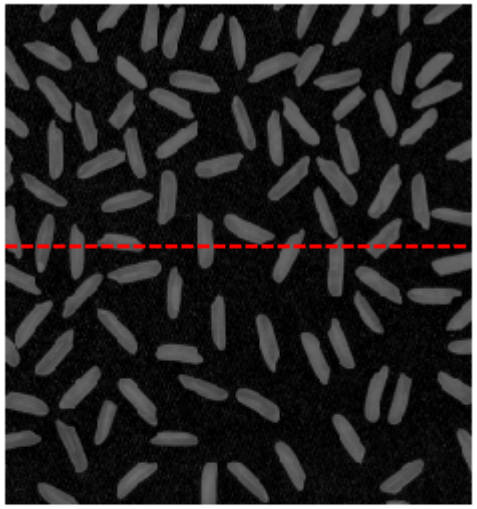
- Ejemplo: preprocesamiento tophat y binarización a partir del histograma

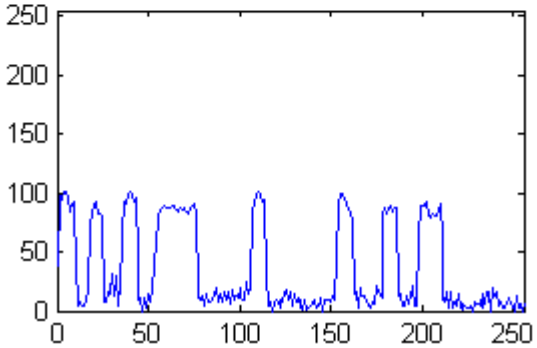
Imagen original



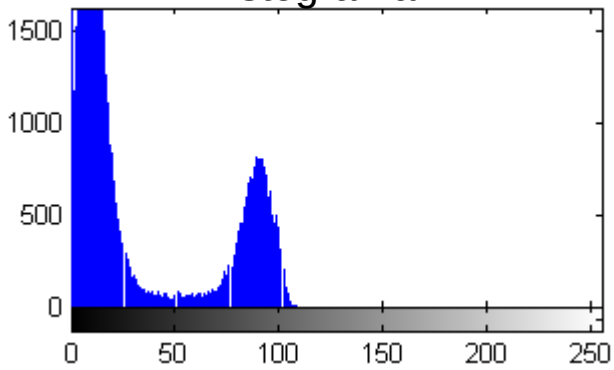


Top hat (disk 10)

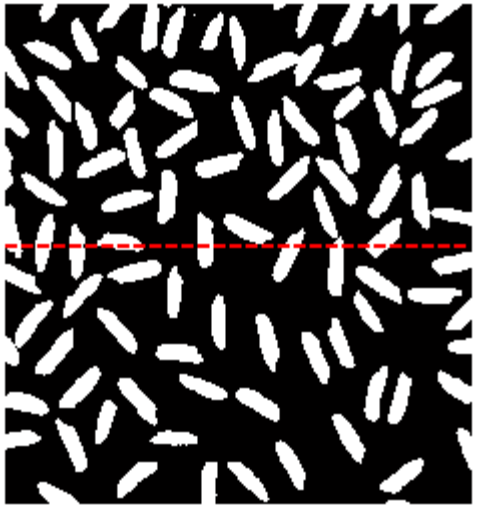




Histograma

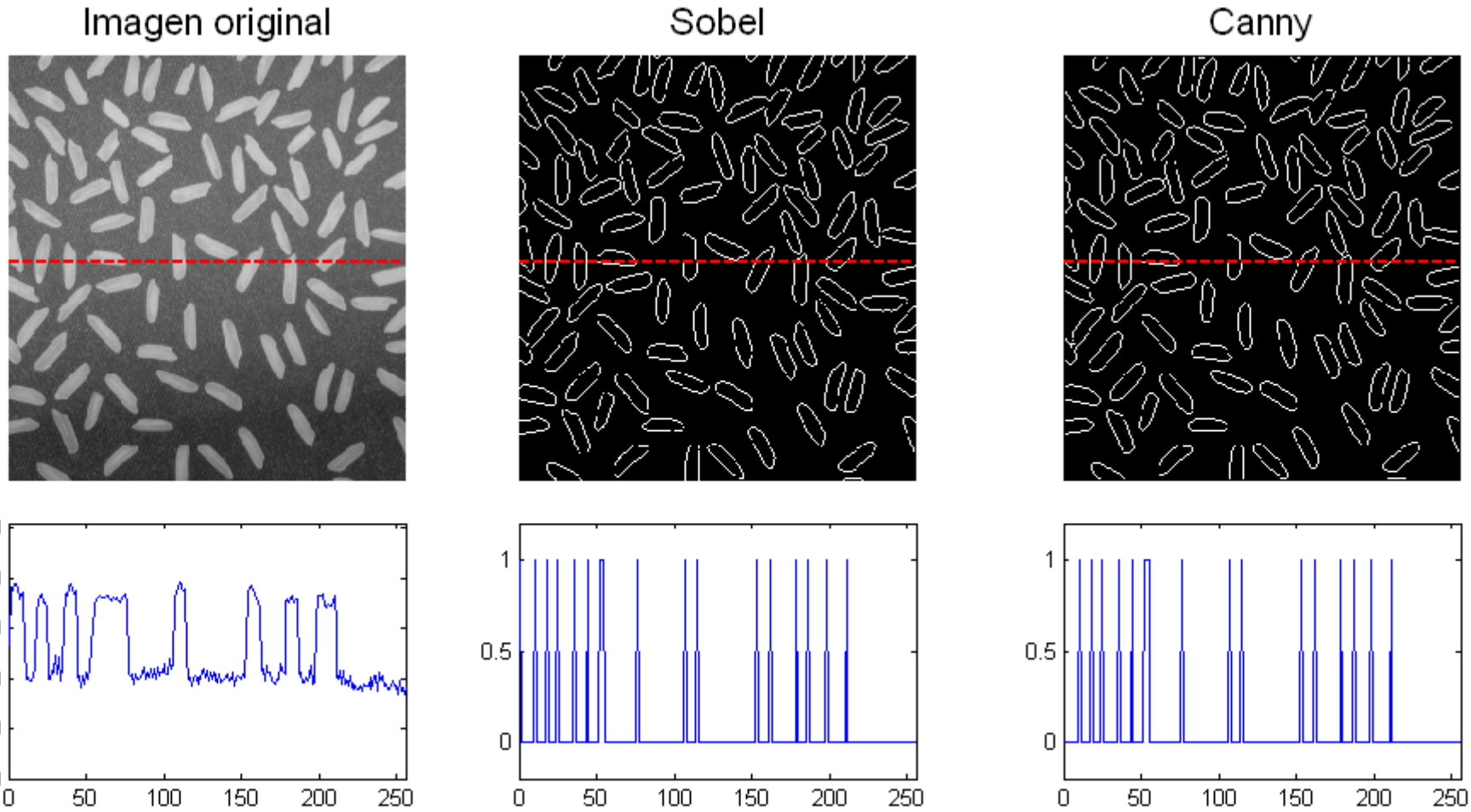


Binarizada (umbral 50)



Binarización por umbral

- Ejemplo: detectores de borde



Binarización por umbral

- Método de Otsu: selección del umbral que minimiza la varianza de las dos clases en que se divide la imagen

Indice

- Estructura del curso
- Detección de bordes
- Binarización por umbral
- **Operaciones con imágenes binarizadas**
- Ejemplos con Matlab

Operaciones con imag. binarias

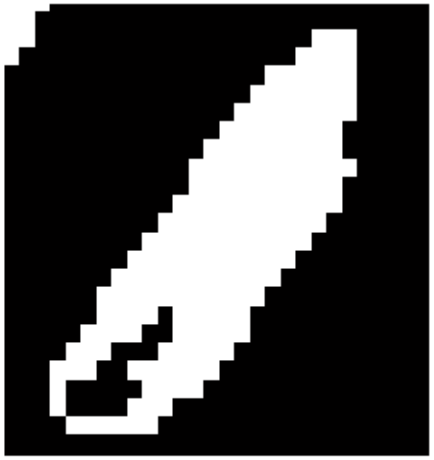
- El resultado de la binarización **casi nunca** es perfecto:
  - Pixels aislados.
  - Huecos.
  - Falta de continuidad en contornos.
  - Contornos gruesos.
  - Objetos unidos incorrectamente.

Pixels a 1:
Objeto o foreground

Pixels a 0:
Fondo o background

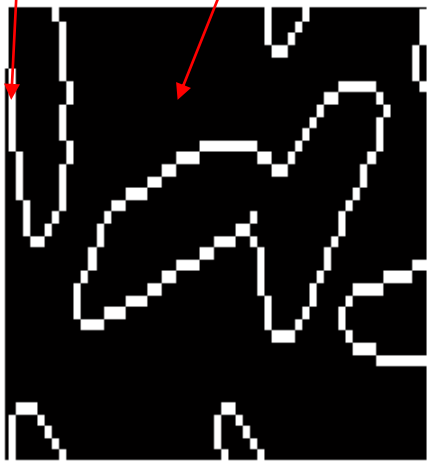
Pixels aislados

Huecos



Contornos abiertos

Contornos unidos



Operaciones con imag. binarias

- Para mejorar el resultado de la binarización:
  - Operaciones morfológicas
    - Modifican el valor de un pixel en función de su entorno
    - El entorno se define a partir de un ‘structuring element’ (strel)
    - Sólo se consideran los pixels del entorno para los cuales el strel es 1
  - Operaciones morfológicas básicas
    - Dilatación o máximo binario:
      - Si un pixel estaba a 0, pasa a 1 si alguno de su entorno es 1.
    - Erosión o mínimo binario:
      - Si un pixel estaba a 1, pasa a 0 si alguno de su entorno es 0.

strel(‘disk’,4)

0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	0	1	1	1	0	0

Operaciones con imag. binarias

- Dilatación:
  - Permite unir o rellenar pixels a 0 que deberían estar a 1
  - Su efecto secundario es el crecimiento de las regiones a 1
- Erosión:
  - Permite eliminar pixels aislados a 1 que deberían estar a 0
  - Su efecto secundario es el adelgazamiento de las regiones a 1

Imagen binaria original

Imagen dilatada (disco r=2)

Imagen erosionada (disco r=2)

Operaciones con imag. binarias

- Mejora de resultados:
  - Encadenar varias operaciones de dilatación y erosión con el mismo strel
  - Dilatación seguida de erosión: cierre
  - Erosión seguida de dilatación: apertura

Imagen binaria original

Imagen cerrada (disco r=2)

Imagen abierta (disco r=2)

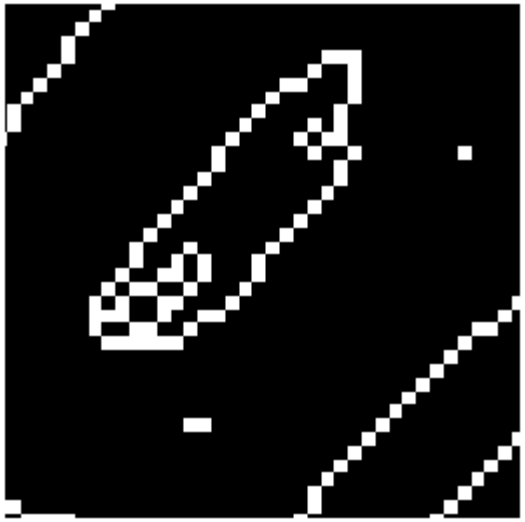
Operaciones con imag. binarias

- Mejora de resultados:
  - Operar la imagen abierta o cerrada con la original
  - Imagen dilatada – original = perímetro exterior
  - Imagen original – erosionada = perímetro interior

Imagen binaria original

Perímetro exterior

Perímetro interior

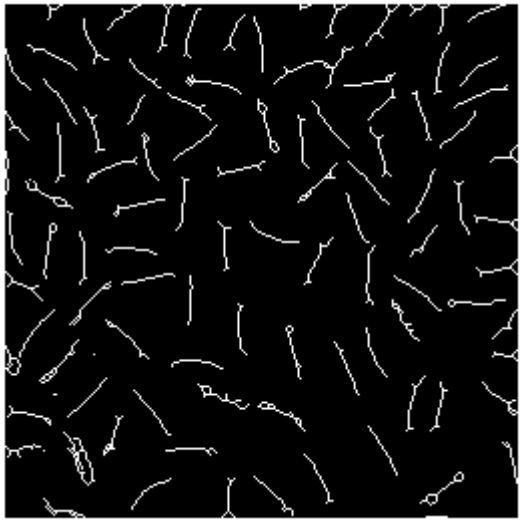


Operaciones con imag. binarias

- Otras combinaciones (ver ayuda bwmorph):
  - Esqueletonización: skel
  - Puente: bridge
  - Eliminar aislados: clean
  - Rellenar huecos: fill
- También es posible combinar strels diferentes:
  - Ejemplo bwhitmiss

Imagen binaria original

Esqueletonización



Operaciones con imag. binarias

- Mejora de resultados:
  - Operaciones de reconstrucción morfológica
    - Ej.: rellenado
    - Utilizan el concepto de conectividad para ir siguiendo los pixels, en lugar de un strel

Conectividad-4 Conectividad-8

    - Están todas basadas en la operación de reconstrucción:
      - Usa 2 imágenes
      - Va dilatando la 1ª (marker) mientras no supere a la 2ª (mask)
      - El resultado es la dilatación máxima

Operaciones con imag. binarias

- Mejora de resultados:
  - Operaciones de reconstrucción morfológica
    - **Rellenado**: utiliza la reconstrucción morfológica para rellenar huecos.
    - Un hueco es una zona de fondo completamente rodeada por pixels de objeto.
    - El concepto ‘completamente rodeada’ depende de la conectividad utilizada:

Hueco usando Conectividad-4

0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	0	0	0	0	1
1	1	1	0	0	1	1
1	1	1	1	0	1	1
0	1	1	1	1	0	0
0	0	1	1	1	0	0

Hueco usando Conectividad-8

0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	0	0	0	0	1
1	1	1	0	0	1	1
1	1	1	1	0	1	1
0	1	1	1	1	0	0
0	0	1	1	1	0	0

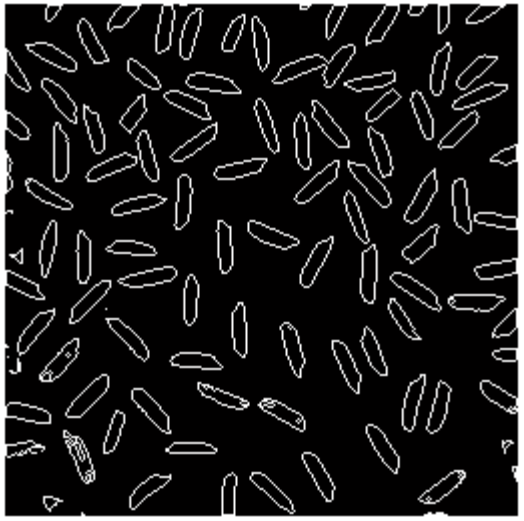
Operaciones con imag. binarias

- Mejora de resultados:
  - Operaciones de reconstrucción morfológica
    - **Rellenado**: rellenar huecos de objetos (usa el fondo).
    - **Perímetro:** buscar límites externos de objetos.

Imagen original

Huecos rellenados

Perímetro



Indice

- Estructura del curso
- Detección de bordes
- Binarización por umbral
- Operaciones con imágenes binarizadas
- **Ejemplos con Matlab**

Ejemplos con Matlab

- Carga de imagen:

```
>> imgs=imread('coins.png');
>> figure; imagesc(imgs); colormap gray; colorbar;
```

- Detección de contornos:

```
>> bw=edge(imgs,tipodetect,params);
```

  - tipodetect: ‘sobel’, ‘prewitt’, ‘canny’, ‘roberts’, ‘log’, ‘zerocross’
  - Cada uno de ellos tiene sus propios parámetros.
  - Todos ellos devuelven una imagen binaria.

- Si se quiere imagen en niveles de gris:

```
>> kernel=fspecial(tipodetect);
>> imgd=imfilter(imgs,kernel);
```

  - O bien

```
>> kernel=[matriz deseada];
>> imgd=imfilter(imgs,kernel);
```

Ejemplos con Matlab

- Binarización por umbral:

```
>> bw=imbinarize(imgs,umbral);
```

O también usando operadores binarios de matlab:

```
>> bw=(imgs>=umbral);
>> bw=imgs>=umbralmin & imgs<=umbralmax;
>> bw=imgs<umbralmin | imgs>umbralmax;
```

- Selección de umbral por método de Otsu:

```
>> umbral=graythresh(imgs);
```

Ejemplos con Matlab

- Operaciones morfológicas:
  - Crear un elemento estructural:

```
>> str=strel('disk',5);
```

    - Tipos: 'disk', 'rectangle', 'diamond', 'line', otros
  - Aplicar operaciones:

```
>> bwd=imdilate(bw,str);
>> bwd=imerode(bw,str);
>> bwd=imclose(bw,str);
>> bwd=imopen(bw,str);
>> bwd=bwmorph(bw,'skel');
```

Ejemplos con Matlab

- Operaciones de reconstrucción morfológica:
  - Rellenado de huecos

```
>> bwd=bwfill(bw);
>> bwd=bwfill(bw, 'holes');
>> bwd=bwfill(bw,[fila col]);
```

Selección interactiva de huecos

Todos los huecos

El hueco que empieza en este punto

  - Obtención de perímetro

```
>> bwd=bwperim(bw);
```

Ejemplos con Matlab

- Otras operaciones interesantes con imágenes binarias:
  - Distancias del fondo a los objetos:

```
>> imgd=bwdist(bw);
```

  - Distancias de los objetos al fondo:

```
>> imgd=bwdist(~bw);
```

  - Nº de euler (nº de objetos – nº de huecos):

```
>> eu=bweuler(bw);
```

  - Area total de objetos:

```
>> area=bwarea(bw);
```