DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Procesamiento y Análisis de Imágenes

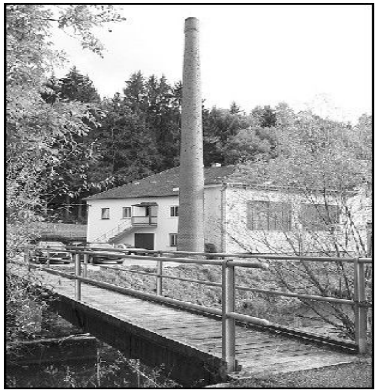
Violeta Chang

violeta.chang@usach.cl

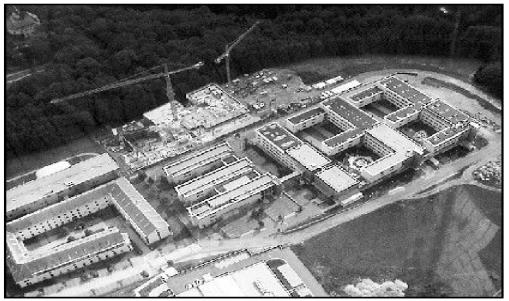
Créditos por slides: José M. Saavedra,
Aaron Bobick, Anne Solberg

Transformada de Hough







Detección de Formas

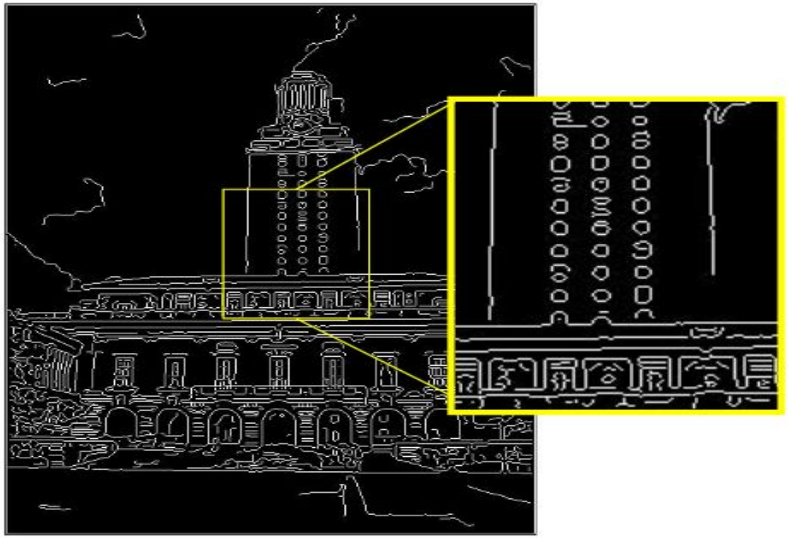
Transformada de Hough

- Why fit lines?

Many objects characterized by presence of straight lines

- Wait, why aren't we done just by running edge detection?

Transformada de Hough



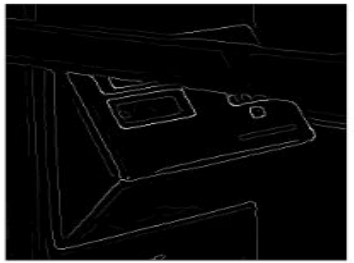
- **Extra** edge points (clutter), multiple models:
  - which points go with which line, if any?
- Only some parts of each line detected, and some parts are **missing:**
  - how to find a line that bridges missing evidence?
- **Noise** in measured edge points, orientations:
  - how to detect true underlying parameters?

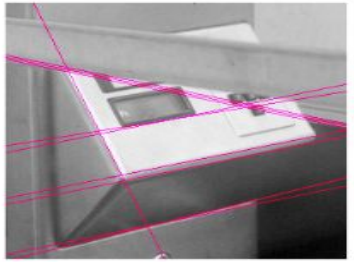
Transformada de Hough

Fitting lines

- Given points that belong to a line, what is the line?
- How many lines are there?
- Which points belong to which lines?

- **Hough Transform** is a voting technique that can be used to answer all of these
  - Main idea:
  - 1. Record all possible lines on which each edge point lies.
  - 2. Look for lines that get many votes.





Transformada de Hough

Detección de Líneas

Input: Una imagen de bordes (Sobel, Canny)

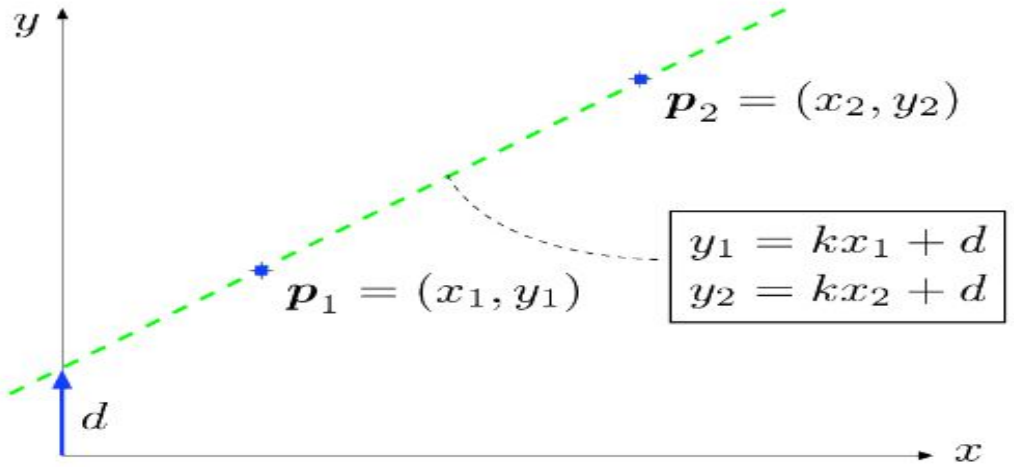
Transformada de Hough

- Detección de Líneas

Ecuación de una línea:

Transformada de Hough

- Detección de Líneas



Transformada de Hough

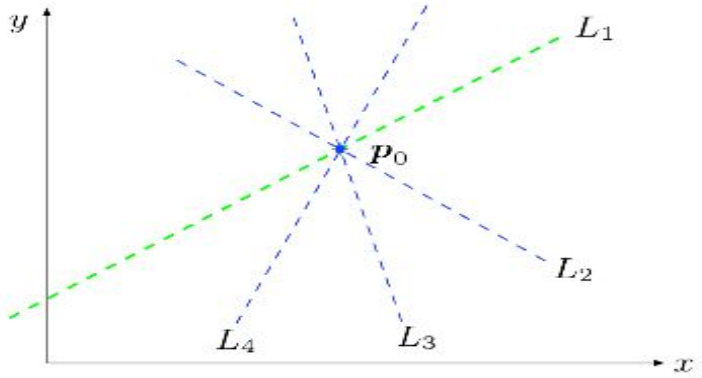
Detección de Líneas

El objetivo es encontrar valores de *k* y *d* tal que una gran cantidad de puntos caigan en la línea que ellos describen.

Transformada de Hough

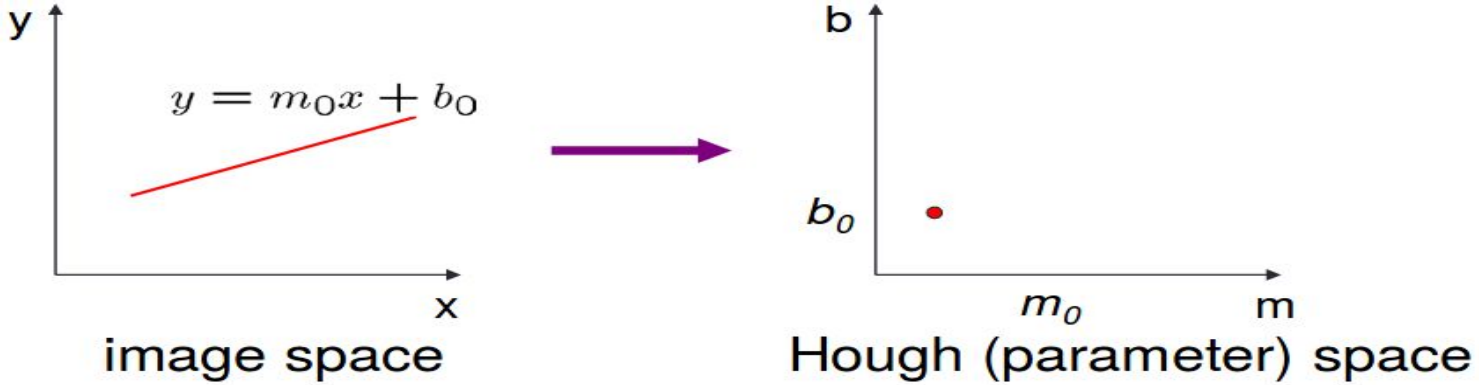
- Detección de Líneas
  - Una línea L_j=<k_j, d_j> que pasa a través de p_o cumple:

$$L_j : y_0 = k_j x_0 + d_j$$



Un punto puede pertenecer a muchas líneas

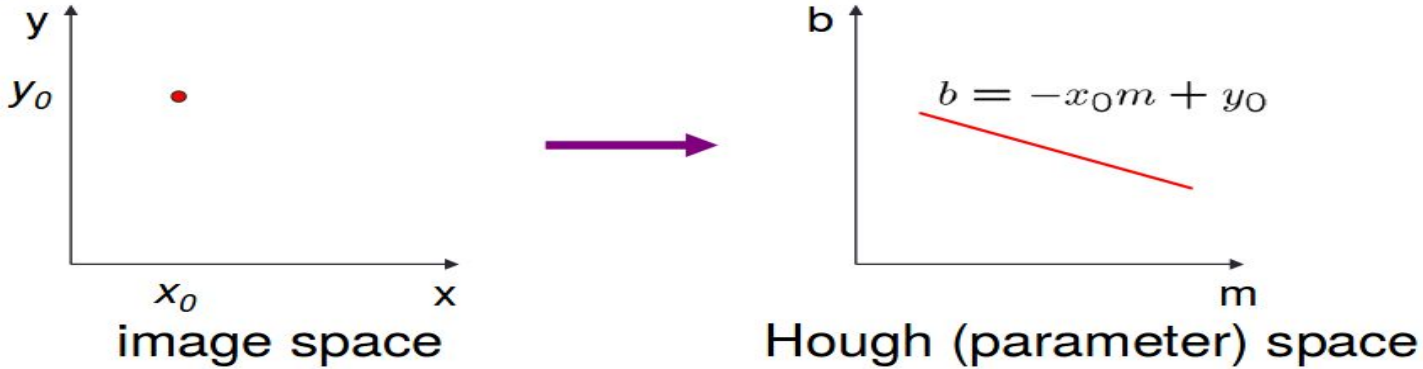
Transformada de Hough



Connection between image (x,y) and Hough (m,b) spaces

- A line in the image corresponds to a point in Hough space
- To go from image space to Hough space:
  - given a set of points (x,y), find all (m,b) such that y = mx + b

Transformada de Hough



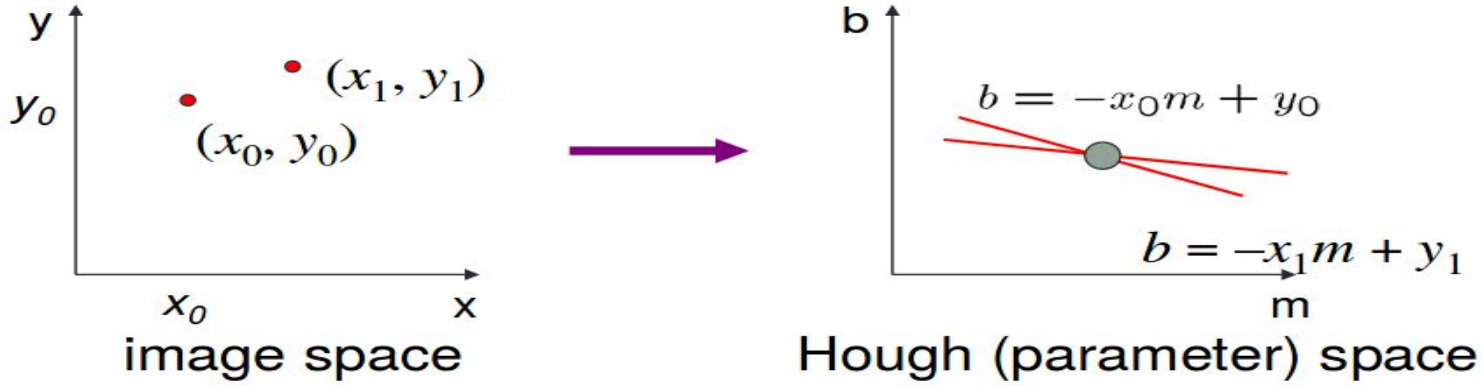
Connection between image (x,y) and Hough (m,b) spaces

What does a point (x_0, y_0) in the image space map to?

Answer: the solutions of $b = -x_0 m + y_0$

this is a line in Hough space

Transformada de Hough

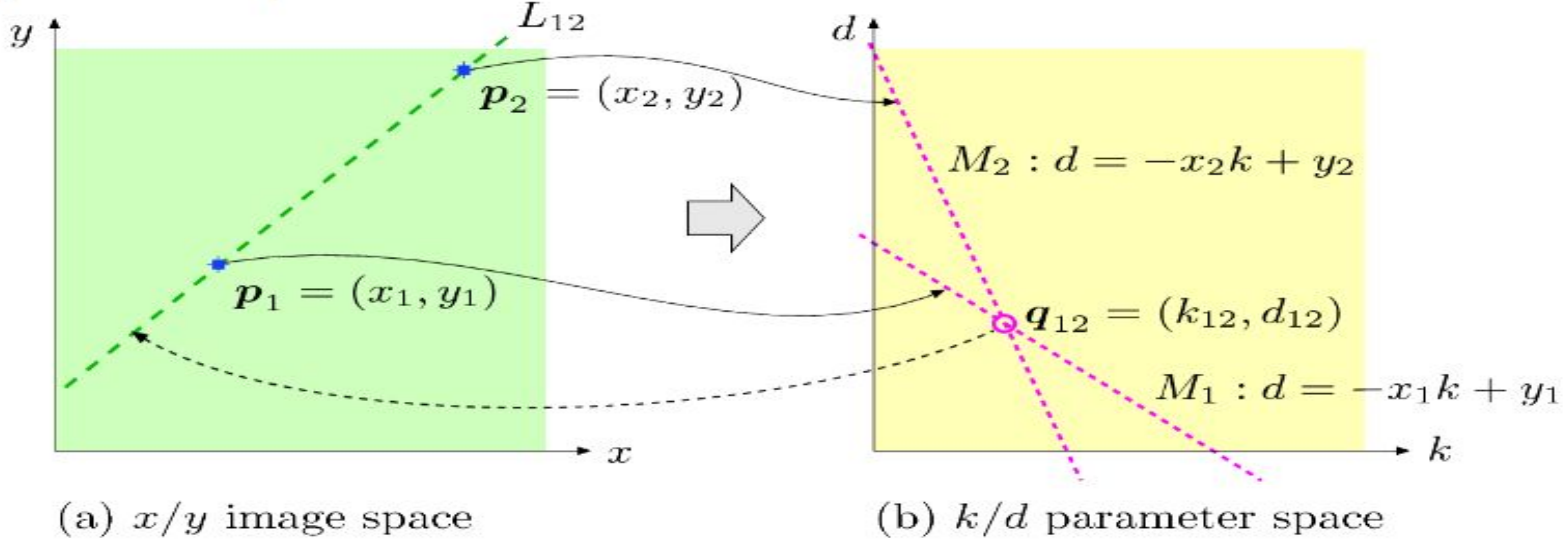


What are the line parameters for the line that contains both (x_0, y_0) and (x_1, y_1)?

- It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$

Transformada de Hough

- Detección de Líneas
  - Espacio de parámetros



(a) x/y image space (b) k/d parameter space

Transformada de Hough

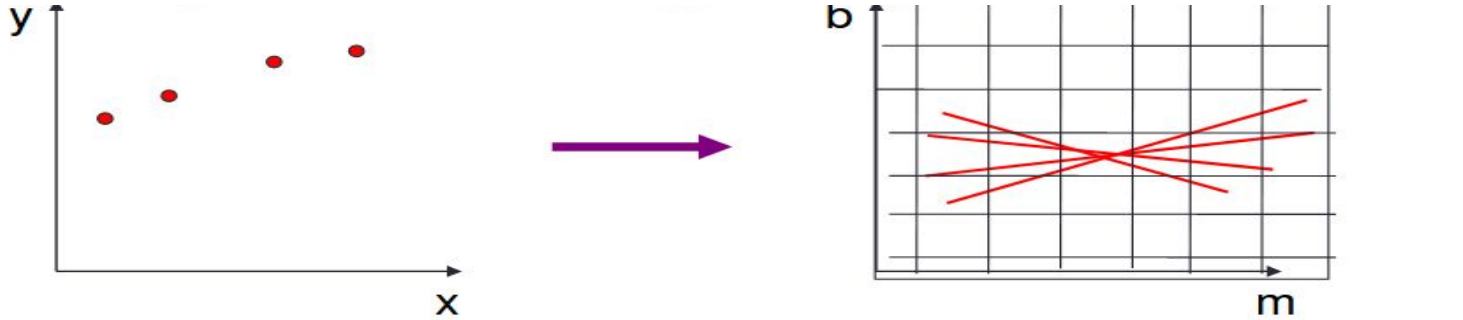


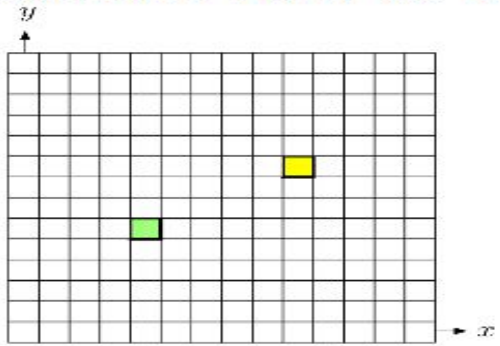
image space

Hough (parameter) space

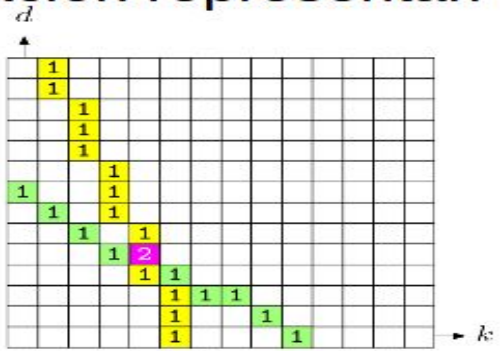
- How can we use this to find the most likely parameters (m,b) for the most prominent line in the image space?
- Let each edge point in image space vote for a set of possible parameters in Hough space
- Accumulate votes in discrete set of bins; parameters with the most votes indicate line in image space.

Transformada de Hough

- Detección de Líneas
  - Estrategia de Votación
    - Cuantizar los posible valores de *k* y *d*.
    - Crear un arreglo 2D que represente los valores de *k* y *d*.
    - Celdas con alto valor de votación representan líneas.



(a) Image Space



(b) Accumulator Array

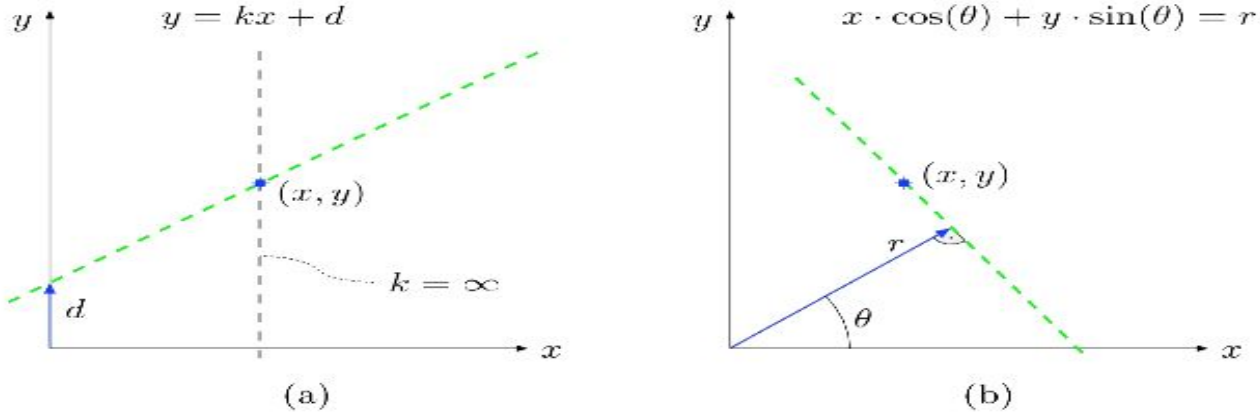
Transformada de Hough

- Detección de Líneas
  - Estrategia de Votación
    - ¿Cómo discretizar *k* y *d*?
    - Asumir como centro de coordenadas el centro de la imagen, *d* tomaría valores entre -H/2 y H/2 (H=height).
    - **k=??**
    - **k puede ser +inf!!!!**

Transformada de Hough

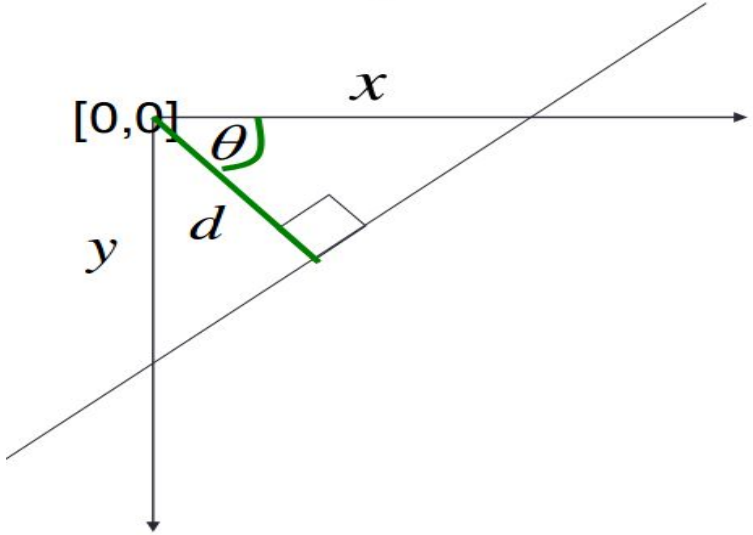
- Detección de Líneas
  - Hessian normal form (HNF)

$$x \cdot \cos(\theta) + y \cdot \sin(\theta) = r$$



Transformada de Hough

Issues with usual (*m*,*b*) parameter space: can take on infinite values, undefined for vertical lines.



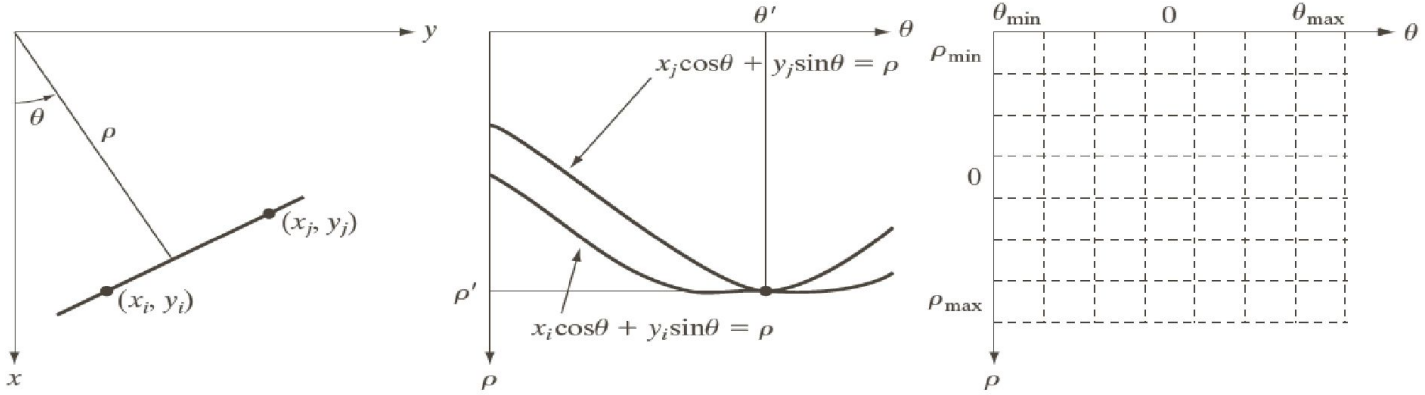
d : perpendicular distance from line to origin

θ : angle the perpendicular makes with the x-axis

$$x\cos\theta - y\sin\theta = d$$

Point in image space → sinusoid segment in Hough space

Transformada de Hough



a b c

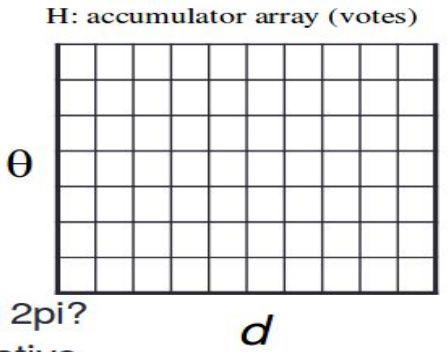
FIGURE 10.32 (a) (ρ, θ) parameterization of line in the xy-plane. (b) Sinusoidal curves in the $\rho\theta$-plane; the point of intersection (ρ', θ') corresponds to the line passing through points (x_i, y_i) and (x_j, y_j) in the xy-plane. (c) Division of the $\rho\theta$-plane into accumulator cells.

Transformada de Hough

Hough transform algorithm

Using the polar parameterization:

$$x\cos\theta - y\sin\theta = d$$



Basic Hough transform algorithm

1. Initialize H[d, θ]=0
2. for each edge point I[x,y] in the image
   for θ = 0 to 180 // some quantization; not 2pi?
   $d = x\cos\theta - y\sin\theta$ // maybe negative
   H[d, θ] += 1
3. Find the value(s) of (d, θ) where H[d, θ] is maximum
4. The detected line in the image is given by $d = x\cos\theta - y\sin\theta$

Space complexity? k^n (n dimensions, k bins each)

Time complexity (in terms of number of voting elements)?

Transformada de Hough

Example: Hough transform for straight lines

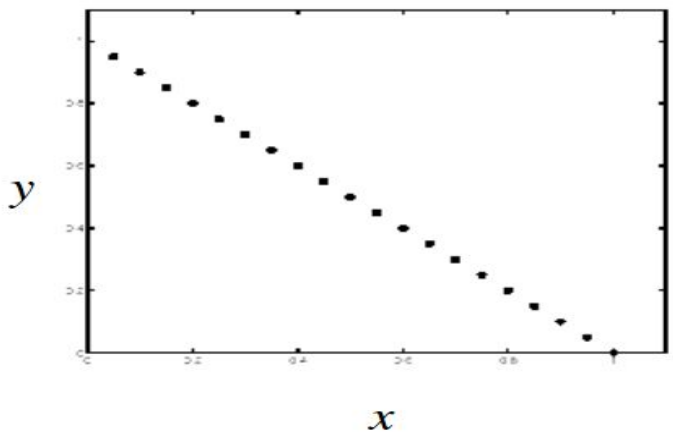
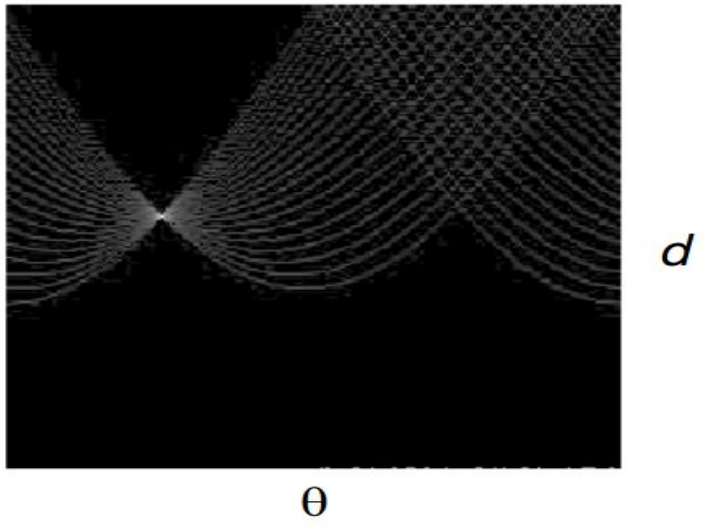


Image space
edge coordinates



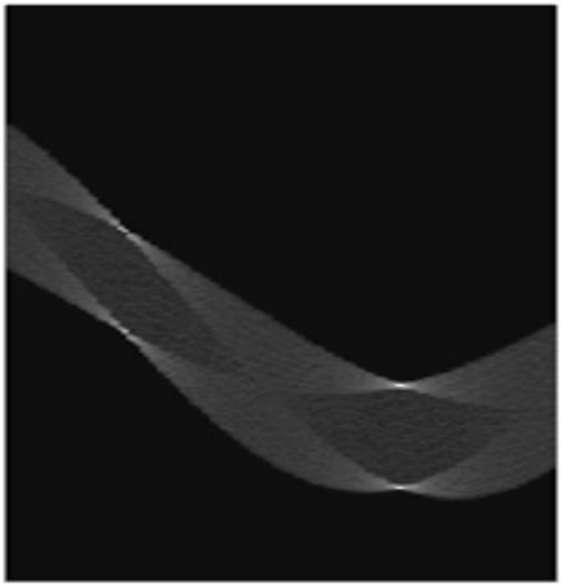
Votes

Bright value = high vote count
Black = no votes

Transformada de Hough

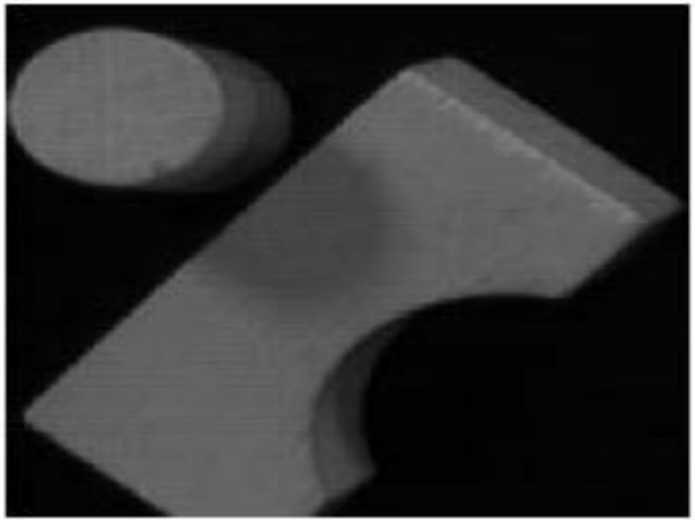
Example: Hough transform for straight lines

Square :



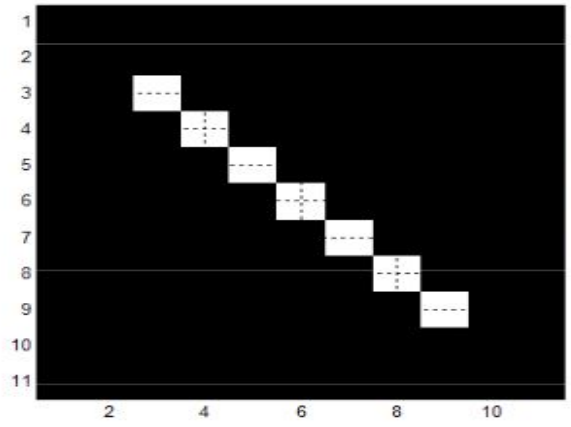
Transformada de Hough

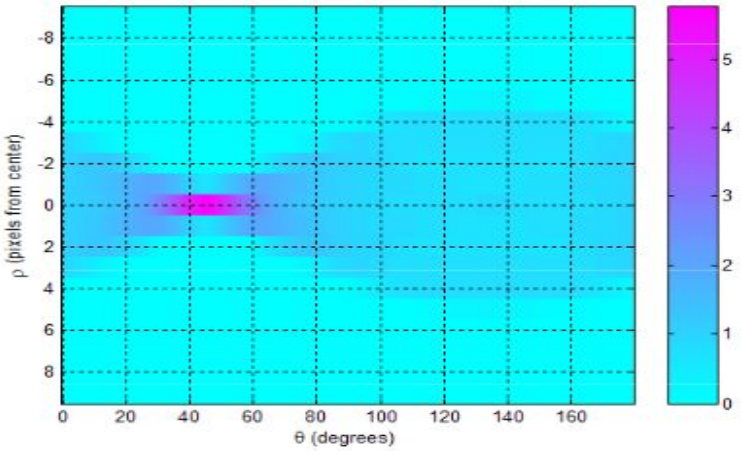
Example: Hough transform for straight lines



Transformada de Hough

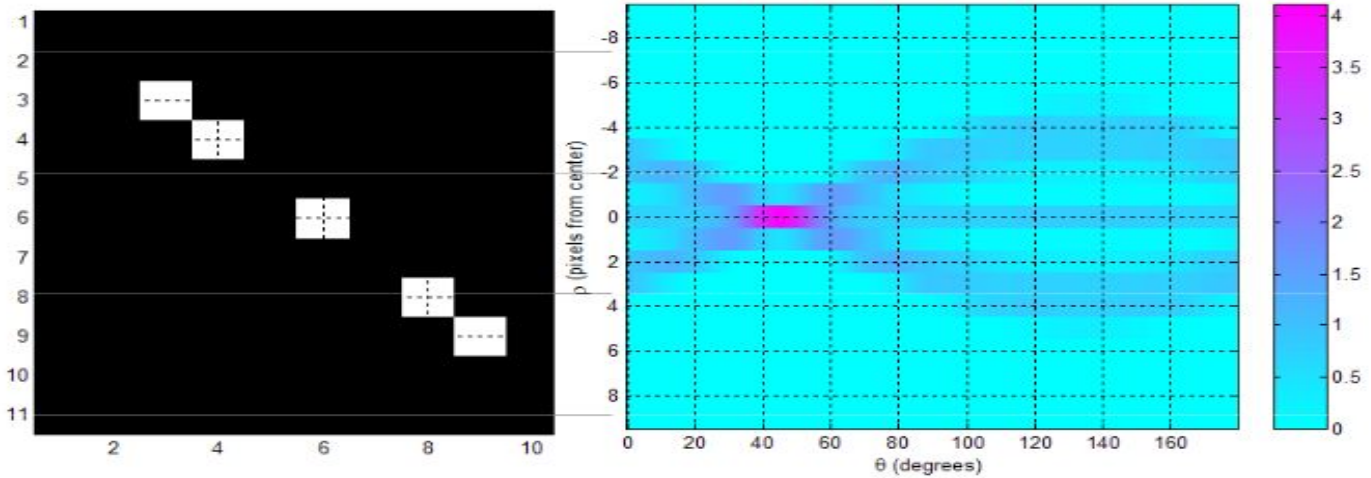
- Example 1: 11x11 image and its Hough transform:





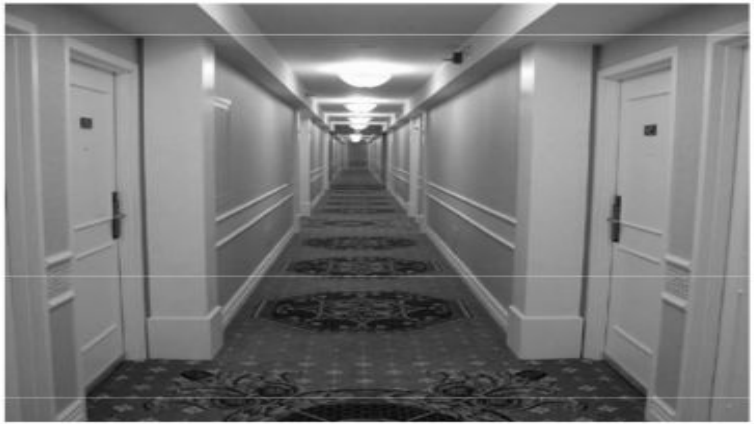
Transformada de Hough

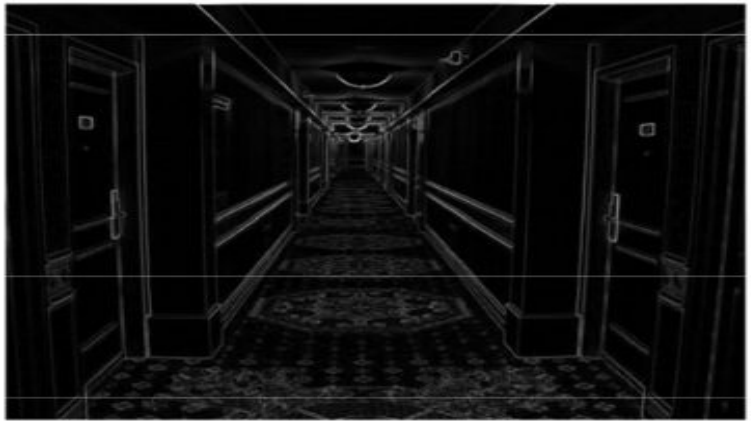
- Example 2: 11x11 image and its Hough transform:



Transformada de Hough

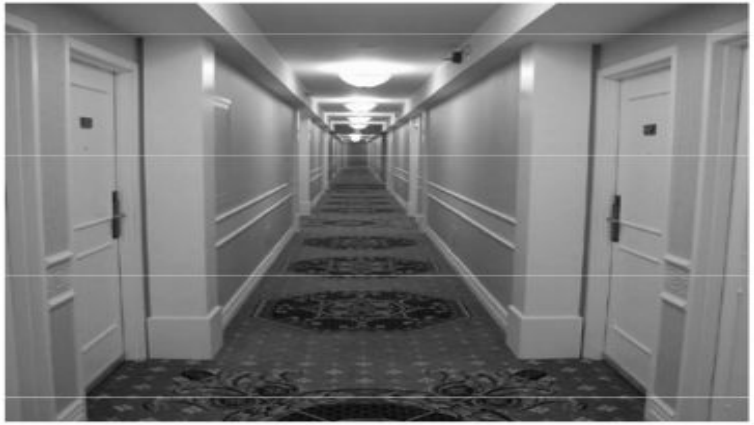
- Example 3: Natural scene and result of Sobel edge detection:

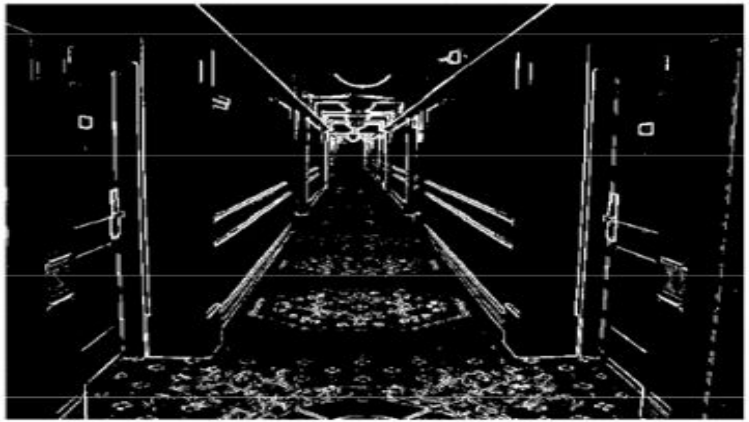




Transformada de Hough

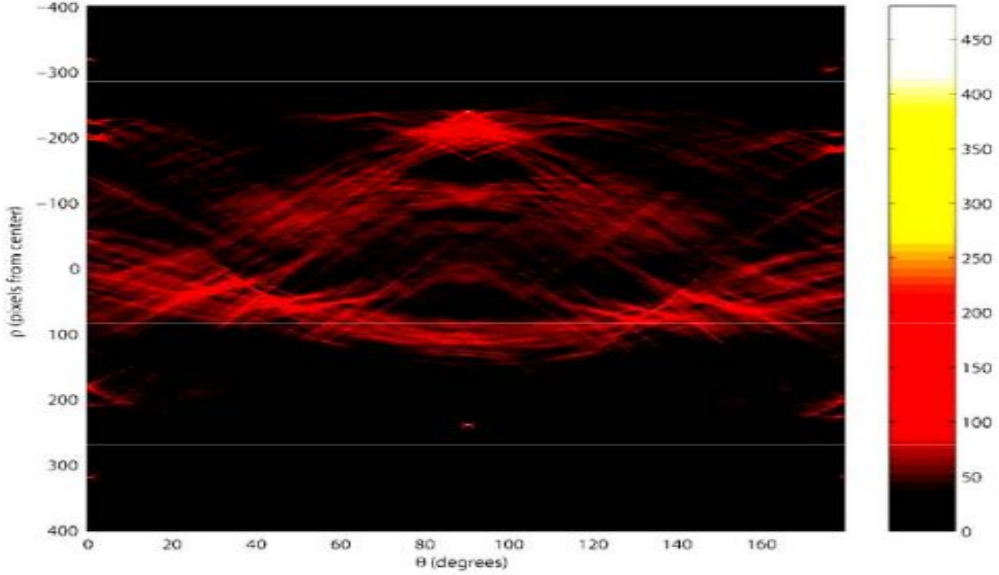
- Example 3: Natural scene and result of Sobel edge detection followed by thresholding:





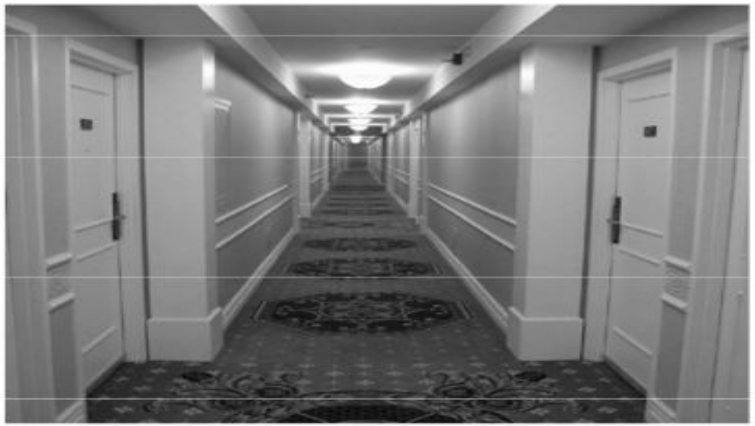
Transformada de Hough

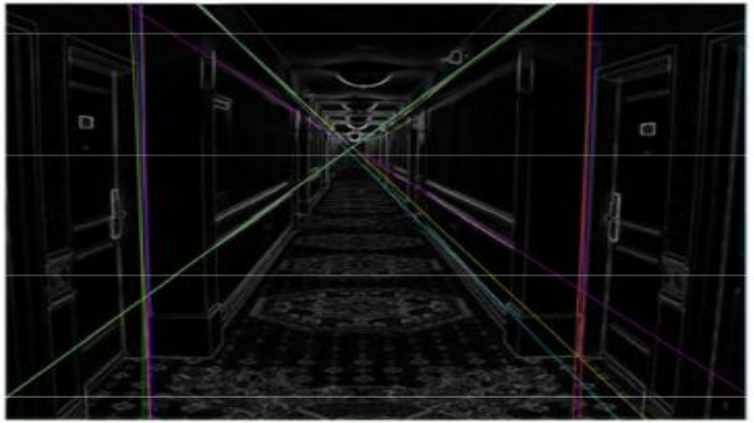
- Example 3: Accumulator matrix:



Transformada de Hough

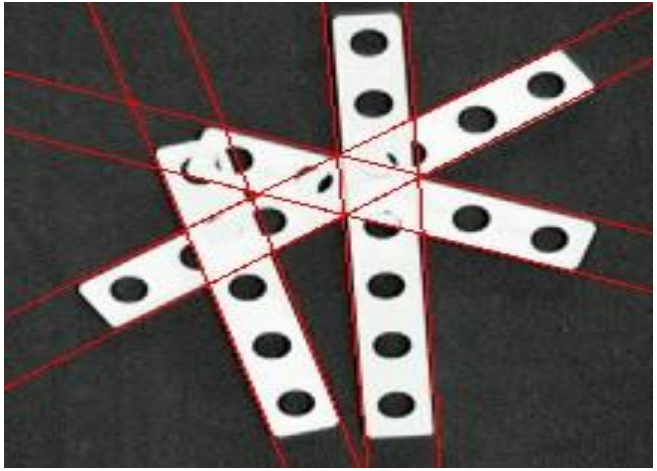
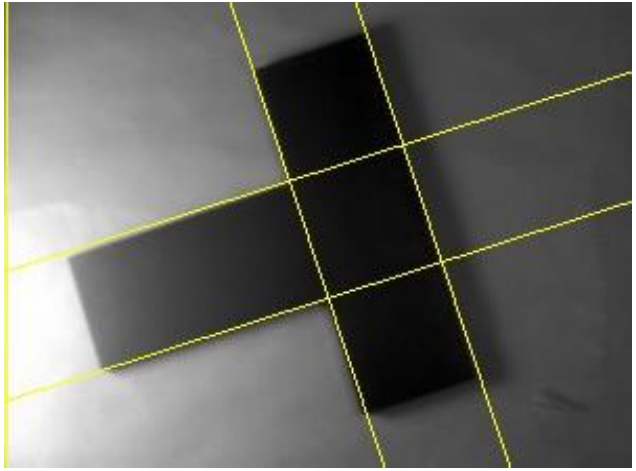
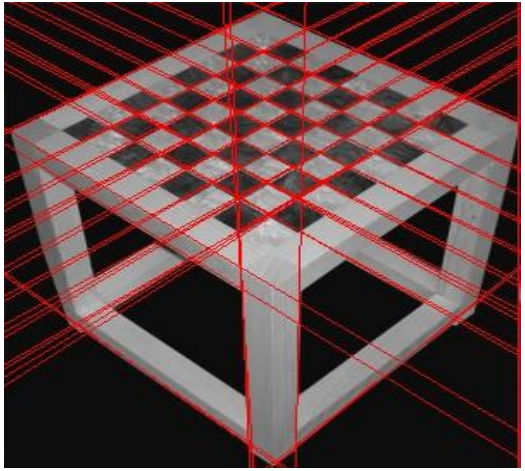
- Example 3: Original image and 20 most prominent lines:



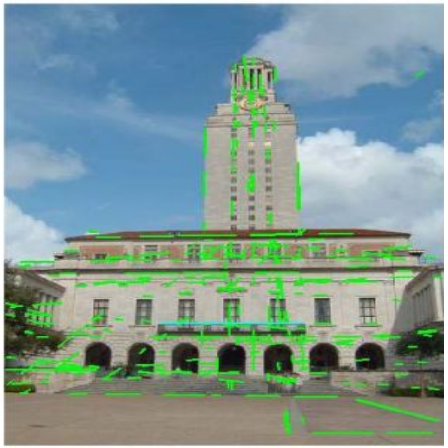


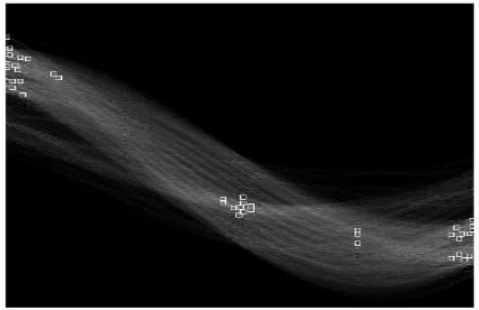
Transformada de Hough

Detección de líneas

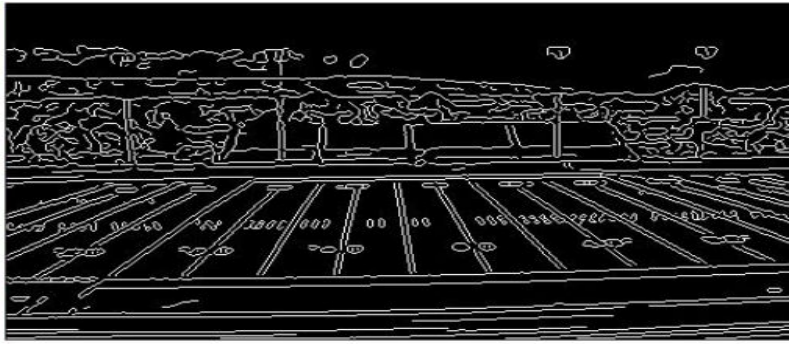




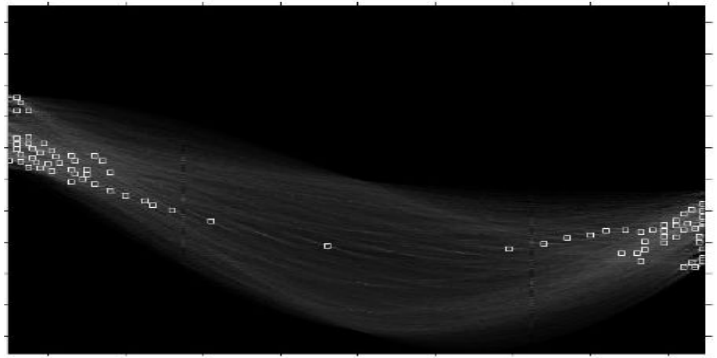
Transformada de Hough

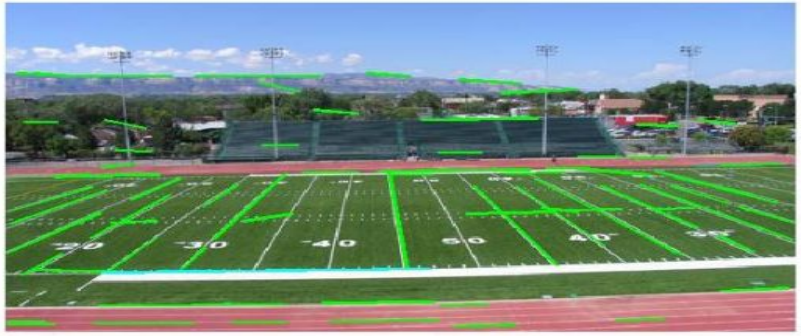




Transformada de Hough





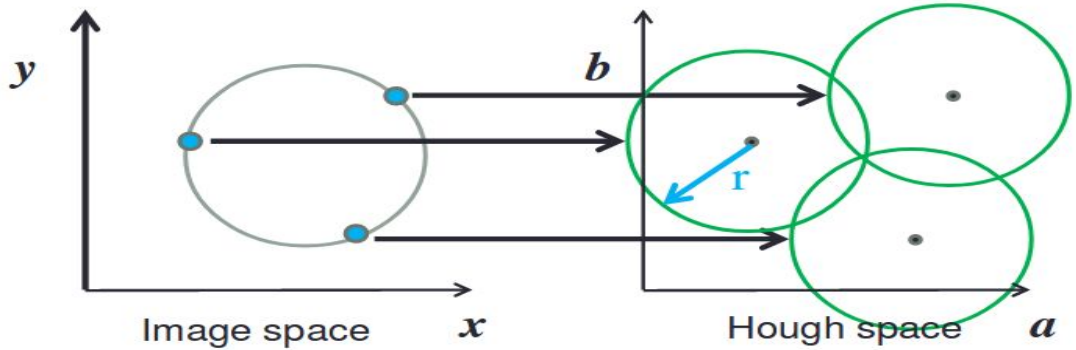


Showing longest segments found

Transformada de Hough

Hough transform for circles

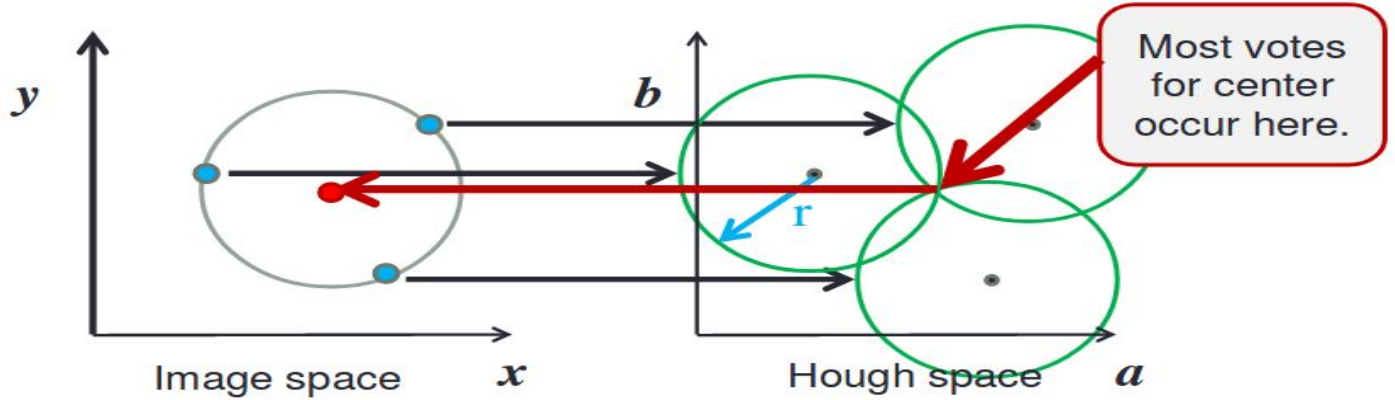
- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For a fixed radius r, unknown gradient direction:



Transformada de Hough

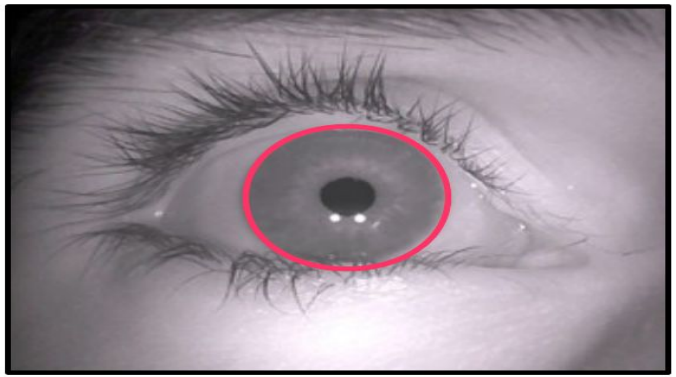
Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For a fixed radius r, unknown gradient direction:



Transformada de Hough

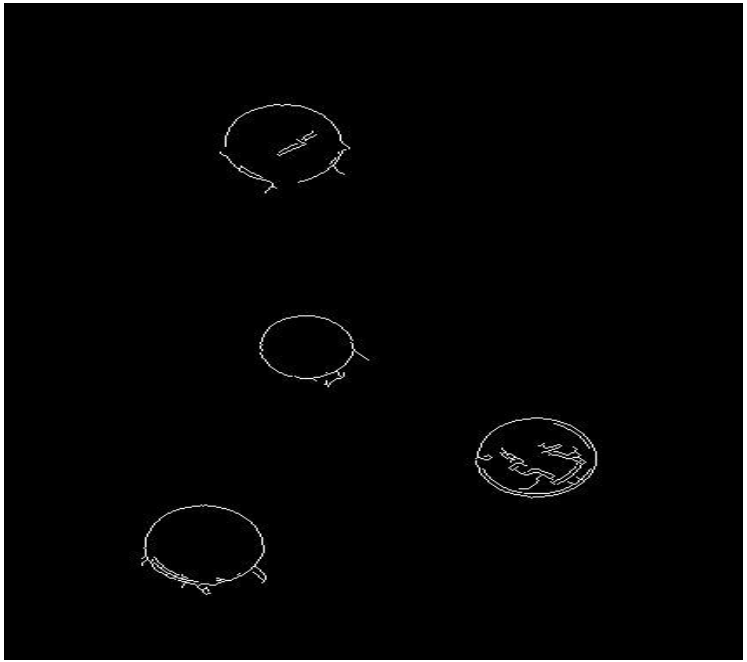
- Detección de Circunferencias
  - Eficiente cuando se conoce el rango de variación del radio.



Transformada de Hough

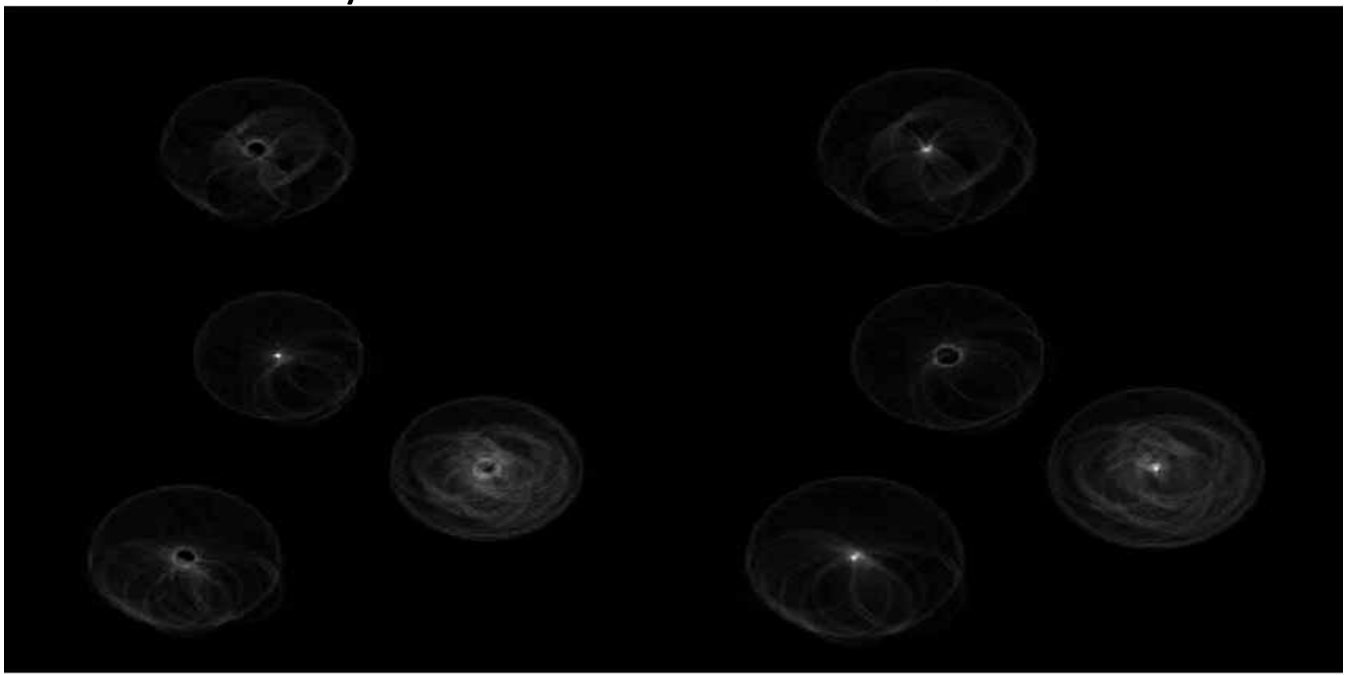
Original

Edges



Transformada de Hough

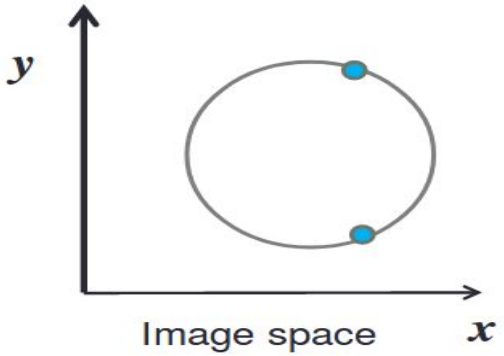
Penny Quarters

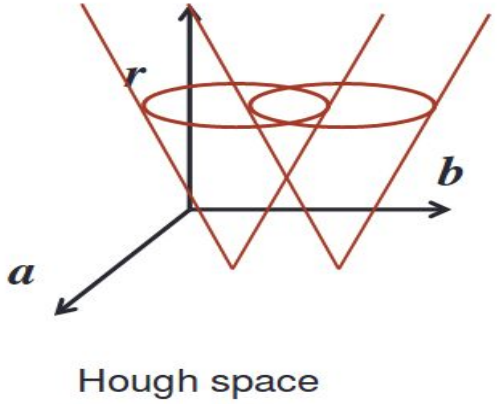


Transformada de Hough

Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For ***unknown*** radius r, no gradient:

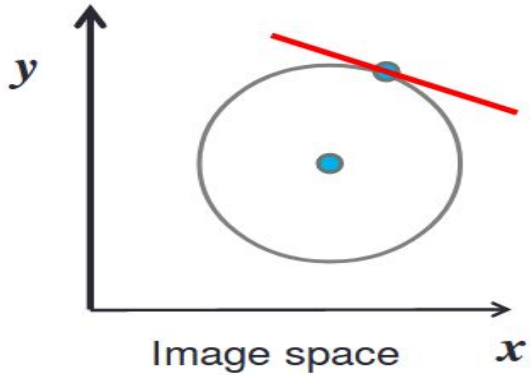


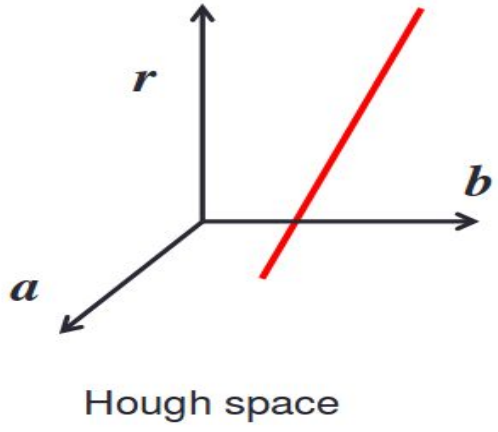


Transformada de Hough

Hough transform for circles

- Circle: center (a,b) and radius r $(x_i - a)^2 + (y_i - b)^2 = r^2$
- For ***unknown*** radius r, with ***gradient***:

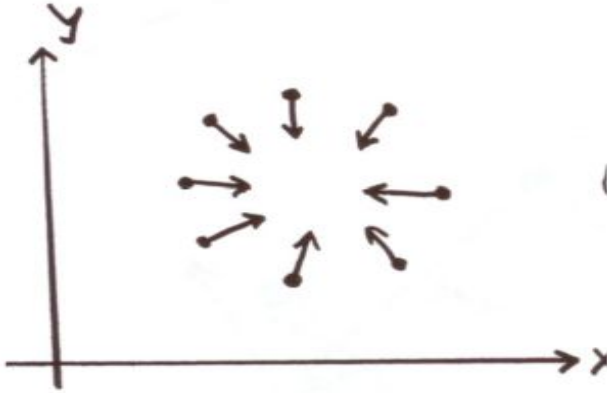




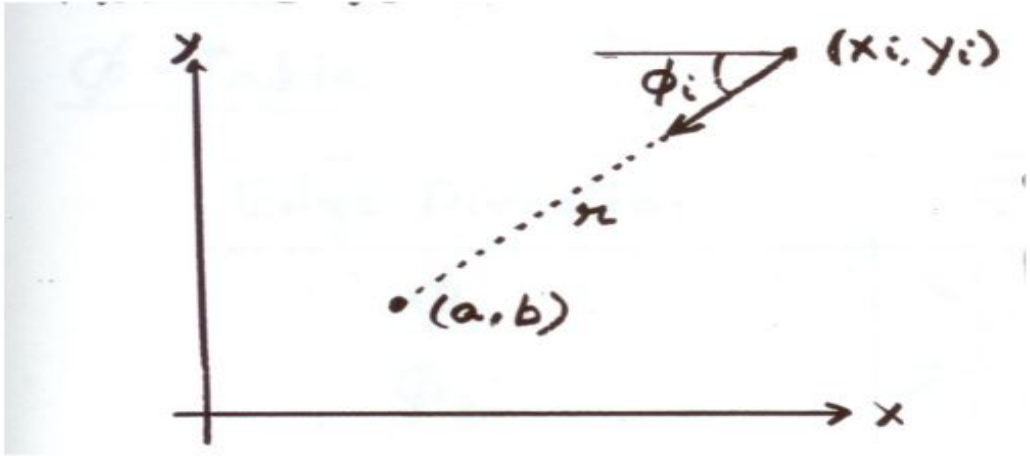
- Gradient information can save lot of computatio

Edge Location (x_i, y_i)

Edge Direction ϕ_i



Assume radius is known:



$$a = x - r\cos\phi$$

$$b = y - r\sin\phi$$

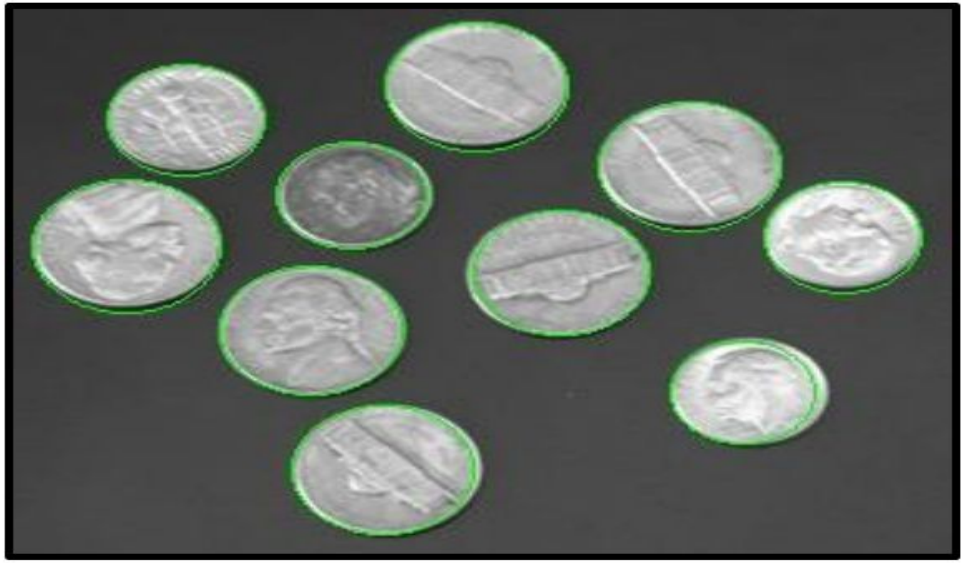
Need to increment only one point in Accumulator!!

Transformada de Hough

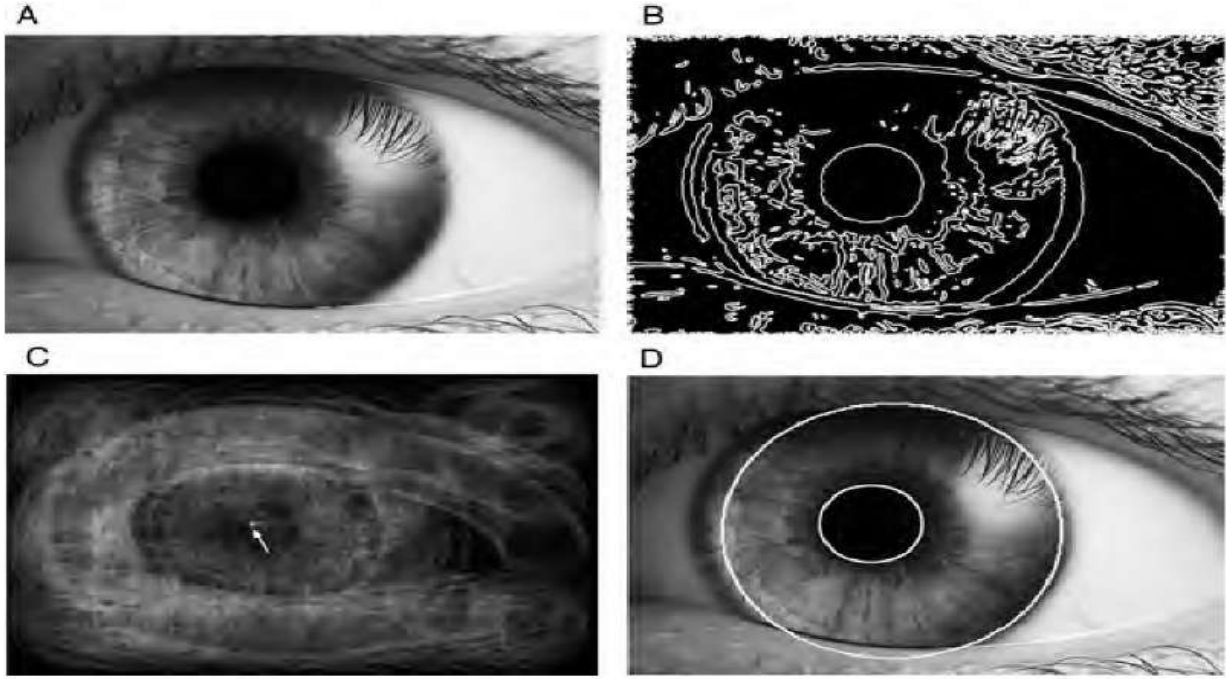
Hough transform for circles

For every edge pixel (x,y) :
  For each possible radius value r:
    For each possible gradient direction θ:
      %% or use estimated gradient
    $a = x - r\cos(\theta)$
    $b = y + r\sin(\theta)$
    $H[a,b,r]$ += 1
  end
end

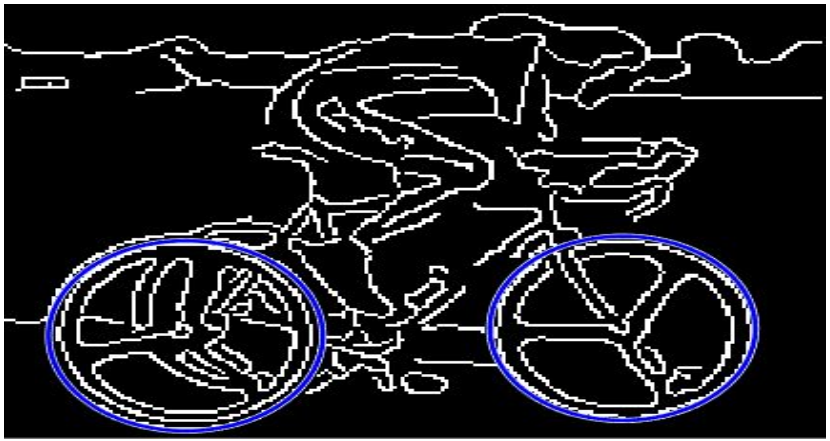
Transformada de Hough



Transformada de Hough



Transformada de Hough



Transformada de Hough