Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images.

It involves the development of a theoretical and algorithmic basis to achieve automatic visual understanding.

From the perspective of engineering, it seeks to automate tasks that the human visual system can do

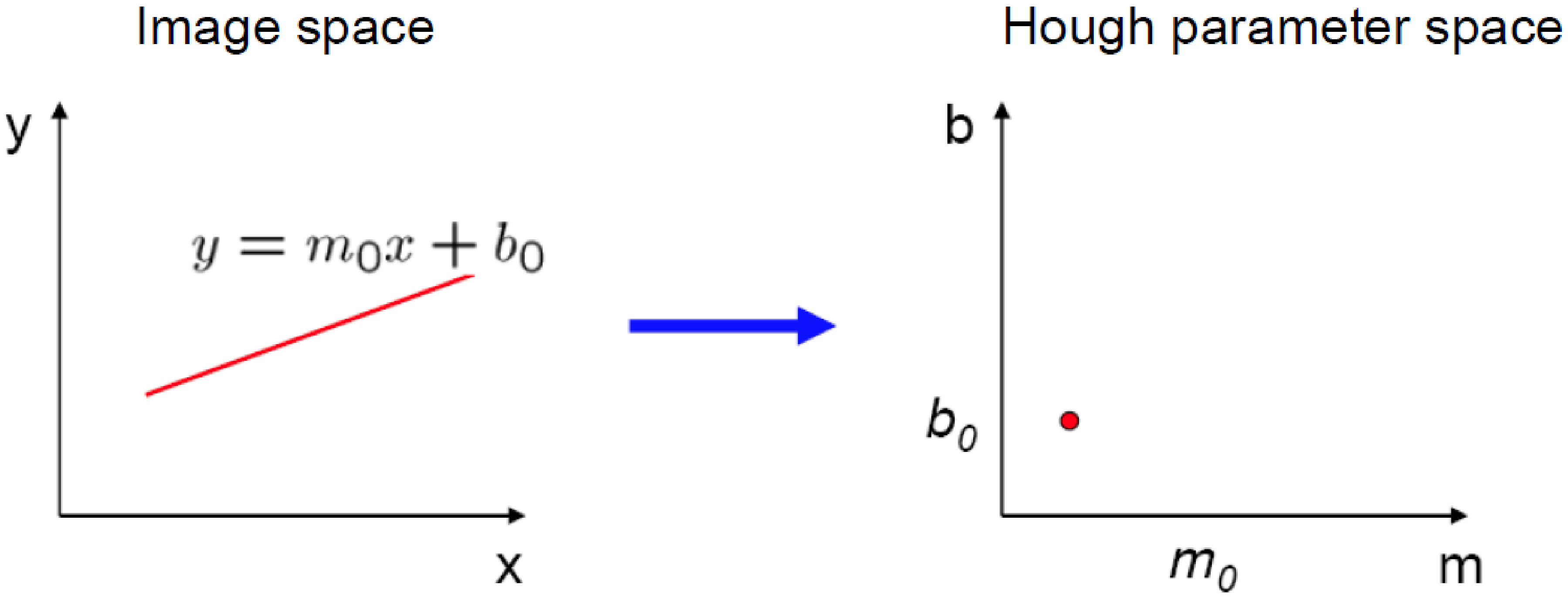
The applications of computer vision are numerous and include:

agriculture
augmented reality
autonomous vehicles
biometrics
character recognition
forensics
industrial quality inspection
face recognition
gesture analysis
geoscience
image restoration
medical image analysis
remote sensing
robotics

- The Hough transform (HT) can be used to detect lines, circles or other parametric curves.
- It was introduced in 1962 (Hough 1962) and first used to find lines in images a decade later (Duda 1972).
- The goal is to find the location of lines in images.
- This problem could be solved by e.g. Morphology and a linear structuring element, or by correlation.
  - Then we would need to handle rotation, zoom, distortions etc.
- Hough transform can detect lines, circles and other structures if their parametric equation is known.
- It can give robust detection under noise and partial occlusion.

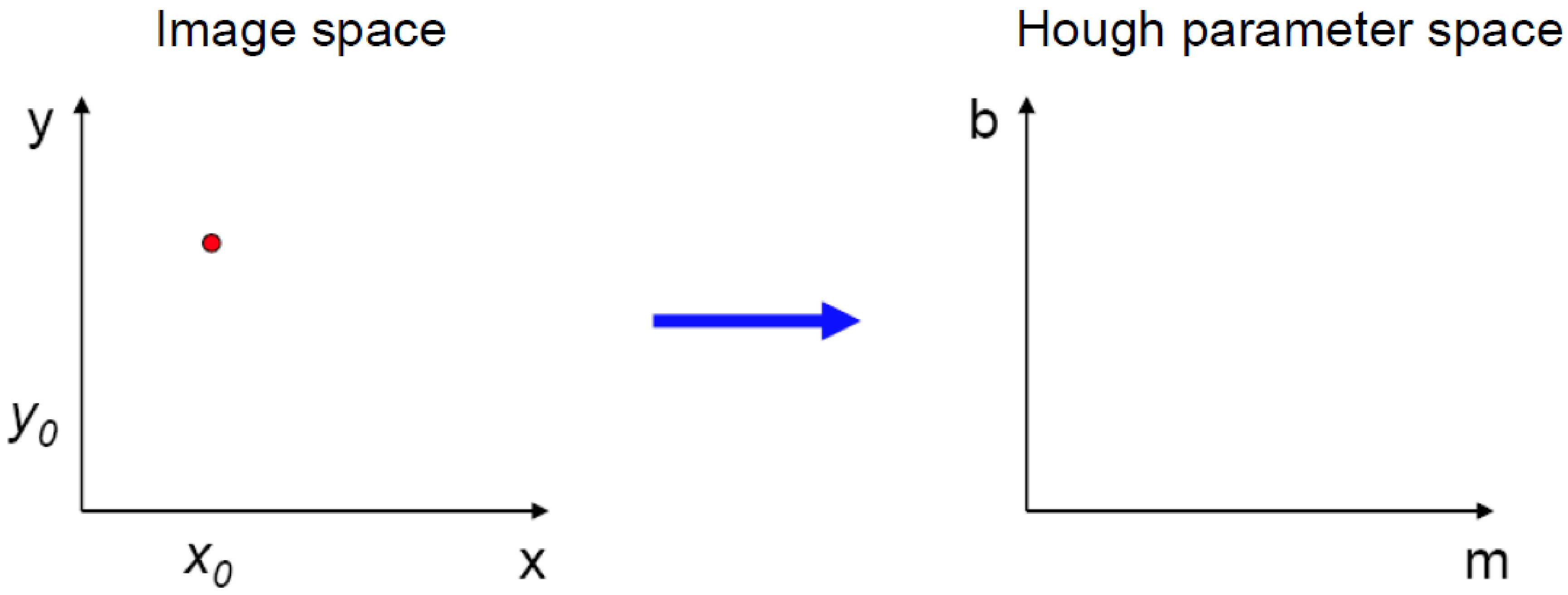
# Parameter space representation

- A line in the image corresponds to a point in Hough space



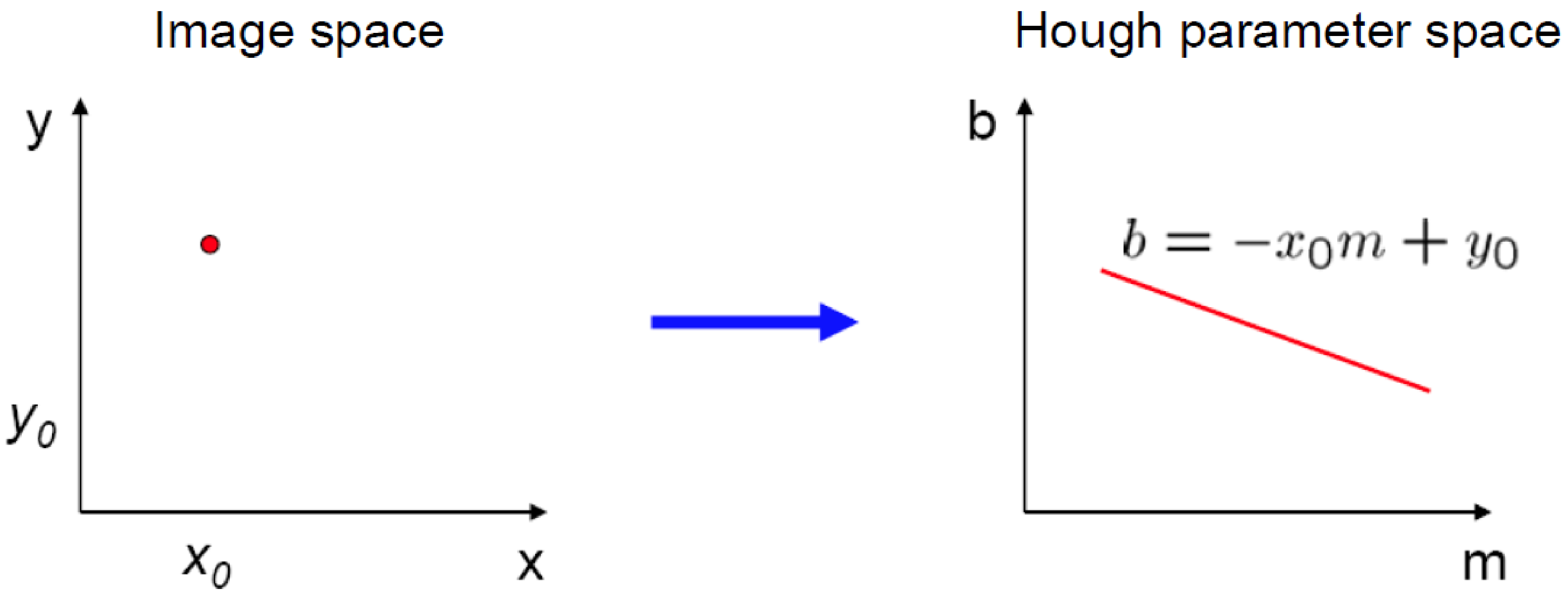
# Parameter space representation

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



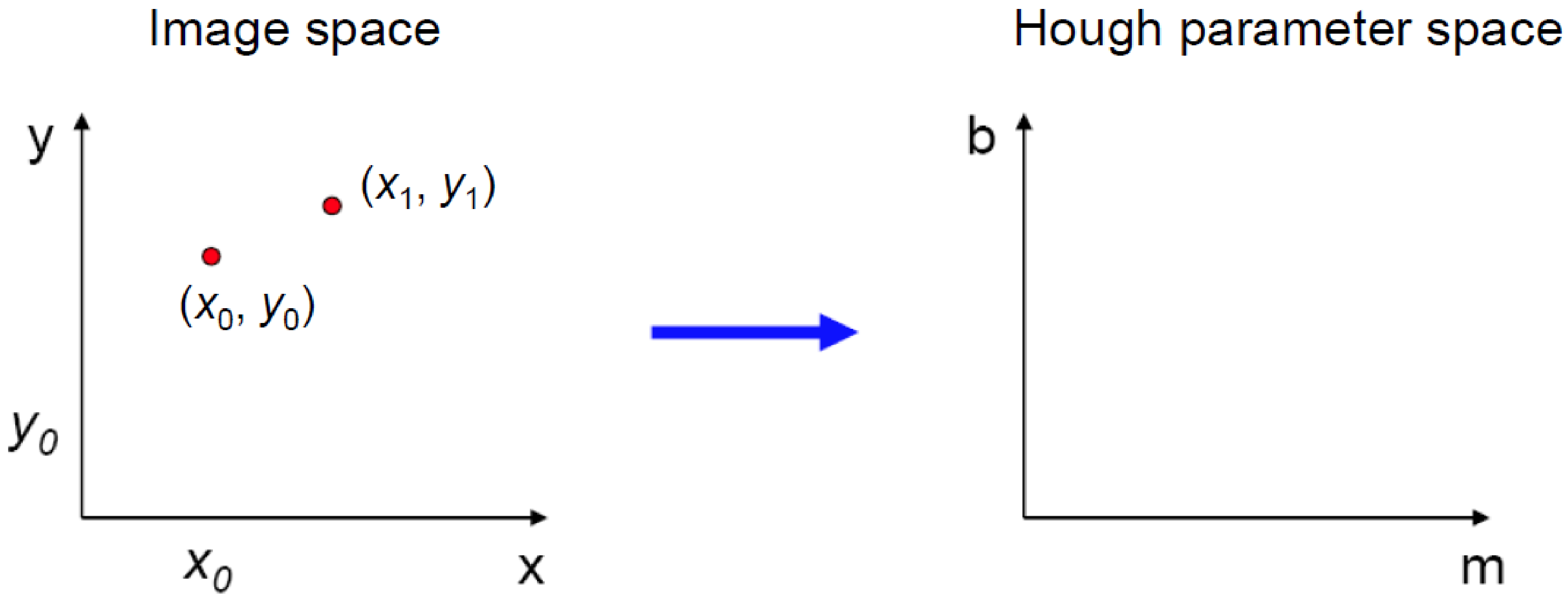
# Parameter space representation

- What does a point $(x_0, y_0)$ in the image space map to in the Hough space?
  - Answer: the solutions of $b = -x_0 m + y_0$
  - This is a line in Hough space



# Parameter space representation

- Where is the line that contains both $(x_0, y_0)$ and $(x_1, y_1)$?

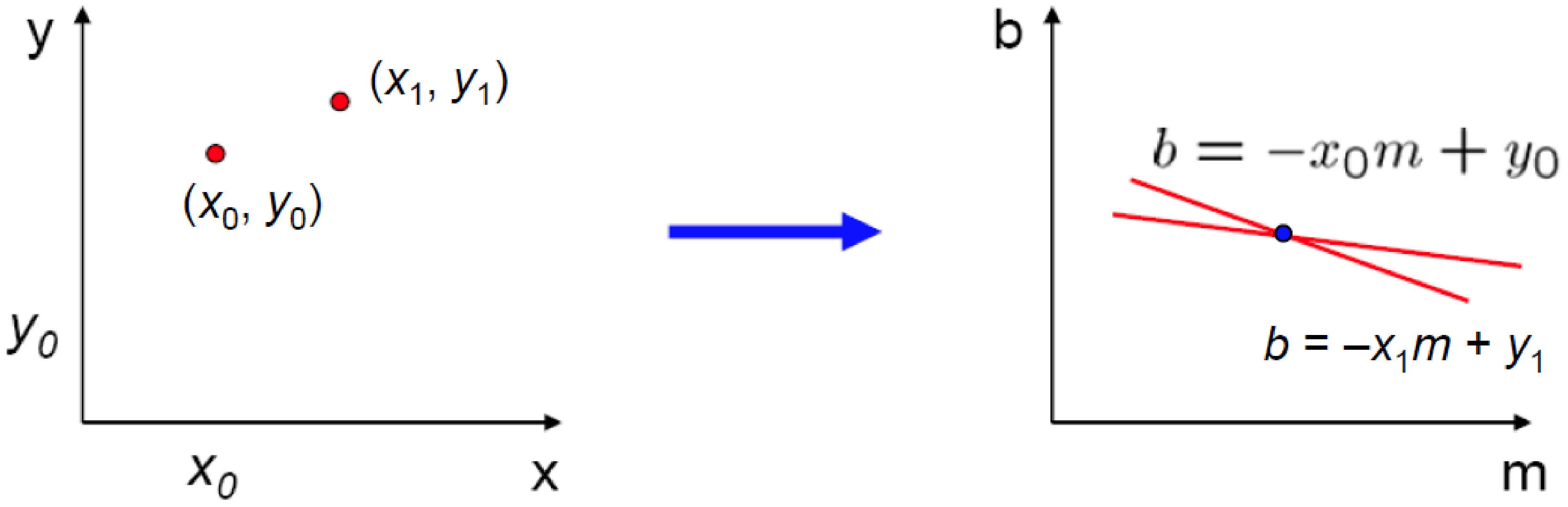


# Parameter space representation

- Where is the line that contains both $(x_0, y_0)$ and $(x_1, y_1)$?
  - It is the intersection of the lines $b = -x_0 m + y_0$ and $b = -x_1 m + y_1$

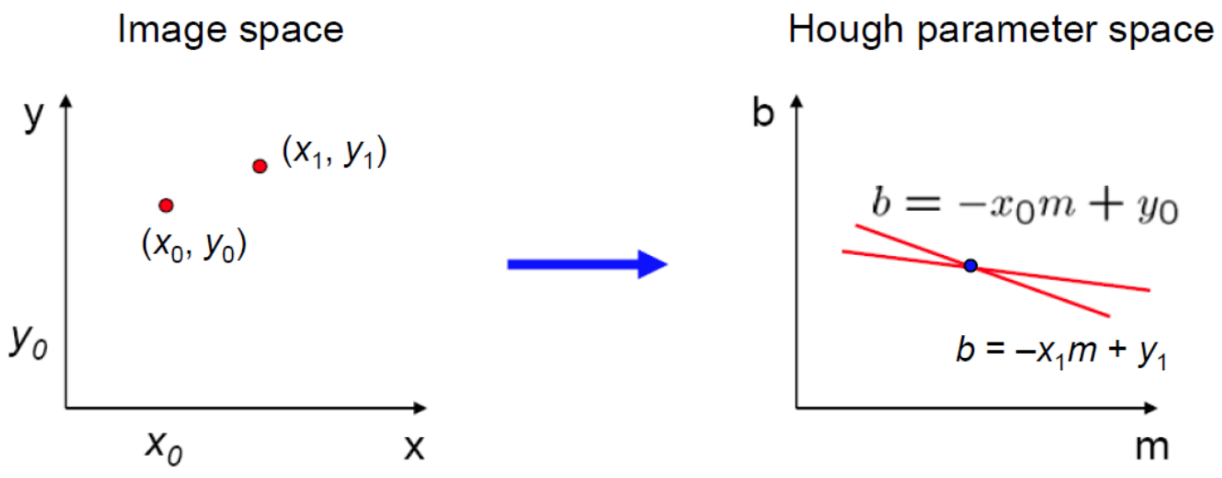
Image space

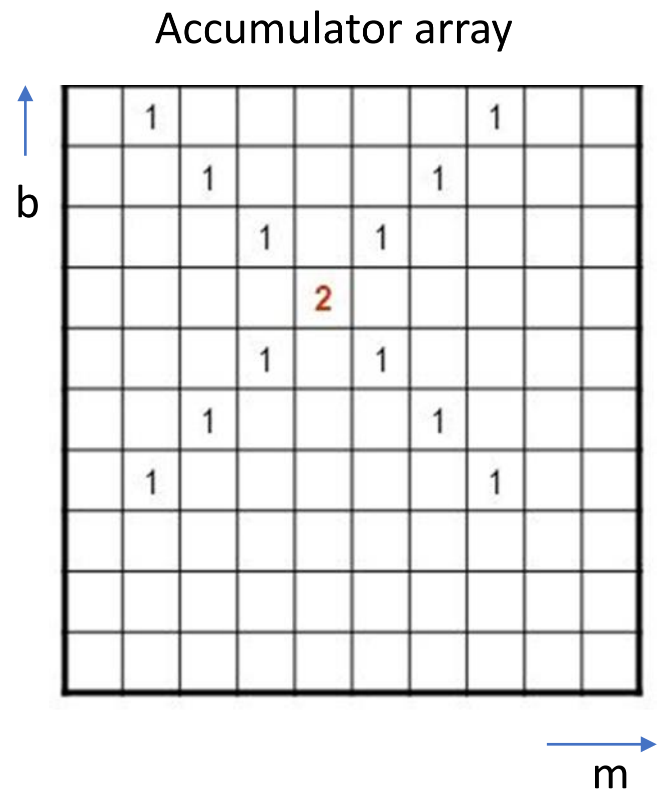
Hough parameter space



# Hough Transform Algorithm

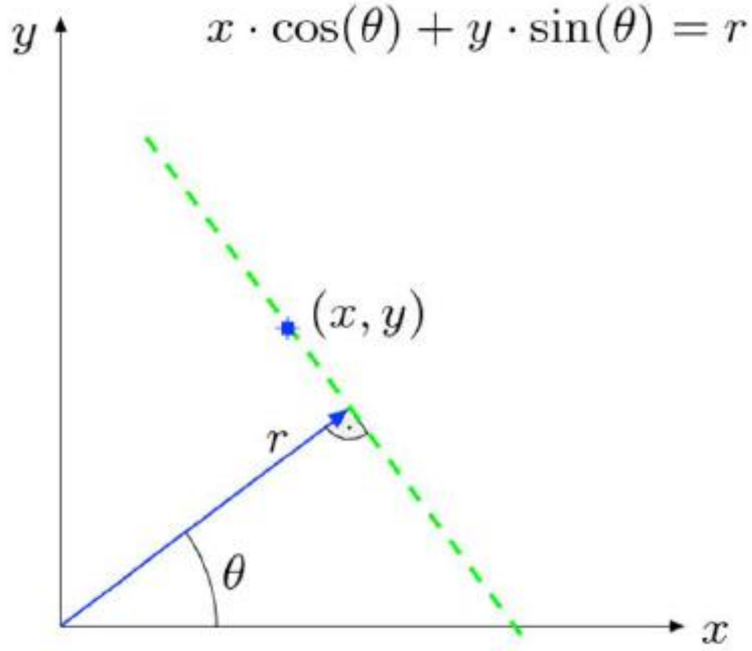
- Initialize an accumulator array A(m,b) to zero
- For each edge element (x,y), increment all cells that satisfy b = -x m + y



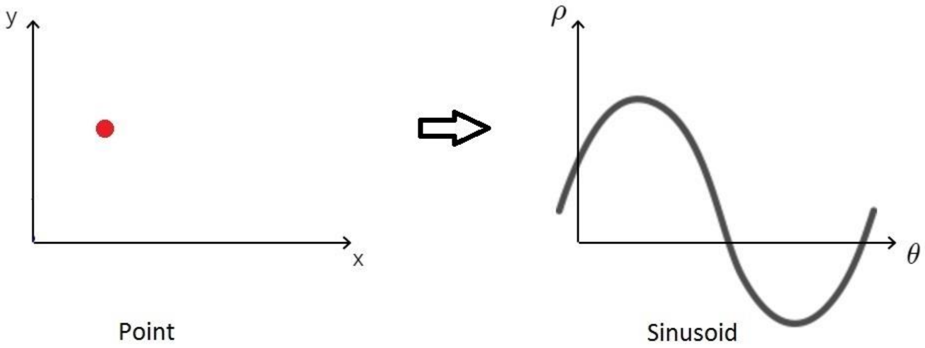



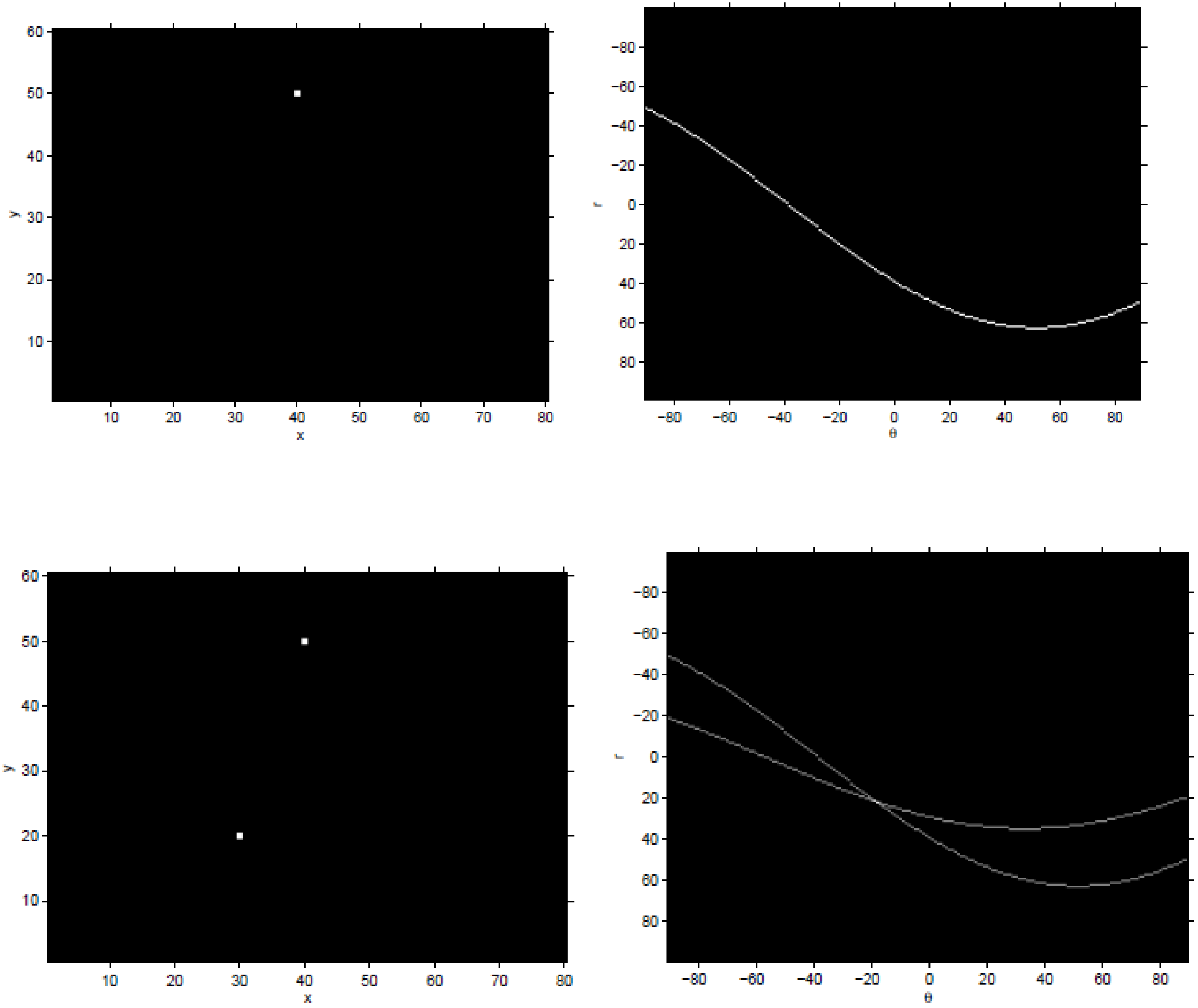
# Parameter space representation

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m
- Alternative: polar representation




- *r,* perpendicular distance of line from origin
- *Θ,* angle of line's perpendicular to *x* axis





y
x
r
θ
y
x
r
θ

# Algorithm outline

- Initialize accumulator H to all zeros
- For each edge point (x,y) in the image

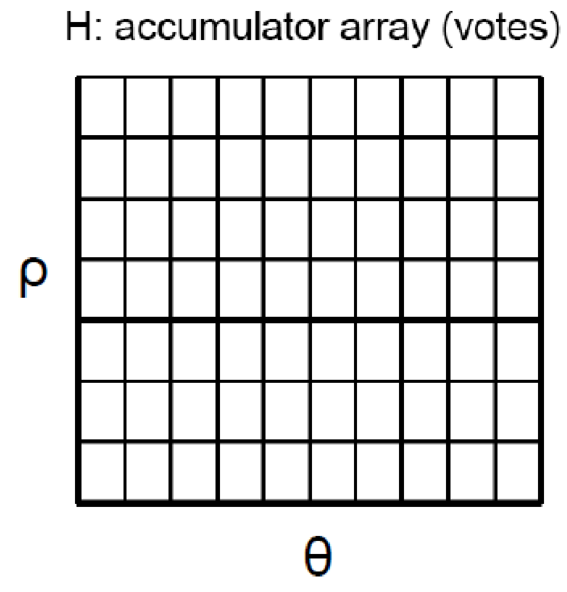
  For $\theta = 0$ to 180

  $\rho = x \cos\theta + y \sin\theta$

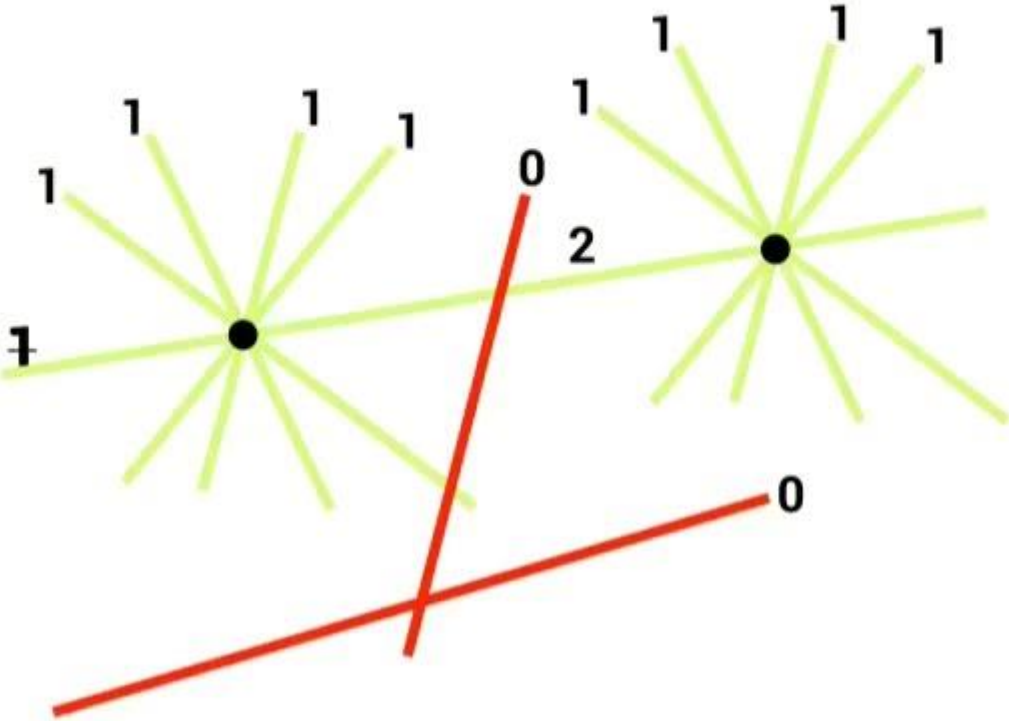
  $H(\theta, \rho) = H(\theta, \rho) + 1$

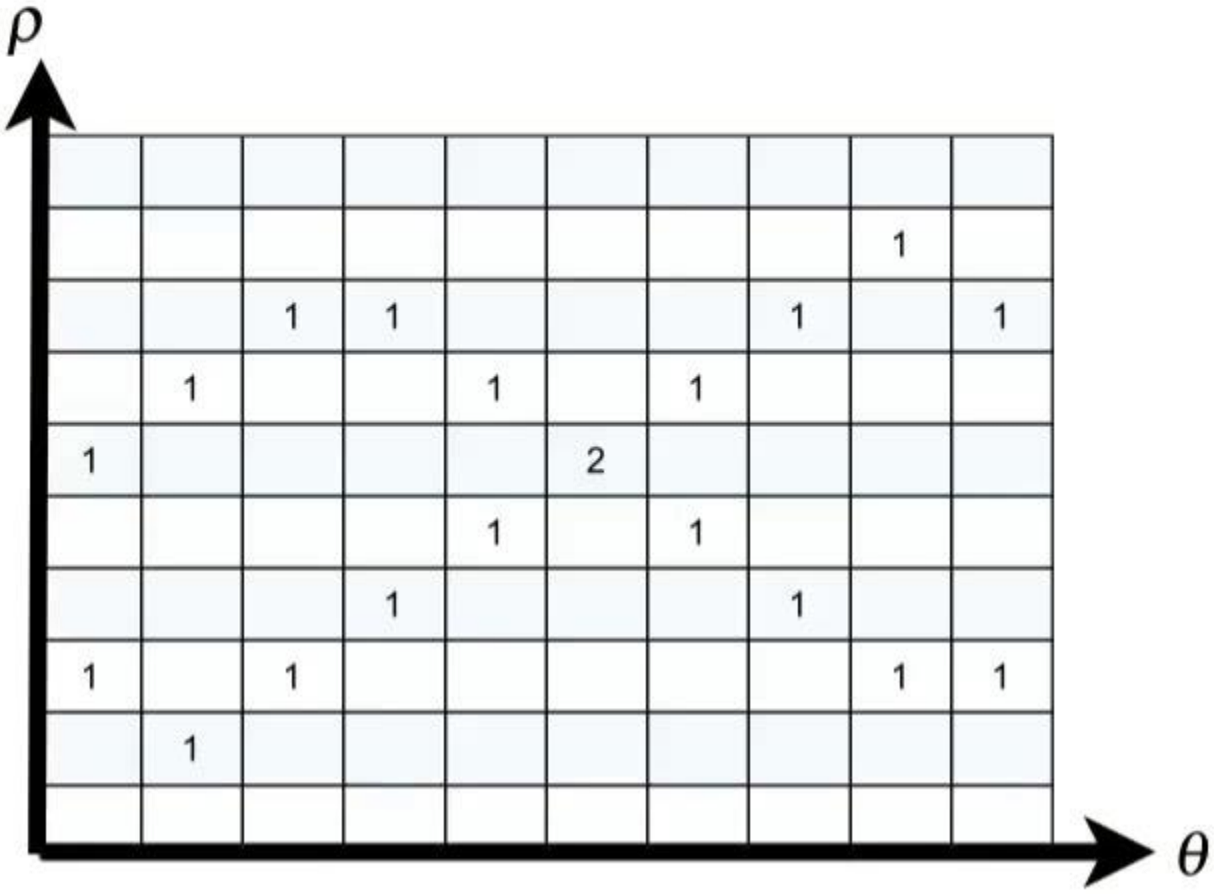
  end

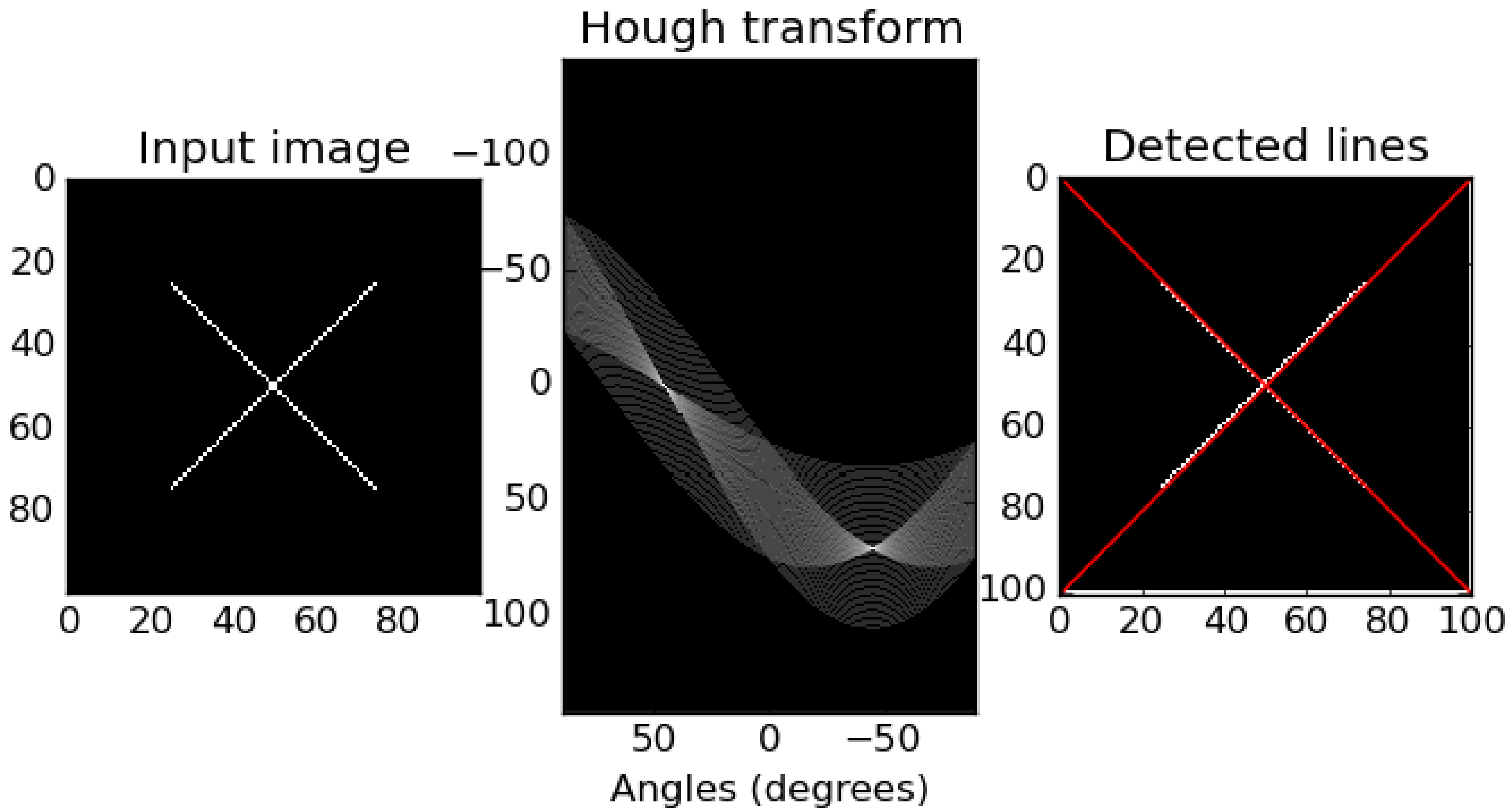
  end
- Find the value(s) of $(\theta, \rho)$ where $H(\theta, \rho)$ is a local maximum
  - The detected line in the image is given by $\rho = x \cos\theta + y \sin\theta$



# Hough **voting**





Input image
Hough transform
Detected lines
Angles (degrees)

# An image with linear structures

- Borders between the regions are straight lines.
- These lines separate regions with different grey levels.
- Edge detection is often used as preprocessing to Hough transform.

# Repetition - Basic edge detection

- A thresholded edge image is the starting point for Hough transform.
- What does a Sobel filter produce?
- Approximation to the image gradient:

$$\nabla f(x)$$

- ...which is a vector quantity given by:

$$\nabla \mathbf{f}(x, y) = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

# Repetition – Edge magnitude

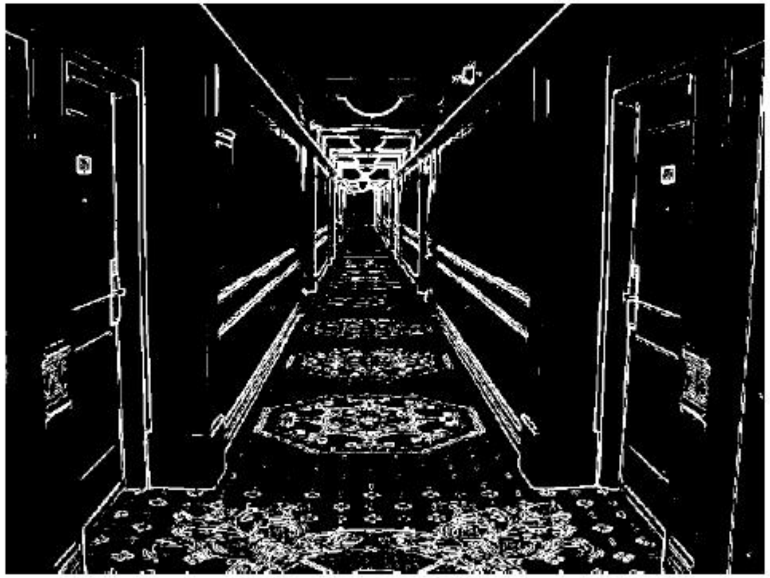
- The gradient is a measure of how the function f(x,y) changes as a function of changes in the arguments x and y.
- The gradient vector points in the direction of maximum change.
- The length of this vector indicates the size of the gradient:

$$\nabla f = |\nabla \mathbf{f}| = \sqrt{G_x^2 + G_y^2}$$

# Input to Hough – thresholded edge image

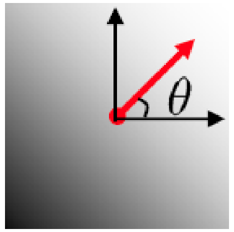
Prior to applying Hough transform:

- Compute edge magnitude from input image.
- As always with edge detection, simple lowpass filtering can be applied first.
- Threshold the gradient magnitude image.



# Extension: Incorporating image gradients

- Recall: when we detect an edge point, we also know its gradient direction
- But this means that the line is uniquely determined!



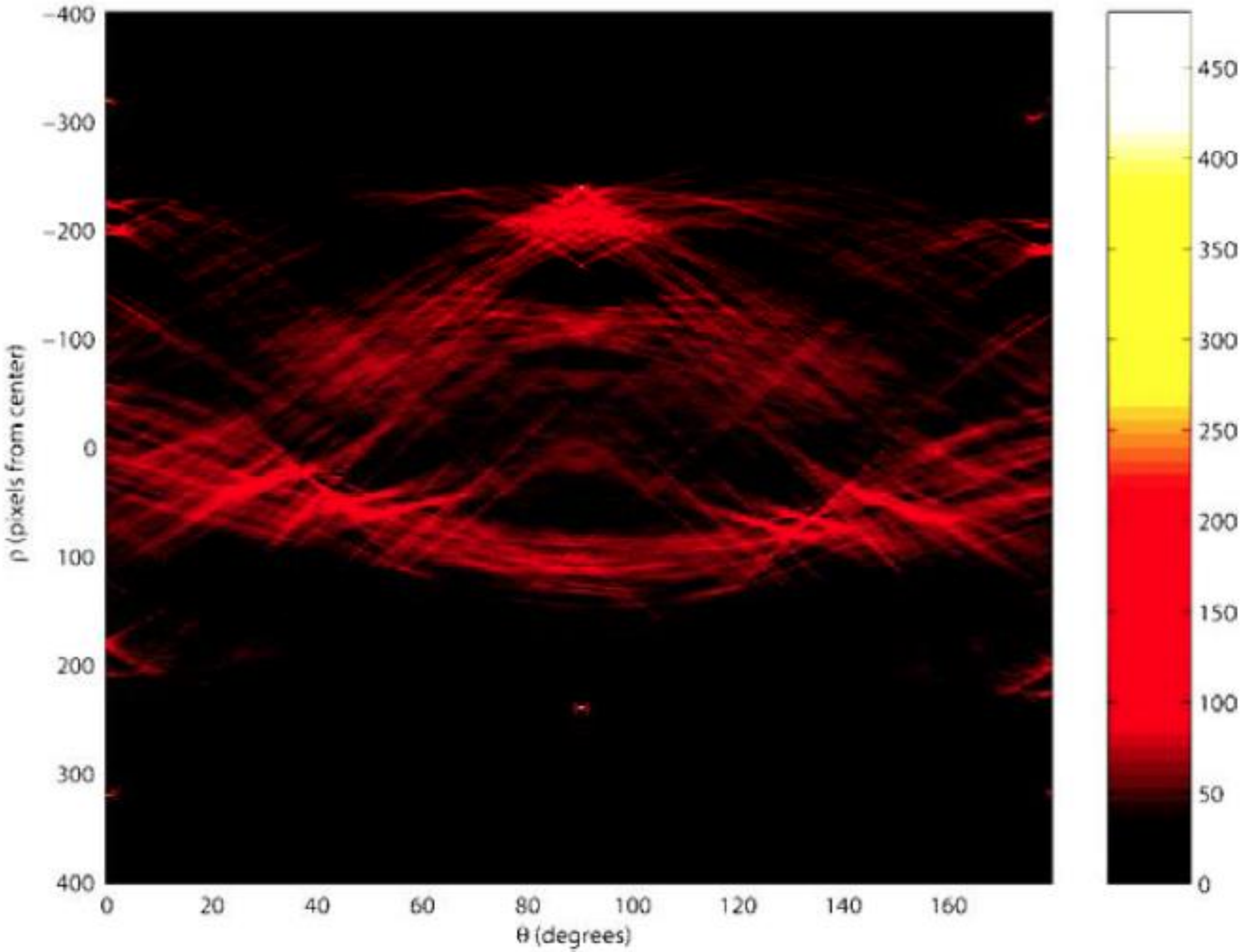

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

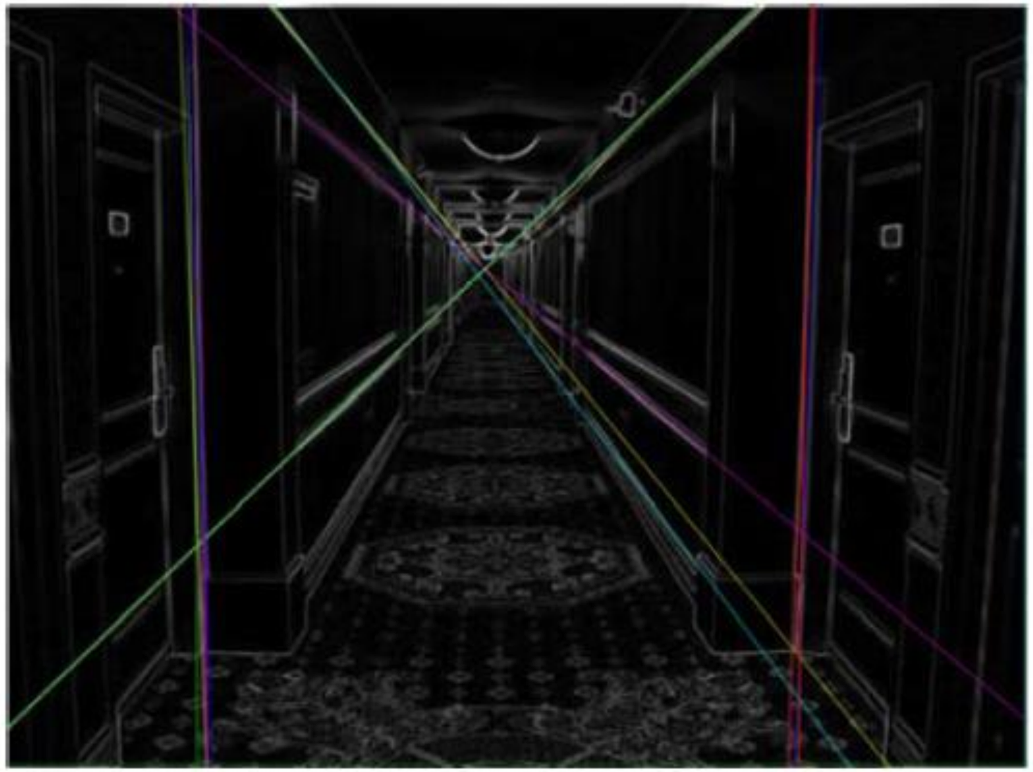
- Modified Hough transform:

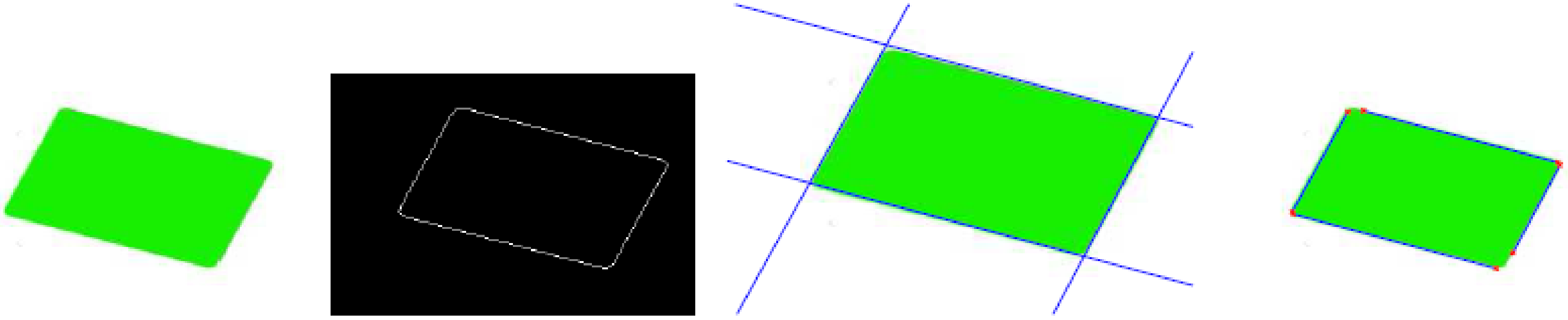
For each edge point (x,y)
    θ = gradient orientation at (x,y)
    ρ = x cos θ + y sin θ
    H(θ, ρ) = H(θ, ρ) + 1
end

- Example 3: Accumulator matrix:

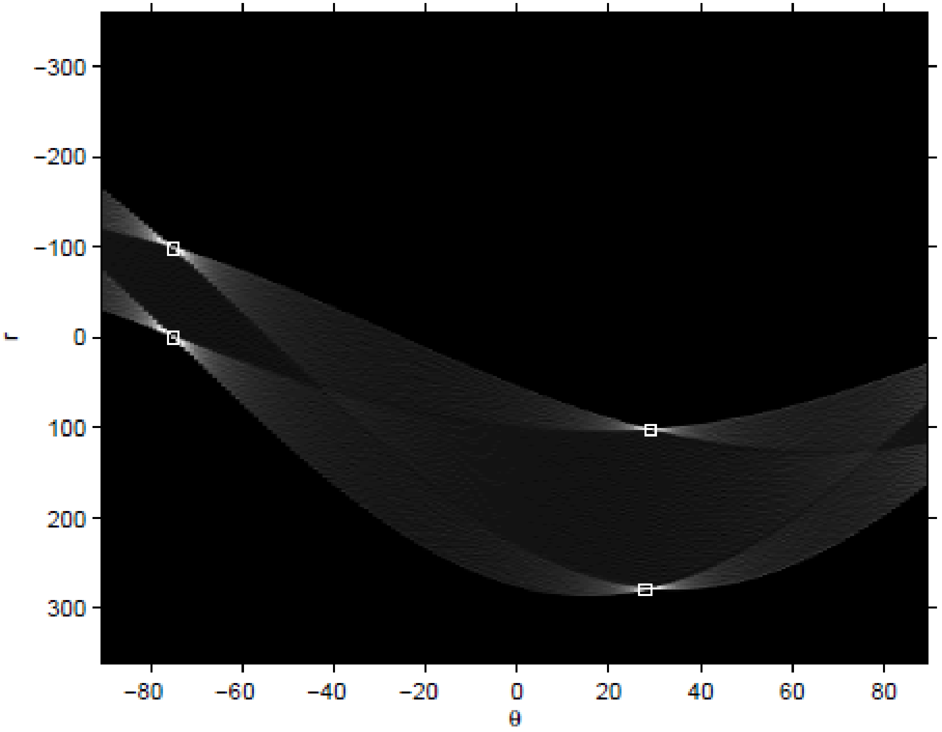


- Example 3: Original image and 20 most prominent lines:

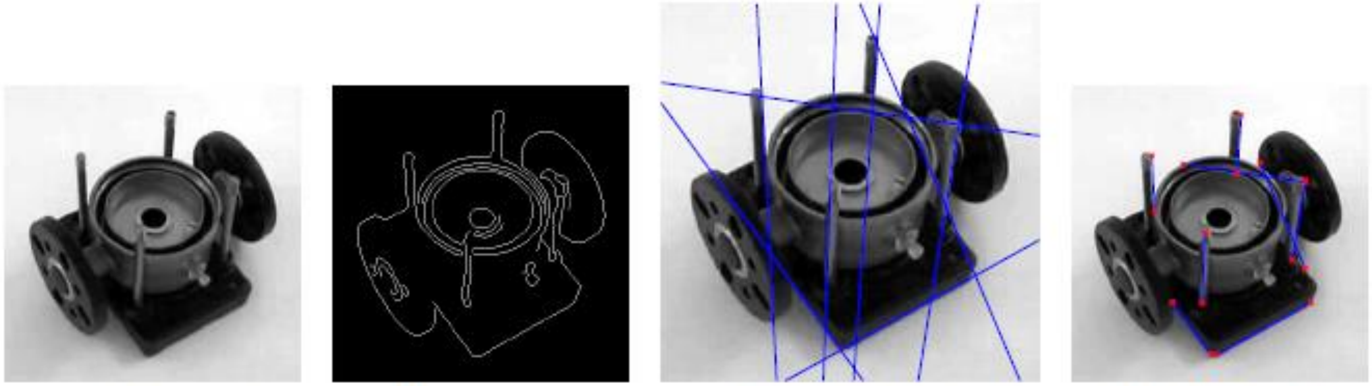




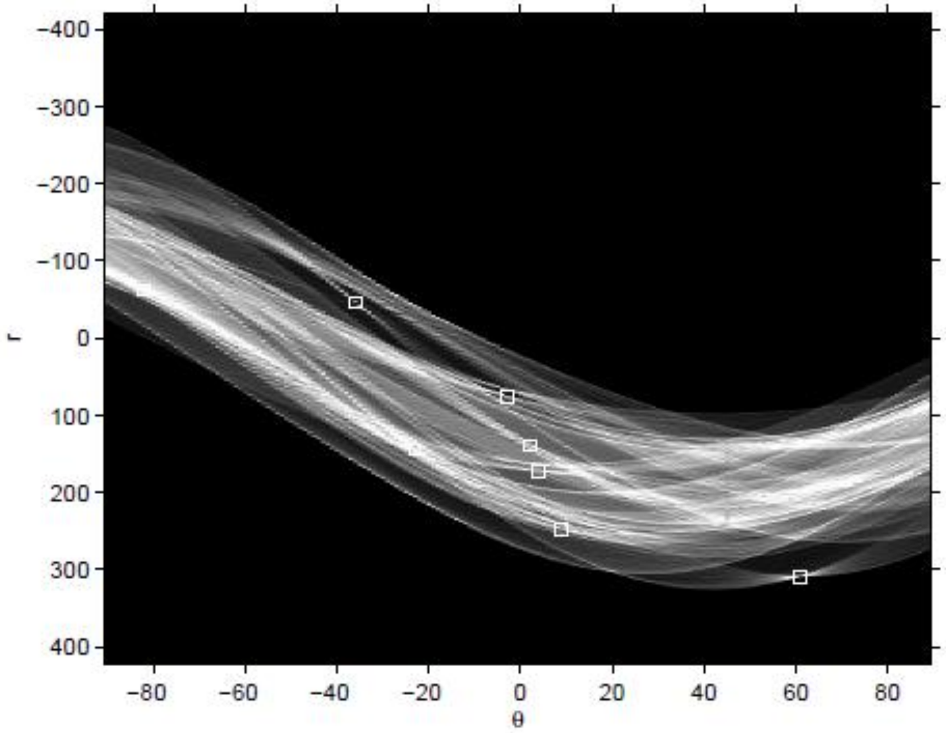
(a) Original (b) Edges (c) Infinite Lines (d) Finite Lines




(e) Hough Transform



(a) Original (b) Edges (c) Infinite Lines (d) Finite Lines




(e) Hough Transform

# Hough transform for circles

- A circle in the xy-plane is given by

$$(x-a)^2 + (y-b)^2 = c^2$$

- So we have a 3D parameter space.
- Simple procedure:

```
set all A[a,b,c]=0
for every (x,y) where g(x,y)>T
    for all a and b
        c=sqrt((x-a)^2+(y-b)^2);
        A[a,b,c] = A[a,b,c]+1;
```