



22.90 – AUTOMACIÓN INDUSTRIAL

Visión: Segregación, Blobs y Detección de Líneas

OBJETIVO

En esta clase revisaremos principalmente el proceso de segregación, por el cual separaremos la imagen en distintas secciones de importancia.

Los grupos generados mediante la segregación (que llamaremos blobs), pueden caracterizarse de diversas maneras. Utilizando el Toolbox de Corke, calcularemos algunas de estas características.

Por ultimo, estudiaremos un método de detección de líneas en imágenes mediante la transformada de Hough.

SEGREGACION

Proceso de particionar la imagen en sectores relevantes.

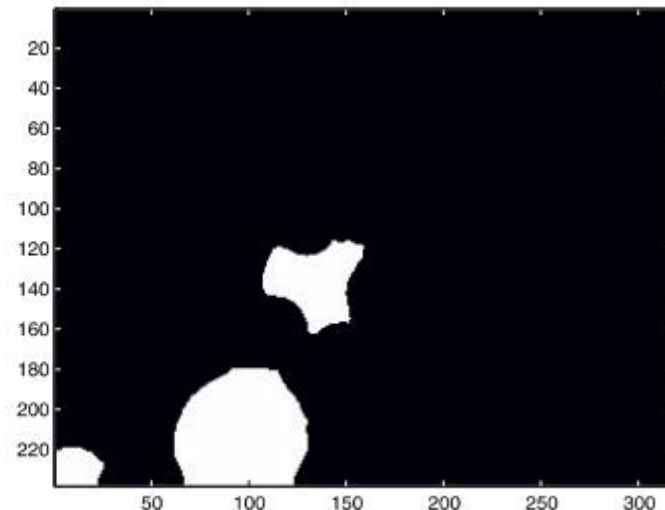
Podemos dividir este proceso en tres etapas:

- Clasificación $C=n$: Tipos posibles de sectores, usualmente $C=2$.
- Segmentación: Detectar conexión entre pixeles sucesivos de la misma clase.
- Descripción: Se describe cada objeto (área, forma, orientación, etc)

CLASIFICACIÓN: THRESHOLDING

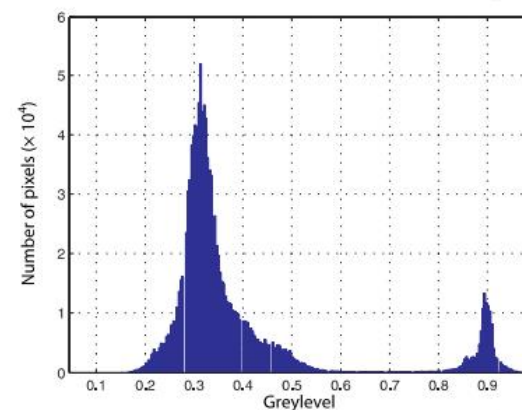
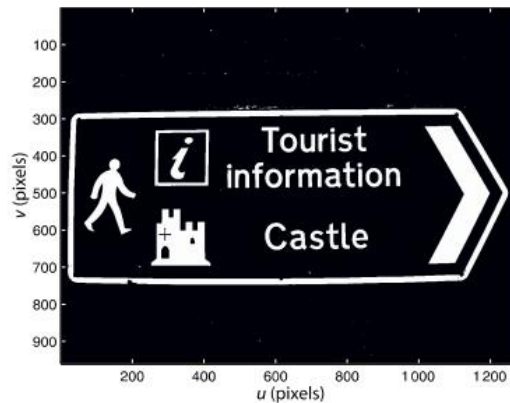
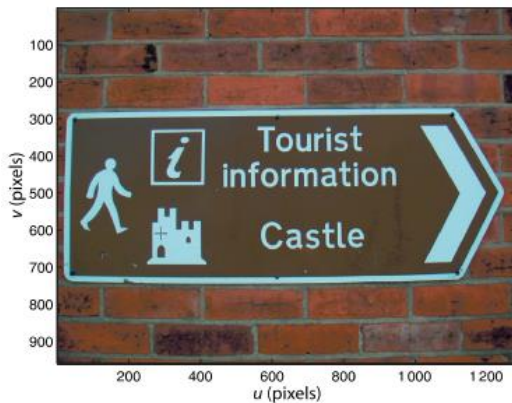
El objetivo de clasificar en un problema de $C=2$, es definir que partes de la imagen son objetos, y que partes son fondo, o no relevantes.

Una de las técnicas mas usuales para clasificar los objetos dentro de una imagen el Thresholding, comentado en la clase 1.



CLASIFICACIÓN: THRESHOLDING

Para seleccionar el umbral optimo, nos basaremos en la información del histograma (en las aplicaciones que lo permitan). Buscaremos Mantener la varianza de cada grupo en el mínimo posible, o lo que es lo mismo, la varianza inter grupo lo mas alta posible.



CLASIFICACIÓN: THRESHOLDING

○ Método de Otsu

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

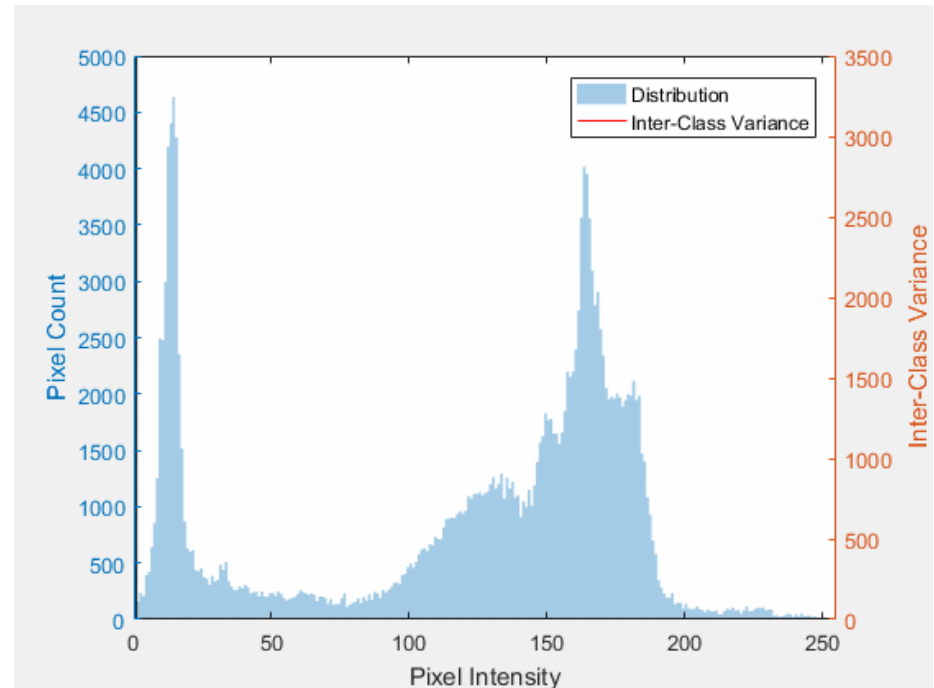
$$\omega_0(t) = \sum_{i=0}^{t-1} p(i)$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i)$$

$$\sigma_b^2(t) = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2$$

$$\mu_0(t) = \frac{\sum_{i=0}^{t-1} ip(i)}{\omega_0(t)}$$

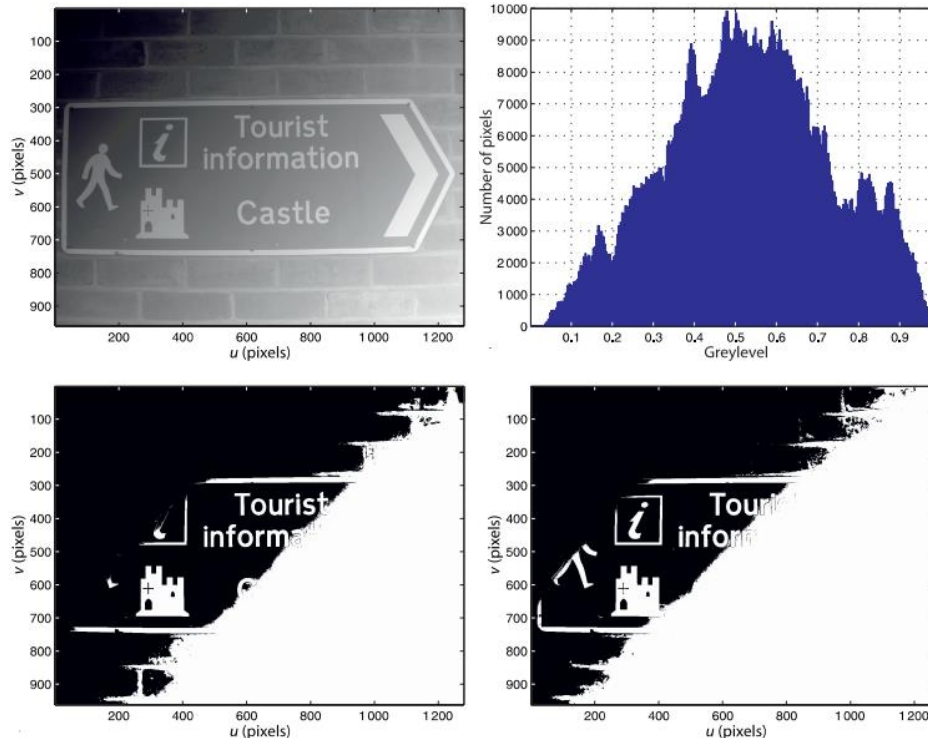
$$\mu_1(t) = \frac{\sum_{i=t}^{L-1} ip(i)}{\omega_1(t)}$$



```
t=otsu(im);
```

CLASIFICACIÓN: THRESHOLDING

En algunas aplicaciones, la distribución de intensidad a lo largo de la imagen no permite separar con facilidad los objetos de interés.



CLASIFICACIÓN: THRESHOLDING

- Este problema puede resolverse mediante el método de Niblack, el cual calcula el umbral de manera localizada, calculado con la media y la distribución en la zona de dimensión W .

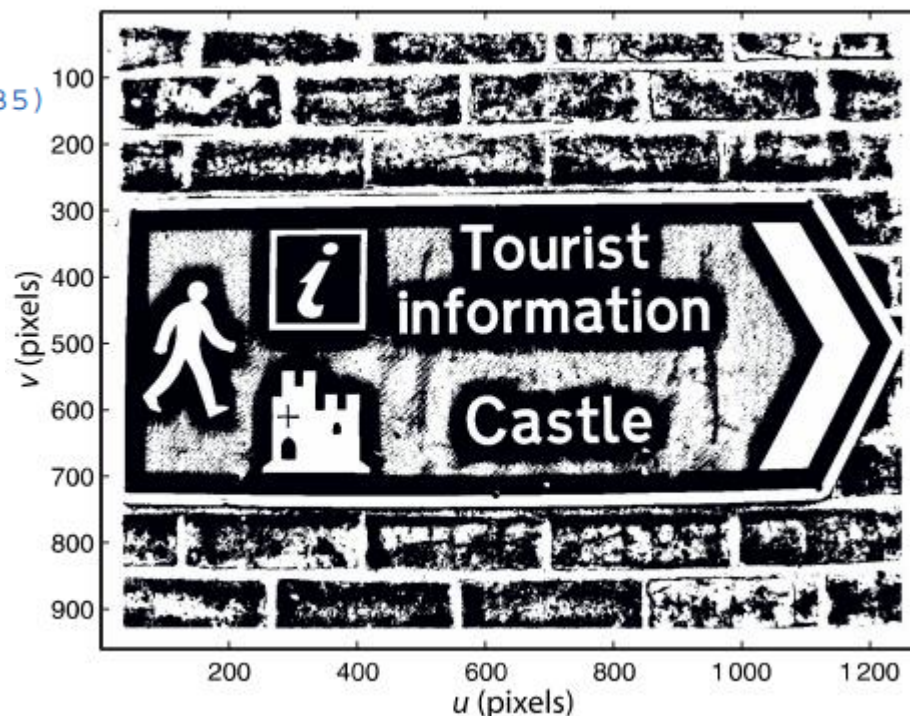
$$t[u, v] = \mu(W) + k\sigma(W)$$

```
>> t = niblack(castle, -0.2, 35)
```

```
>> idisp(t)
```

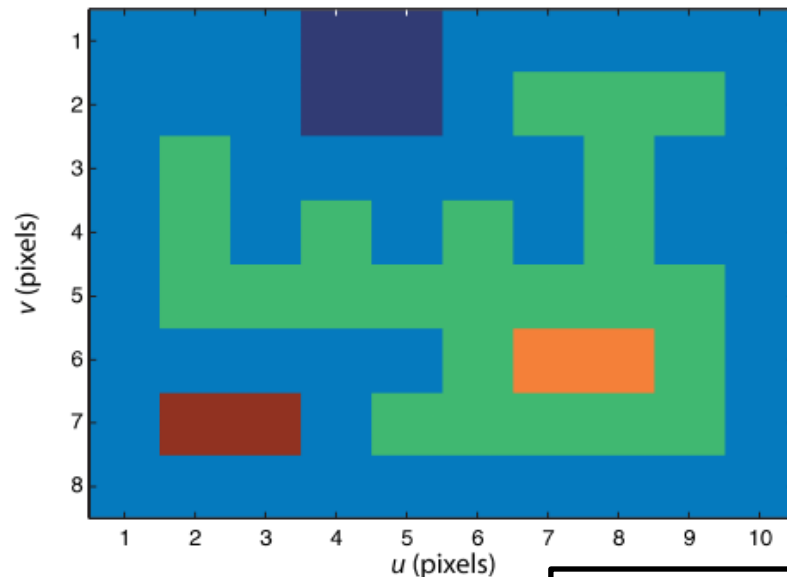
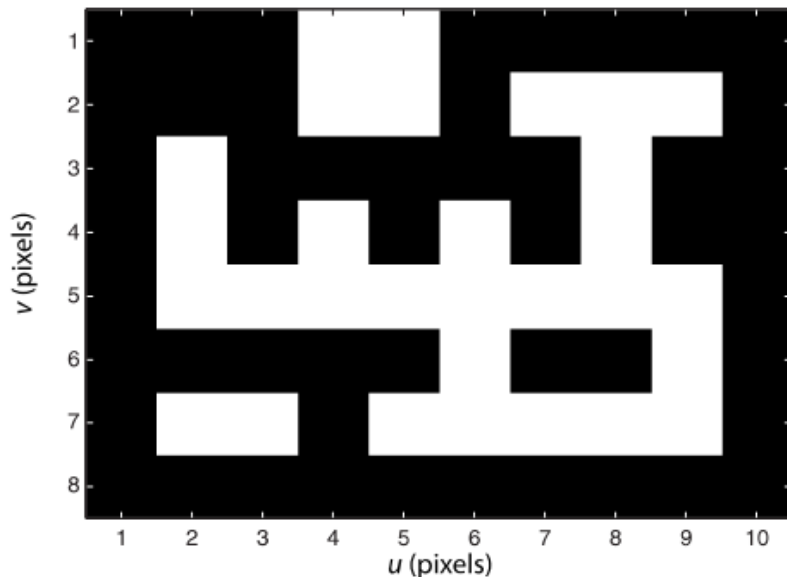
```
>> idisp(castle >= t)
```

Ver algoritmo MSER
(Maximally stable
extremal regions)



SEGMENTACIÓN

Mediante la segmentación, buscaremos que la imagen sea dividida entre cada uno de los objetos presentes en ella, buscando si existe conexión entre un pixel y sus vecinos, y asignando una clasificación a cada grupo.



`ilabel(im)`

SEGMENTACIÓN: BLOBS

Definiremos cada grupo conectado de pixeles como BLOBS, y buscaremos para cada uno de ellos características descriptivas principales:

- Área
- Jerarquía
- Centroide
- Orientación
- Relación de Aspecto
- Morfología (si la figura es simple)

BLOBS

- Área

$$M_{00} = \sum u^0 v^0 I[u, v] = Area$$

- Centroide

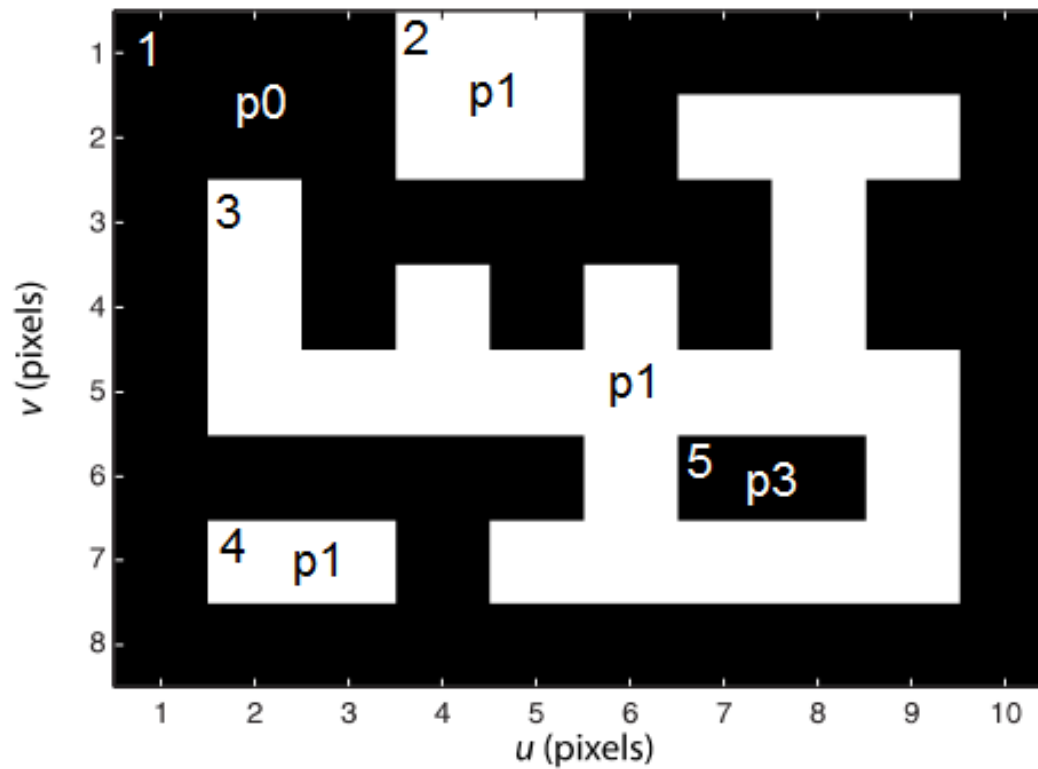
$$M_{10} = \sum u^1 v^0 I[u, v] \quad M_{01} = \sum u^0 v^1 I[u, v]$$

$$U_c = \frac{M_{10}}{M_{00}}$$

$$V_c = \frac{M_{01}}{M_{00}}$$

BLOBS

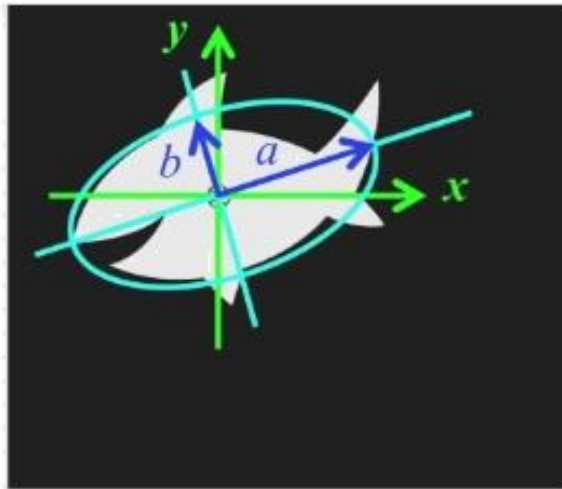
○ Jerarquía



BLOBS

- Elipse equivalente (eig val/vec)

$$\mu_{pq} = \sum_{(u,v) \in \mathbf{I}} (u - u_c)^p (v - v_c)^q \mathbf{I}[u, v] \quad \text{Momentos en centroide}$$



$$\mathbf{J} = \begin{pmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{pmatrix} \quad \text{Matriz de Inercia}$$

$$a = 2\sqrt{\frac{\lambda_1}{m_{00}}}, \quad b = 2\sqrt{\frac{\lambda_2}{m_{00}}}$$

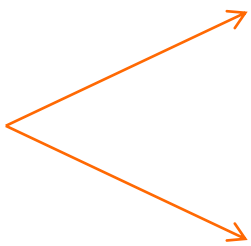
$$\theta = \tan^{-1} \frac{v_y}{v_x}$$

BLOBS

- Relación de aspecto

$$r = \frac{b}{a}$$

- Morfología

$$\rho = \frac{4\pi m_{00}}{p^2}$$


=1 para círculo

= $\pi/4$ para cuadrado

IBLOBS

El Toolbox de Peter Corke realiza tanto la segmentación de la imagen como la descripción de los blobs con la función `iblobs`:

```
>> iblobs(im)
```

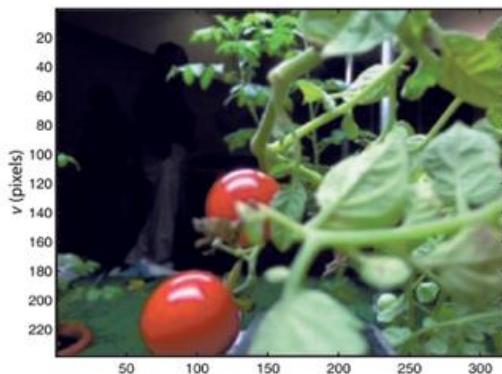
```
ans =
```

```
(1) area=4, cent=(4.5,1.5), theta=1.57, b/a=1.000, color=1, label=1, touch=1, parent=0  
(2) area=48, cent=(5.3,4.4), theta=-0.06, b/a=0.786, color=0, label=2, touch=1, parent=0  
(3) area=24, cent=(6.2,4.8), theta=0.06, b/a=0.690, color=1, label=3, touch=0, parent=2  
(4) area=2, cent=(7.5,6.0), theta=0.00, b/a=0.000, color=0, label=4, touch=0, parent=3  
(5) area=2, cent=(2.5,7.0), theta=0.00, b/a=0.000, color=1, label=5, touch=0, parent=2
```

DETECCIÓN DE LÍNEAS

En clases anteriores hemos estudiado algunos métodos para detectar bordes en imágenes, sin embargo, esta detección no es suficiente para calificar si el borde es o no de importancia para nuestro análisis.

La aplicación para la cual utilizamos un sistema de visión define a fin de cuentas que ‘forma’ de borde es de interés.



DETECCIÓN DE LÍNEAS

En entornos urbanos, es usual que una imagen este cargada de líneas rectas, dada la intervención del hombre en el panorama (edificios, vías, carteles, etc).

Detectar líneas nos permitirá resaltar los bordes que realmente pertenecen a un objeto de interés, y desplazar los bordes generados por figuras del fondo de la imagen.

DETECCIÓN DE LÍNEAS: TRANSFORMACIÓN DE HOUGH

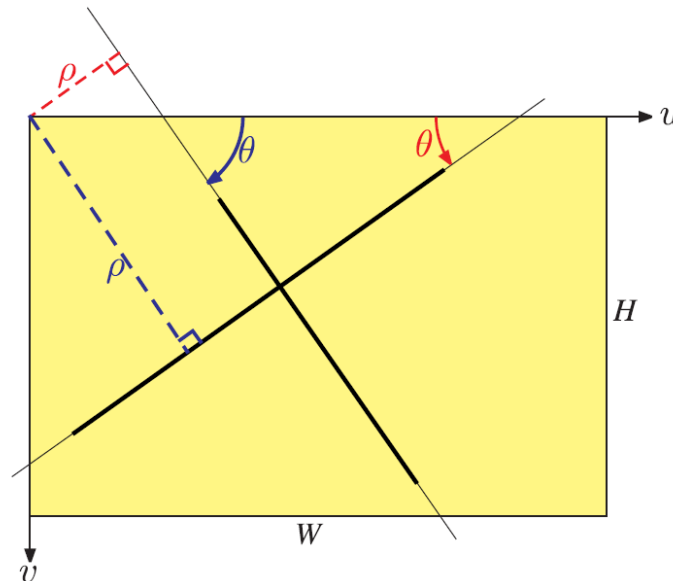
El método de Hough consiste en generar una matriz de ‘votos’, para definir cuales son las líneas de mayor relevancia dentro de la imagen.

- En primer lugar, definimos el método que utilizaremos para identificar cada posible recta de la imagen:

$$v = -u \tan \theta + \frac{\rho}{\cos \theta}$$

$$\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$$

$$\rho \in [-\rho_{\min}, \rho_{\max}]$$



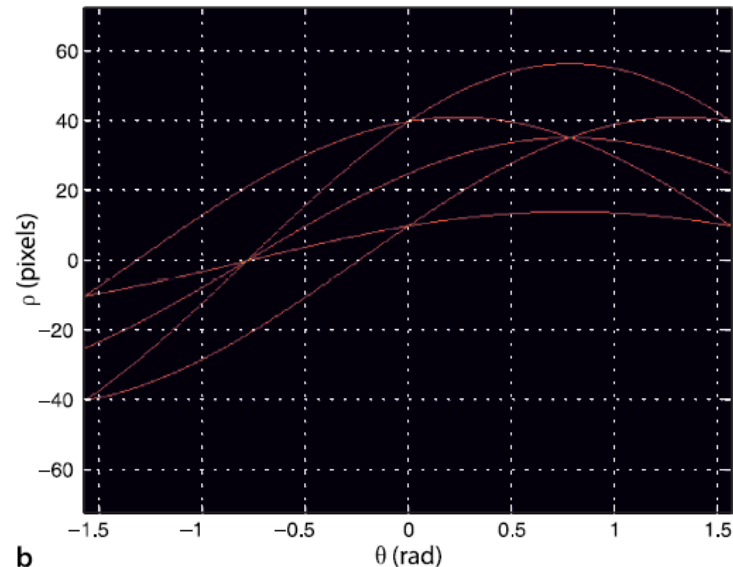
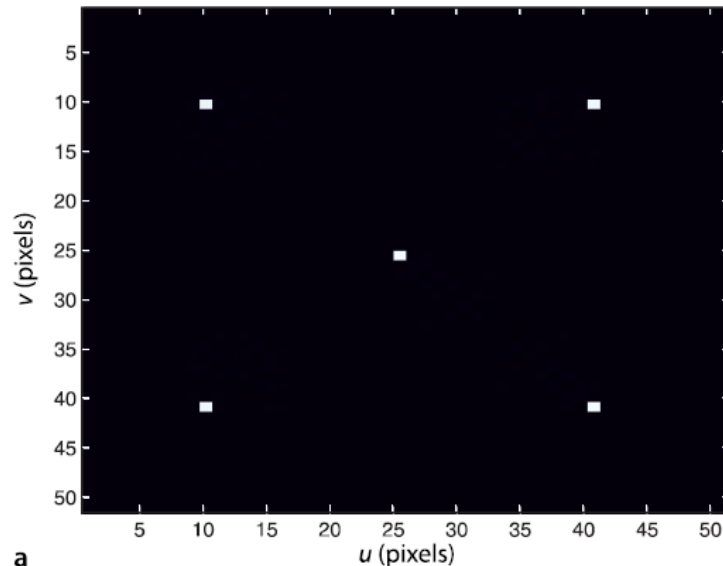
DETECCIÓN DE LÍNEAS: TRANSFORMACIÓN DE HOUGH

- Luego, para poder generar la matriz de votaciones, debemos discretizar el espacio de θ y ρ . El intervalo depende de la aplicación.

X	$-\frac{\pi}{2N} in$...	0	...	$\frac{\pi}{2N} ip$
ρ_{\min}					
...					
$\rho_{\min} + \Delta\rho i$					
...					
ρ_{\max}					

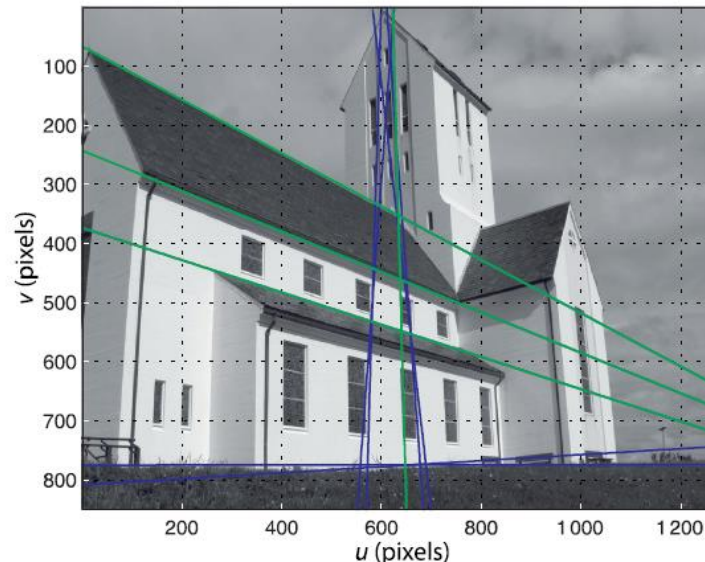
DETECCIÓN DE LÍNEAS: TRANSFORMACIÓN DE HOUGH

- Completamos la matriz de votos en base a la cantidad de píxeles que están resaltados en la imagen original, y pertenecen a cada línea proyectada.



DETECCIÓN DE LÍNEAS: TRANSFORMACIÓN DE HOUGH

- Por ultimo, seleccionamos los picos dentro de la matriz como líneas reales en la imagen, y resaltamos los resultados en la imagen original. Para esto tenemos dos opciones, resaltar las líneas calculadas, o filtrar la imagen para que solo puedan verse los pixeles positivos dentro de la línea calculada.



DETECCIÓN DE LÍNEAS

El procedimiento podría tener la necesidad de seguir procesando la imagen para llegar a resultados mas terminantes, por ejemplo, eliminando líneas múltiples, por criterio de numero de pixeles mínimos, y numero de pixeles seguidos mínimos.

Como siempre, las necesidades dependerán de la aplicación de interés.

TEMAS PENDIENTES DEL CAPITULO 13

Extracción de puntos y zonas de interés.

