Visión artificial IV

Segmentación

Segmentación

- La segmentación consiste en dividir una imagen en regiones u objetos cuyos píxeles poseen atributos similares (niveles de grises, texturas, …).
- Las técnicas de segmentación se clasifican en:
  - Detección de los contornos de los objetos: Localiza las fronteras de los objetos.
  - Búsqueda de regiones homogéneas: agrupa los píxeles por que cumplen algún criterio de similitud y tienen conectividad entre ellos.

Umbralización

- Transformación de una imagen en niveles de grises a una imagen binaria
- Suponemos fondos y objetos uniformes

Objetos claros sobre fondo oscuro → $g(x,y) = \begin{cases} 1 & si \quad f(x,y) \geq T \\ 0 & en\ otro\ caso \end{cases}$

Objetos oscuros sobre fondo claro → $g(x,y) = \begin{cases} 1 & si \quad f(x,y) \leq T \\ 0 & en\ otro\ caso \end{cases}$

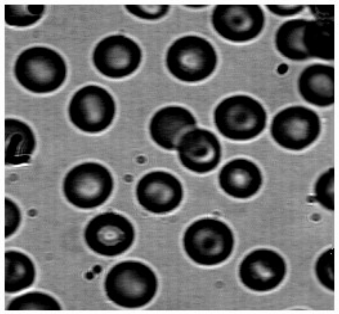
Objetos en un intervalo de grises → $g(x,y) = \begin{cases} 1 & si \quad T_{\min} \leq f(x,y) \leq T_{\max} \\ 0 & en\ otro\ caso \end{cases}$

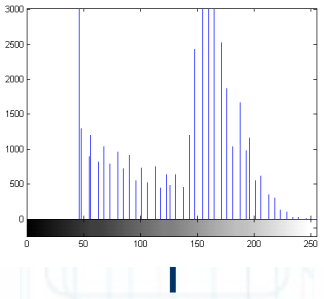
Umbralización

- Clasificación de los umbrales:
  - Global: sólo depende de los valores de los píxeles de la imagen: *f(x,y)*.
  - Local: depende además de alguna propiedad local del píxel (por ejemplo, valor medio de la vecindad).
  - Dinámico: depende además de la posición del píxel.

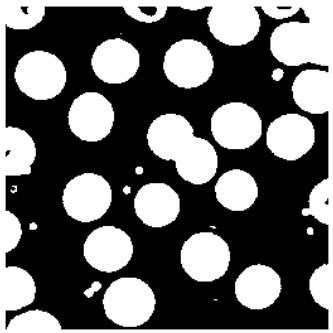
Umbralización

- Segmentación por histograma:
  - Técnica global.
  - Considera:
    - Una definición clara de los objetos respecto del fondo.
    - El histograma lo constituye dos picos y un valle (se puede generalizar, aunque los resultados son menos fiables).



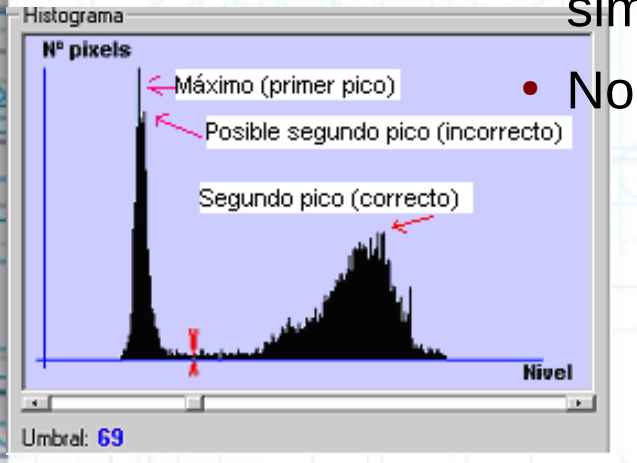


umbral



Umbralización

- Segmentación por histograma:
  - Limitaciones:
    - Difícil identificar los mínimos del histograma.
    - Problemas cuando las regiones varían suavemente su nivel (por ejemplo, sombras).
    - Aplicable con pocas regiones.
    - No distingue regiones separadas de niveles similares de gris.
    - No considera la conectividad de los píxeles.



Umbralización

- Segmentación por histograma:
  - Estrategias para la elección del umbral óptimo:
    - Ajuste gaussiano.
    - Minimización de la varianza interclase.
    - Entropía del histograma.
    - Análisis de la concavidad.
    - Métodos basados en momentos.

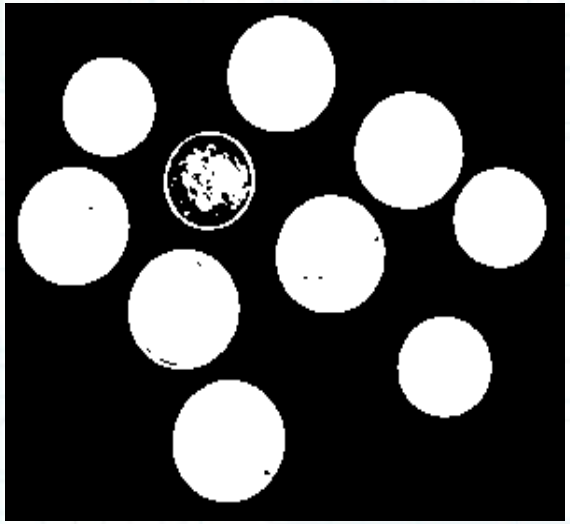
Umbralización

- Segmentación por histograma:
  - Método de Otsu:
    - Método basado en la suposición de que la función de densidad del fondo y la de los objetos tienen un modelo gaussiano.
    - Minimiza la suma ponderada de cada una de las varianzas de las clases.
    - MATLAB: graythresh(imagen)

Umbralización

- Segmentación por histograma:
  - Método de Otsu:

```
imEnt = imread('coins.png');
imshow(imEnt,[]);
umbral = graythresh(imEnt);
imBW = im2bw(imEnt,umbral);
figure, imshow(imBW);
```



Orientada a las regiones

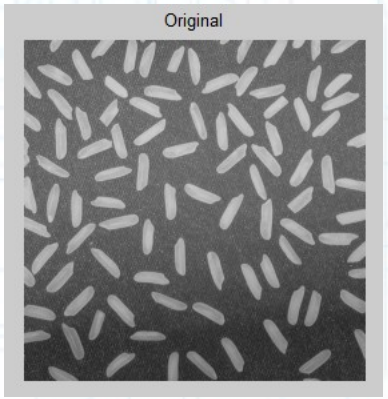
- Etiquetado de componentes conexas:
  - Sobre imágenes binarizadas.
  - Agrupar los píxeles de la misma región asignándoles la misma etiqueta.

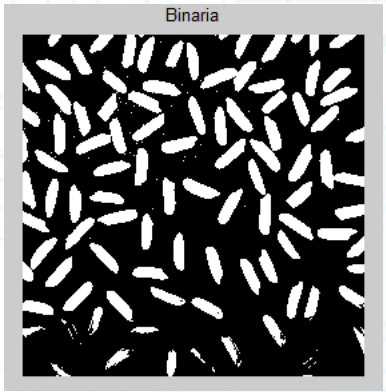
0	1	0	0	0	2
1	1	0	0	0	2
0	1	1	0	0	2
1	1	0	0	0	2
0	1	1	1	0	2

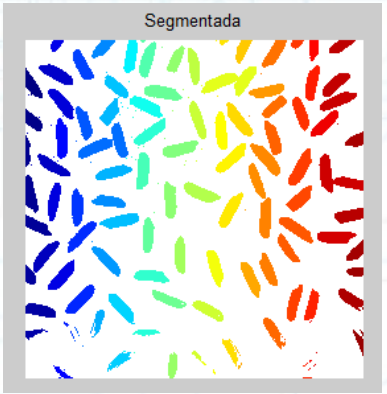
Orientada a las regiones

- Etiquetado de componentes conexas:

```
I = imread('rice.png');
figure, imshow(I);
title('Original')

% umbralización
Umbral=graythresh(I);
B = im2bw(I, Umbral);
figure, imshow(B);
title('Binaria');

% Componentes conexas (conectividad 4)
L = bwlabel(B,4);
C = label2rgb(L); % colorea cada región
figure, imshow(C);
title('Segmentada');
```







Orientada a las regiones

- Etiquetado de componentes conexas:

```
I = imread('rice.png');
figure, imshow(I);
title('Original')

% umbralización
Umbral=graythresh(I);
B = im2bw(I, Umbral);
figure, imshow(B);
title('Binaria');

% Componentes conexas (conectividad 4)
CC = bwconncomp(B,4);
% CC es una estructura con: conectividad,
% tamaño, n° de objetos y una lista a los
% índices de cada objeto
L = labelmatrix(CC); % Crea la matriz de etiquetas
C = label2rgb(L); % colorea cada región
figure, imshow(C);
title('Segmentada');
```

Orientada a las regiones

- Crecimiento de regiones mediante la adición de píxeles:
  - Se eligen una serie de semillas iniciales (puntos) y píxeles contiguos con valores similares se unen a las semillas.
  - Se repite mientras se añadan píxeles.
  - Cuando ninguna región puede crecer se añaden nuevas semillas.

Orientada a las regiones

- Crecimiento de regiones mediante la adición de píxeles:

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
0	1	5	6	5
0	1	5	6	5

Semillas: (3,2) y (3,4)

Umbral=3

$$si \begin{cases} |f(i,j) - f(3,2)| < umbral & R_a \\ |f(i,j) - f(3,4)| < umbral & R_b \end{cases}$$

a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b
a	a	b	b	b

Tantas regiones como semillas
¿Cuántas semillas?¿Dónde las coloco?
¿Qué umbral?

Orientada a las regiones

- Crecimiento de regiones mediante la adición de píxeles:

```
imEnt = [7 7 1 1 1 1 1 1 2 2 2 2
         2 2 1 2 2 1 1 1 2 2 2 2
         1 3 3 3 4 2 2 2 2 2 2 2
         1 2 3 4 5 2 2 2 2 2 3 2
         2 2 2 2 2 2 2 2 2 2 2 2
         1 1 1 1 1 1 1 1 1 1 1 1
         1 1 1 1 1 1 1 2 2 2 2 2
         1 3 4 4 4 4 5 6 7 3 4 1
         1 2 3 3 3 3 3 3 3 3 3 3
         4 4 2 3 4 5 6 8 8 8 8 8
         1 2 3 4 4 4 8 8 8 8 8 8
         1 2 3 4 5 6 8 8 8 8 8 8];

[imG,num_reg,imSeed,imThreshold] = regiongrow(imEnt,8,2)
```

  - Selecciona los píxeles con valor igual que la semilla (8). Ver imSeed.
  - Selecciona los píxeles que satisfacen el umbral (2). Ver imThreshold.
  - Selecciona aquellos píxeles con conectividad 8.

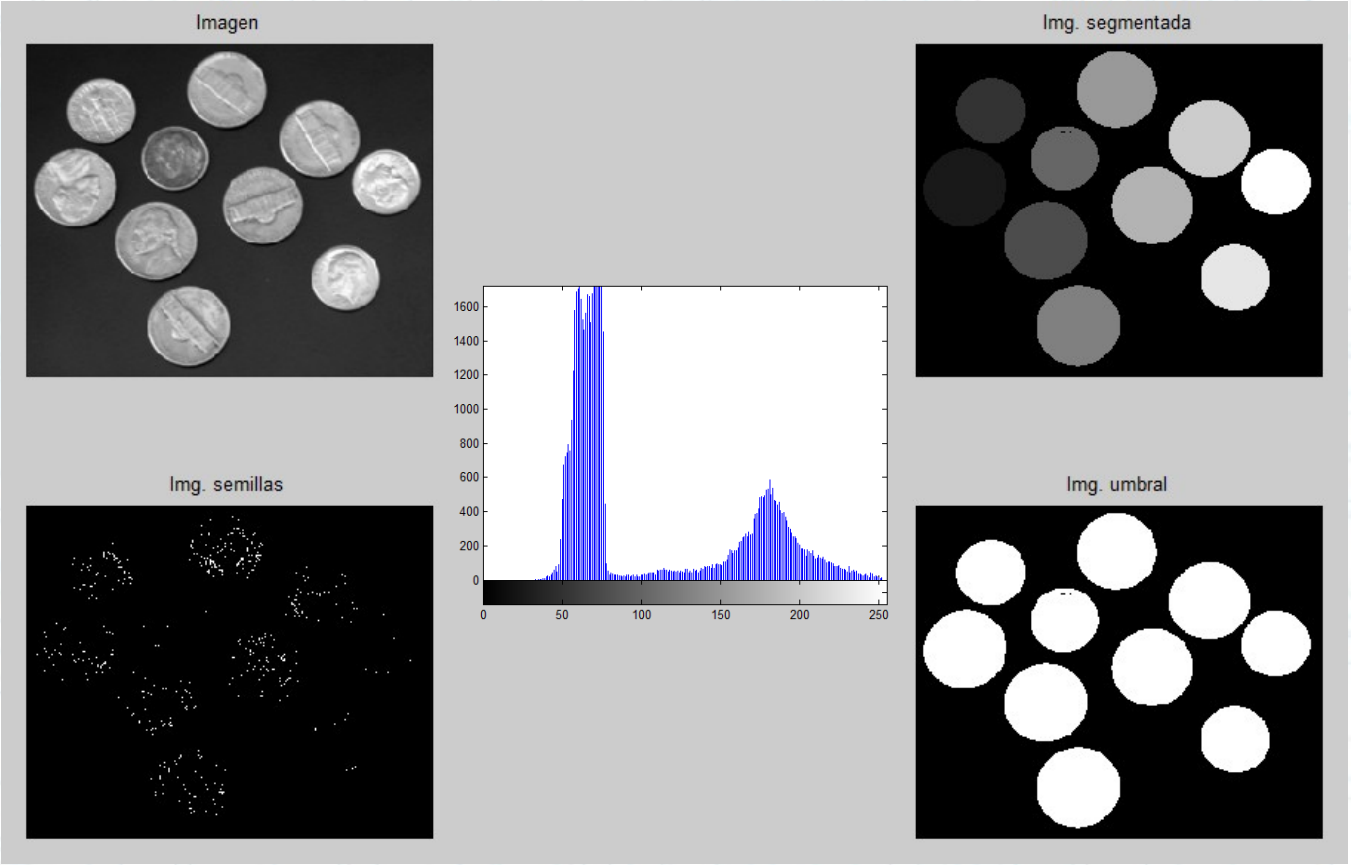
Orientada a las regiones

- Crecimiento de regiones mediante la adición de píxeles:

```
imEnt = imread('coins.png');
imEnt = imfilter(imEnt,fspecial('gaussian'));
[imG,num_reg,imSeed,imThreshold] = regiongrow(imEnt,180,90);
subplot(2,2,1);
imshow(imEnt,[]);
title('Imagen');
subplot(2,2,2);
imshow(imG,[]);
title('Img. segmentada');
subplot(2,2,3);
imshow(imSeed,[]);
title('Img. semillas');
subplot(2,2,4);
imshow(imThreshold,[]);
title('Img. umbral');
```

Orientada a las regiones

- Crecimiento de regiones mediante la adición de píxeles:



Orientada a las regiones

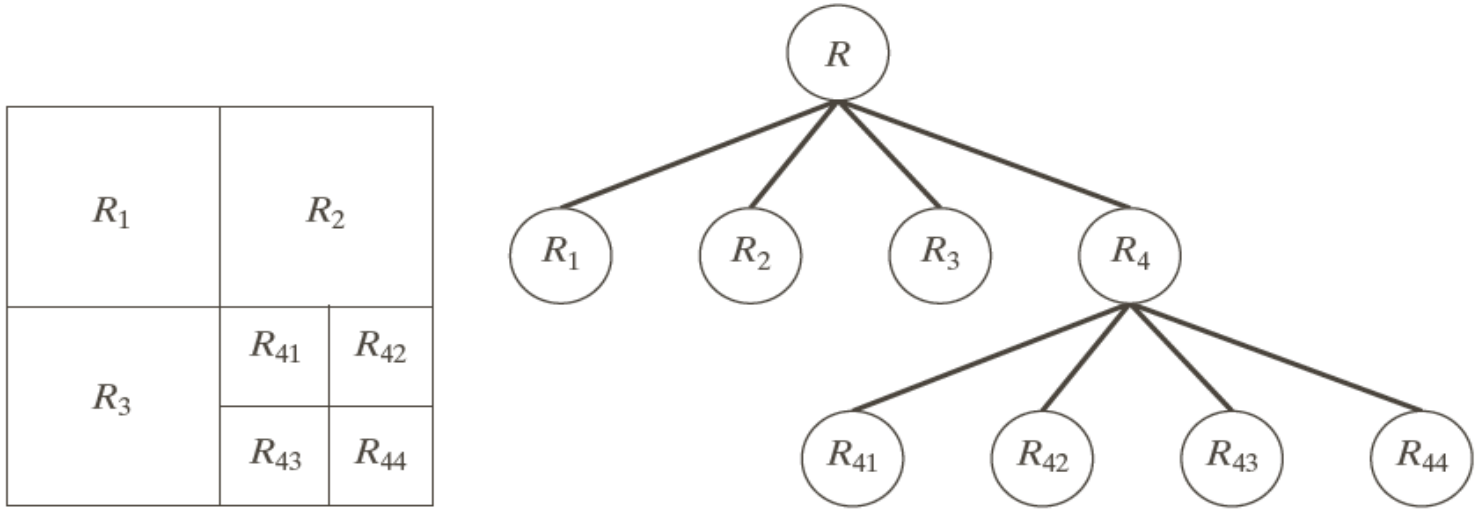
- Crecimiento de regiones mediante subdivisión y fusión:
  - Se subdivide la imagen inicialmente en un conjunto de regiones disjuntas, dentro de las cuales, se volverá a realizar una subdivisión o bien una fusión entre ellas, dependiendo de si se verifican las condiciones prefijadas.

Orientada a las regiones

- Crecimiento de regiones mediante subdivisión y fusión:
  - La estructura más usada para la subdivisión es el árbol cuaternario. Los pasos a seguir son:
    1. Se define un test de homogeneidad
    2. Se subdivide la imagen en los cuatro cuadrantes disjuntos
    3. Se calcula la medida de homogeneidad para cada cuadrante
    4. Se fusionan dos regiones si la condición de homogeneidad se verifica para la unión de las mismas.
    5. Si una región no verifica la condición, se vuelve a subdividir en sus cuatro cuadrantes y se repite el proceso hasta que todas las regiones pasan el test de homogeneidad.

División de regiones

- Crecimiento de regiones mediante subdivisión y fusión:



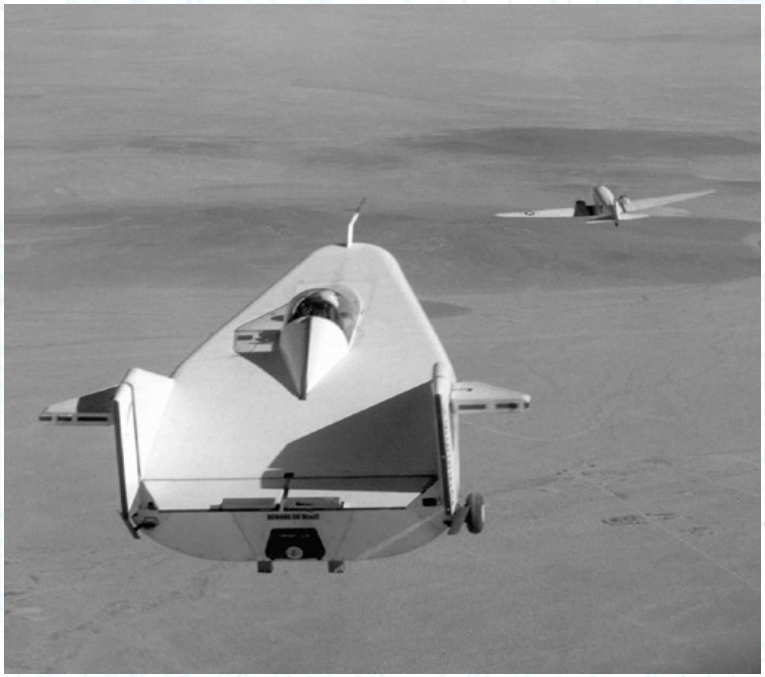
Orientada a las regiones

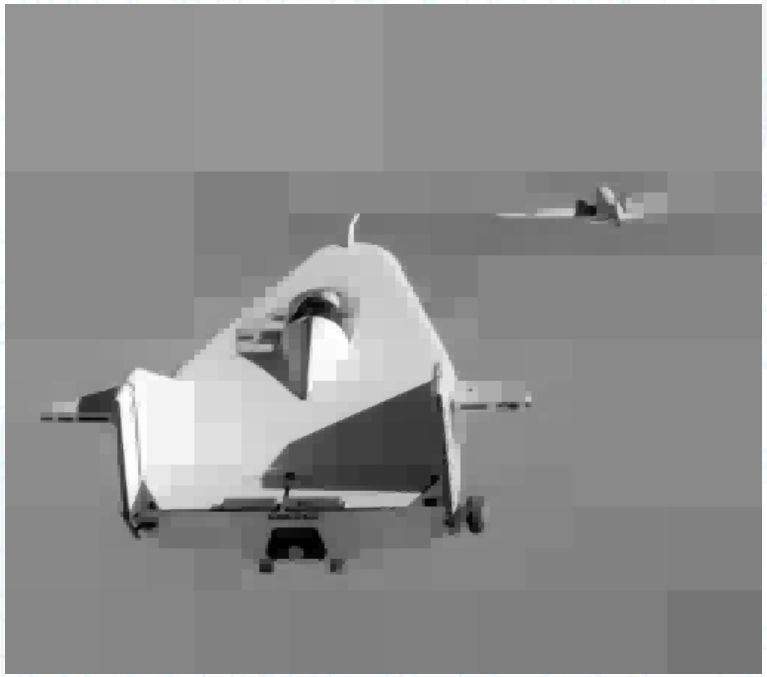
- Crecimiento de regiones mediante subdivisión y fusión:

```
imgEnt = imread('liftingbody.png');
im1 = imfilter(imgEnt,fspecial('gaussian'));
imgDescomp = qtdecomp(im1,0.27);
imgDivision = imgEnt;
for dim = [256 128 64 32 16 8 4 2 1];
    [valores,fila,columna] = qtgetblk(im1,imgDescomp,dim);
    if (~isempty(valores)),
        doublesum = sum(sum(valores,1,'double'),2);
        imgDivision = qtsetblk(imgDivision,imgDescomp, ...
            dim, doublesum./dim^2);
    end
end
imshow(imgDivision,[])
```

Orientada a las regiones

- Crecimiento de regiones mediante subdivisión y fusión:



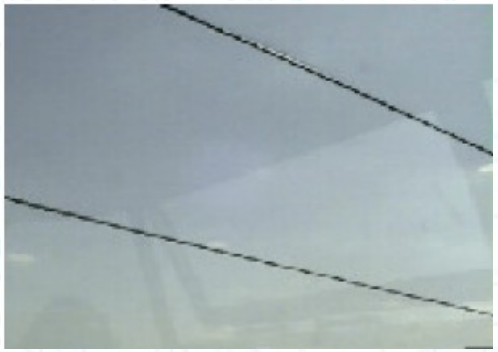
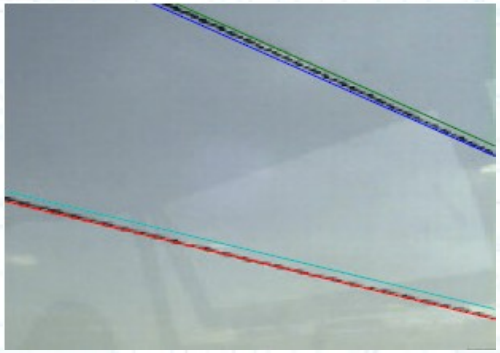


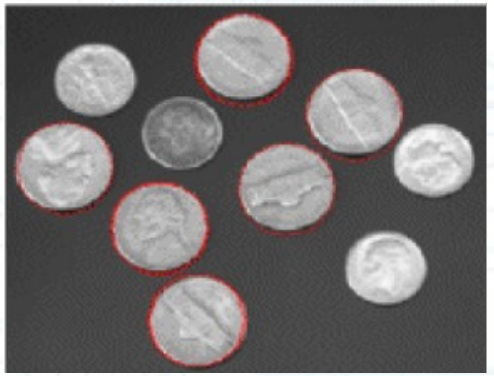
Detección de fronteras

- Las técnicas de detección de bordes son necesarias, pero no suficientes para localizar las fronteras de los objetos a segmentar debido a que la presencia de ruido, sombras, iluminación no uniforme, … produce contornos que no son del todo continuos y cerrados sobre los objetos.
- Se agrupan los píxeles etiquetados como bordes empleando la propiedad de conectividad (para que un pixel etiquetado como borde sea un píxel frontera se necesita que otros píxeles bordes tengan similar dirección y módulo del gradiente).
- Técnicas para cerrar contornos: transformada de Hough, transformada de Radon.

Detección de fronteras

- Transformada de Hough:
  - Permite encontrar en la imagen ciertos patrones (líneas, círculos, …)



Detección de fronteras

- Transformada de Hough (detección de líneas):
  - Sobre un pixel de coordenadas (x_i,y_i), seleccionado como elemento del borde, pasarán infinitas rectas que satisfacen la siguiente ecuación:

$$y_i = ax_i + b$$

  - Si consideramos el plano ab (espacio de parámetros) tenemos una única recta para el par (x_i,y_i) constante.

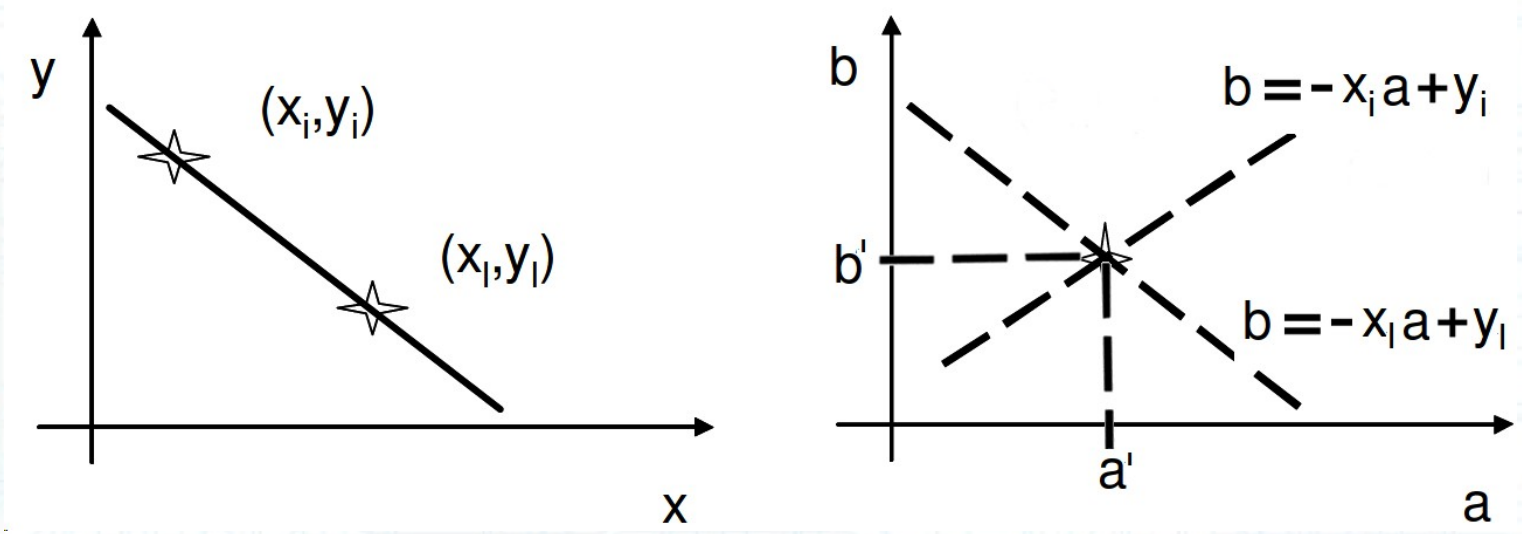
$$b = y_i - ax_i$$

Detección de fronteras

- Transformada de Hough (detección de líneas):
  - Si consideramos un segundo punto (x_l,y_l), su recta asociada al espacio de parámetros se cortará en el punto (a',b') con la del punto (x_i,y_i) si las dos parejas de puntos pertenecen a la misma recta en el plano xy.
  - a' es la pendiente y b' la ordenada al origen de la recta en el plano xy.

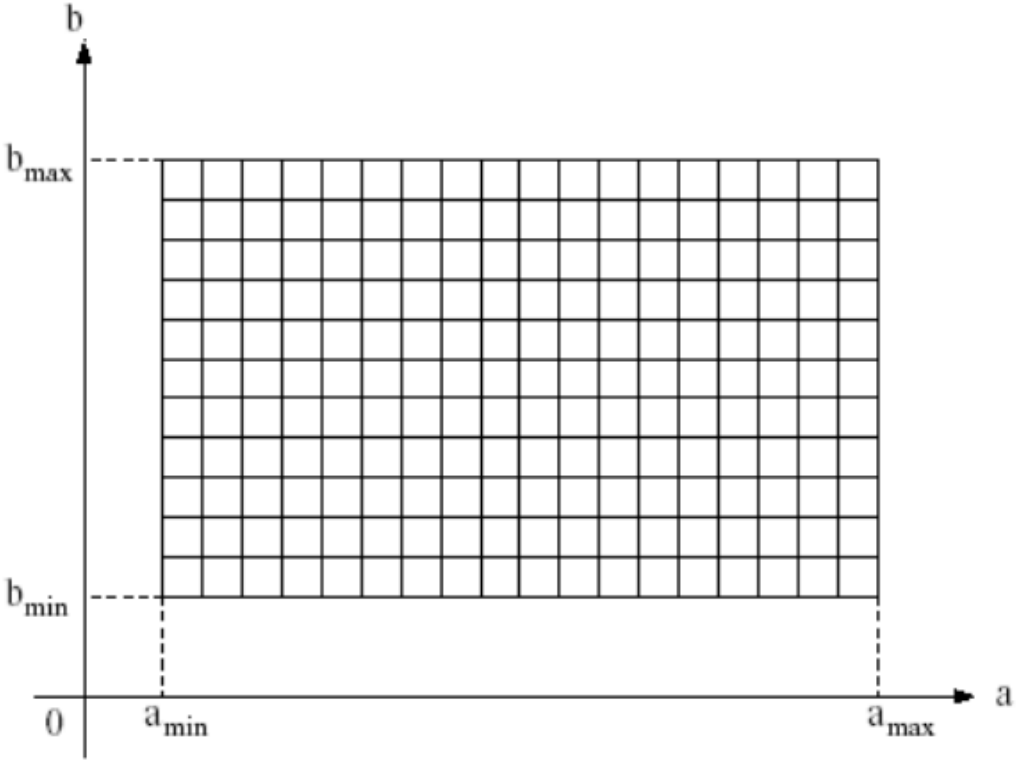
Detección de fronteras

- Transformada de Hough (detección de líneas):



Detección de fronteras

- Transformada de Hough (detección de líneas):
  - Subdividimos el espacio de parámetros en celdas acumuladoras.



Detección de fronteras

- Transformada de Hough (detección de líneas):
  - Algoritmo:
    1) Poner todos los acumuladores a cero.
    2) Para cada punto (x_k,y_k):
       a) el parámetro toma cualquier valor de los a_i permitidos para calcular $b = -x_k a + y_k$.
       b) Se redondea b a b_j.
       c) A(i,j) = A(i,j) + 1

Detección de fronteras

- Transformada de Hough (detección de líneas):
  - Un valor M en el acumulador A(i,j) indica que M puntos del plano xy pertenecen a la recta $y = a_i x + b_j$.
  - La precisión de la colinealidad de los puntos depende del número de celdas del espacio de parámetros.
  - La carga computacional es del orden *nK*, donde *n* es el número de puntos de la imagen y *K* el nº de celdas en que subdividimos el eje *a*.

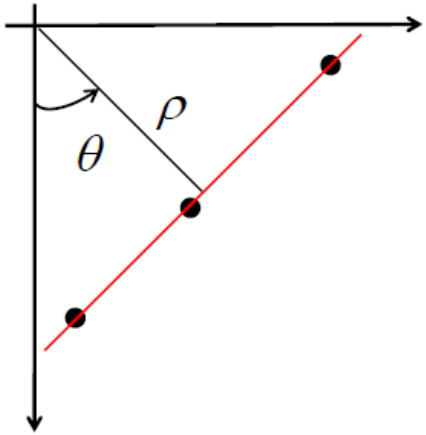
Detección de fronteras

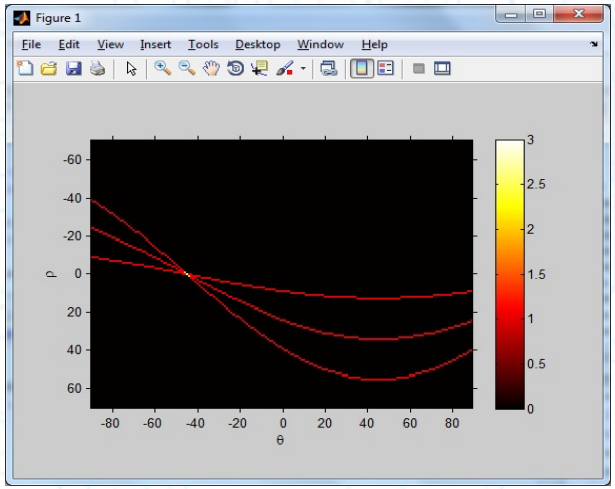
- Transformada de Hough (detección de líneas):
  - Problema: Que con la ecuación y = ax + b, tanto la pendiente como la ordenada al origen pueden llegar a valer infinito (según la línea se hace vertical).
  - Solución: Utilizar la representación polar de la recta:

$$x * \cos\theta + y * sen\theta = \rho$$

Detección de fronteras

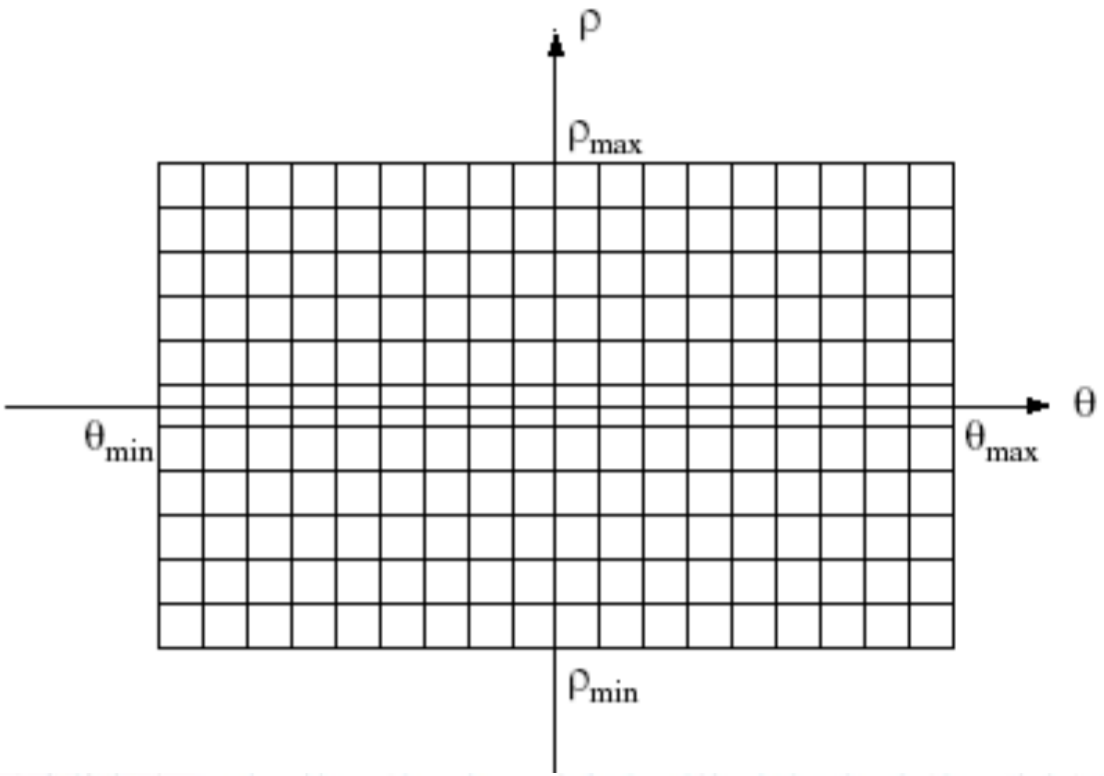
- Transformada de Hough (detección de líneas):
  - Solución: Utilizar la representación polar de la recta:
    - A cada punto del plano le corresponde una sinusoide en el plano $\rho\theta$





Detección de fronteras

- Transformada de Hough (detección de líneas):
  - Solución: Utilizar la representación normal de la recta:



- El rango de θ es ± 90º con respecto al eje de abcisas.

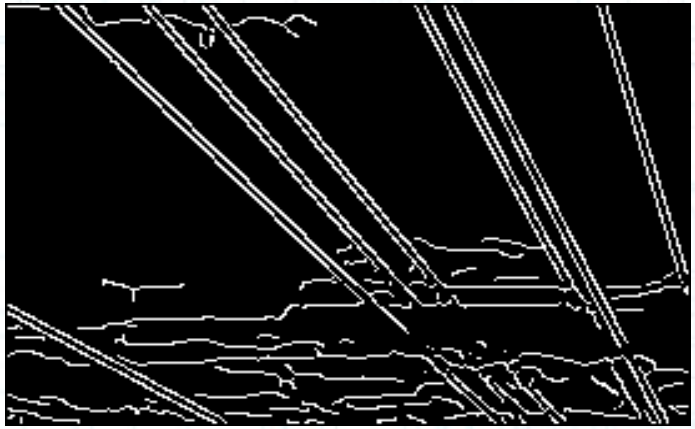
Detección de fronteras

- Transformada de Hough (detección de líneas):

```
I =rgb2gray(imread('cableado02.jpg'));
max_dist = sqrt(size(I,1)^2 + size(I,2)^2);
imshow(I,[]);
E = edge(I,'canny');
figure,imshow(E);
% Calcula la transformada de Hough sobre una imagen binaria
[H, theta, rho] =hough(E);
% Obtenemos 5 líneas
peaks = houghpeaks(H,5);
lines = houghlines(E, theta, rho,
peaks,'FillGap',max_dist);
figure,imshow(I,[]);
for k = 1:length(lines),
   xy = [lines(k).point1; lines(k).point2];
   line(xy(:,1),xy(:,2),'LineWidth',1.5,'Color','g');
end
```

Detección de fronteras

- Transformada de Hough (detección de líneas):



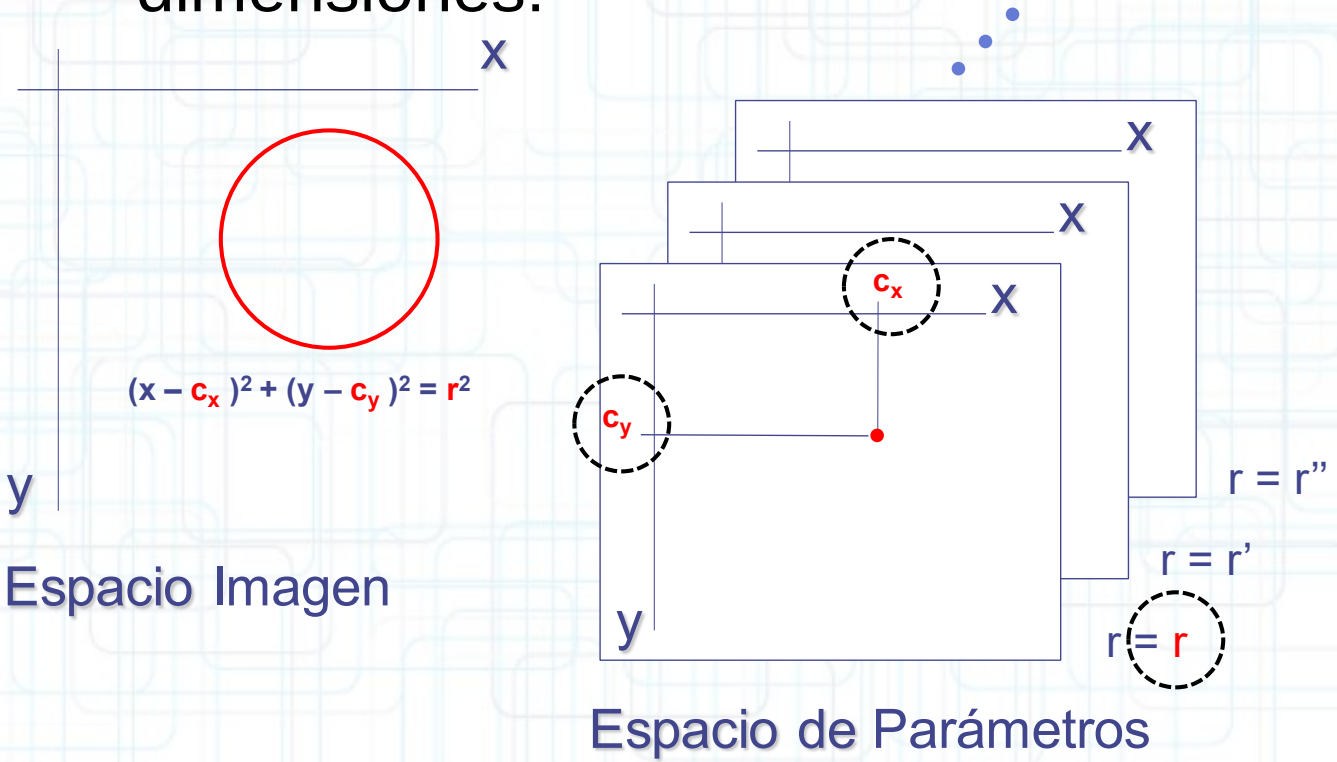
Detección de fronteras

- Transformada de Hough (detección de circunferencias):
  - La transformada de Hough es aplicable a cualquier función de la forma g(v,c)=0, donde:
    - v = vector de coordenadas.
    - c = vector de coeficientes.
  - Para detección de círculos, el espacio paramétrico será 3D:
    - La posición del centro del círculo.
    - El radio.

$$(x - c_x)^2 + (y - c_y)^2 = r^2$$

Detección de fronteras

- Transformada de Hough (detección de circunferencias):
  - El espacio de parámetros tendrá tres dimensiones:



de la Fuente López, E., & Trespaderne, F. M. (2012). Visión artificial industrial: procesamiento de imágenes para inspección automática y robótica. Universidad de Valladolid, Secretariado de Publicaciones e Intercambio Editorial.

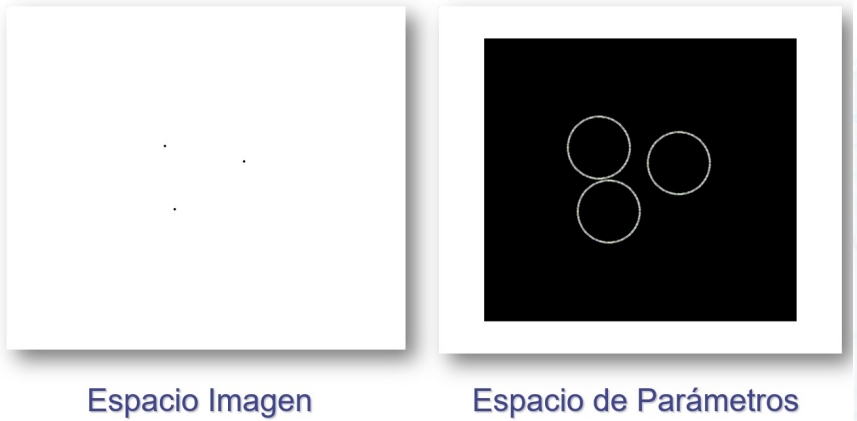
Detección de fronteras

- Transformada de Hough (detección de circunferencias):
  - Cada píxel del contorno vota por todas las circunferencias que pasan por él:

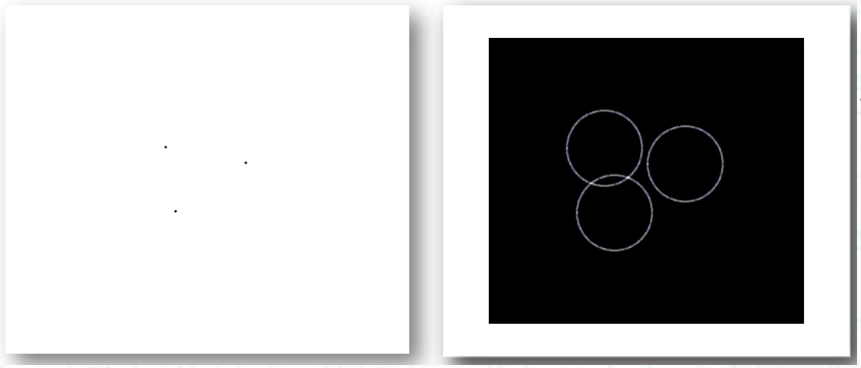
```
for r=r_min:r_max
  for theta=0:360
        c_x=x_0 + cos(theta)*r;
        c_y=y_0 + sin(theta)*r;
        espacio_hough(c_x,c_y,r)=espacio_hough(c_x,c_y,r)+1;
  end
end
```

  - Los parámetros de los puntos del espacio con muchos votos corresponden a a circunferencias por las que pasan una gran cantidad de puntos de contorno.

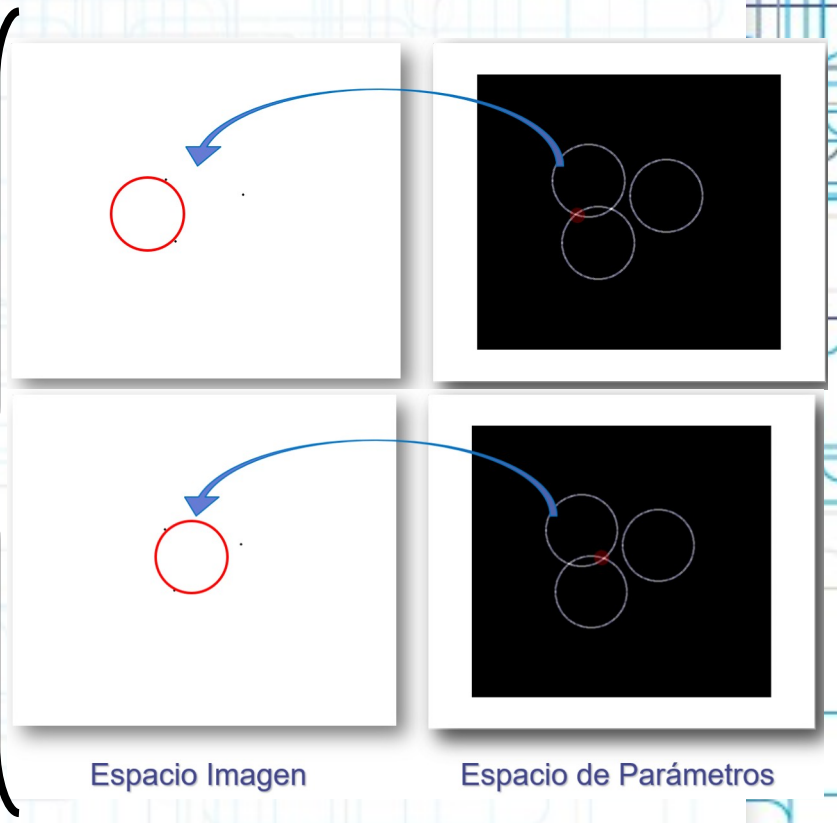
Detección de fronteras



Radio = 50



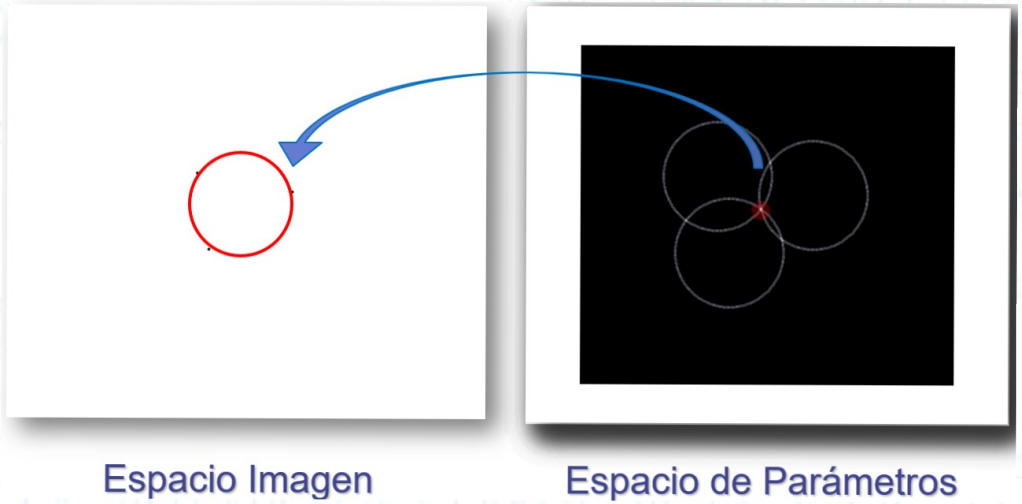
Radio = 60



de la Fuente López, E., & Trespaderne, F. M. (2012). Visión artificial industrial: procesamiento de imágenes para inspección automática y robótica. Universidad de Valladolid, Secretariado de Publicaciones e Intercambio Editorial.

Detección de fronteras

Radio = 73

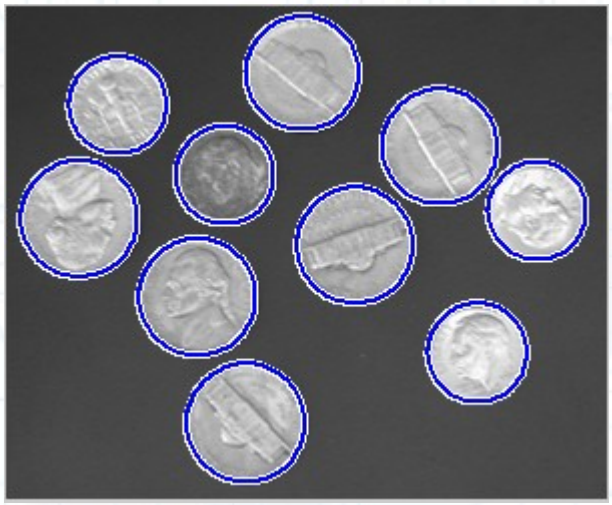


de la Fuente López, E., & Trespaderne, F. M. (2012). Visión artificial industrial: procesamiento de imágenes para inspección automática y robótica. Universidad de Valladolid, Secretariado de Publicaciones e Intercambio Editorial.

Detección de fronteras

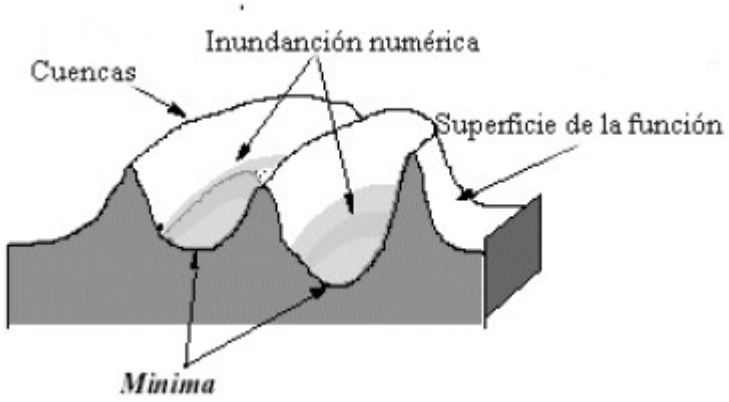
- Transformada de Hough (detección de círculos):

```
% La imagen de entrada puede ser color, escala de grises o binaria
imagen = imread('coins.png');

% Encuentra las círculos cuyos radios se encuentran entre 15 y 30
[centros, radios, metricas] = imfindcircles(imagen,[15 30]);

% Dibuja el perímetro de los círculos
imshow(imagen);
viscircles(centros, radios,'EdgeColor','b');
```



Detección de fronteras

- Watershed:
  - Simula una *inundación numérica* de la imagen a partir de los mínimos regionales.
  - El proceso de inundación hace que cuencas contiguas se unan (las líneas de unión representan las fronteras de las regiones homogéneas).



Detección de fronteras

- Watershed:

```
close all
imagen = imread('cell.tif');
ele = strel('disk',10);
% Restar el fondo a la imagen
imagentophat = imagen - imopen(imagen,ele);
imagenmascontraste = imadjust(imagentophat);
imagenumbralizada =
im2bw(imagenmascontraste,graythresh(imagenmascontraste));
imnegativo = ~imagenumbralizada;
% bwdist: Distancia de cada pixel al más cercano distinto
de cero
distancias = -bwdist(imnegativo);
%surf(double(distancias))
distancias(imnegativo) = -Inf;
imshow(distancias,[]);
L = watershed(distancias);
figure,imshow(L);
regiones_coloreadas = label2rgb(L);
imagen2 = imagen;
imagen2(L==0) = 0;
figure, imshow(imagen2);
```