EE 604
Digital Image Processing

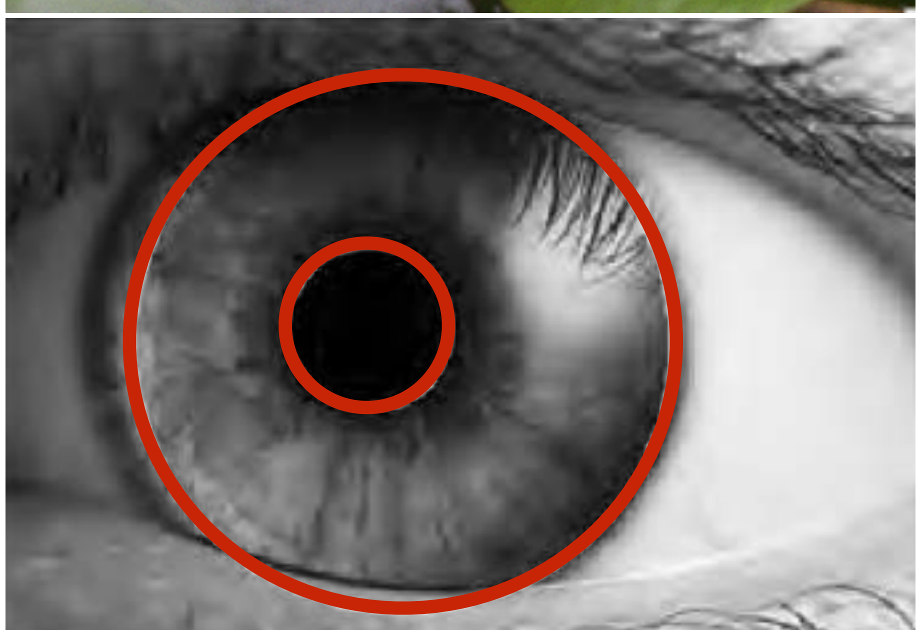
IIT KANPUR
Indian Institute of Technology, Kanpur

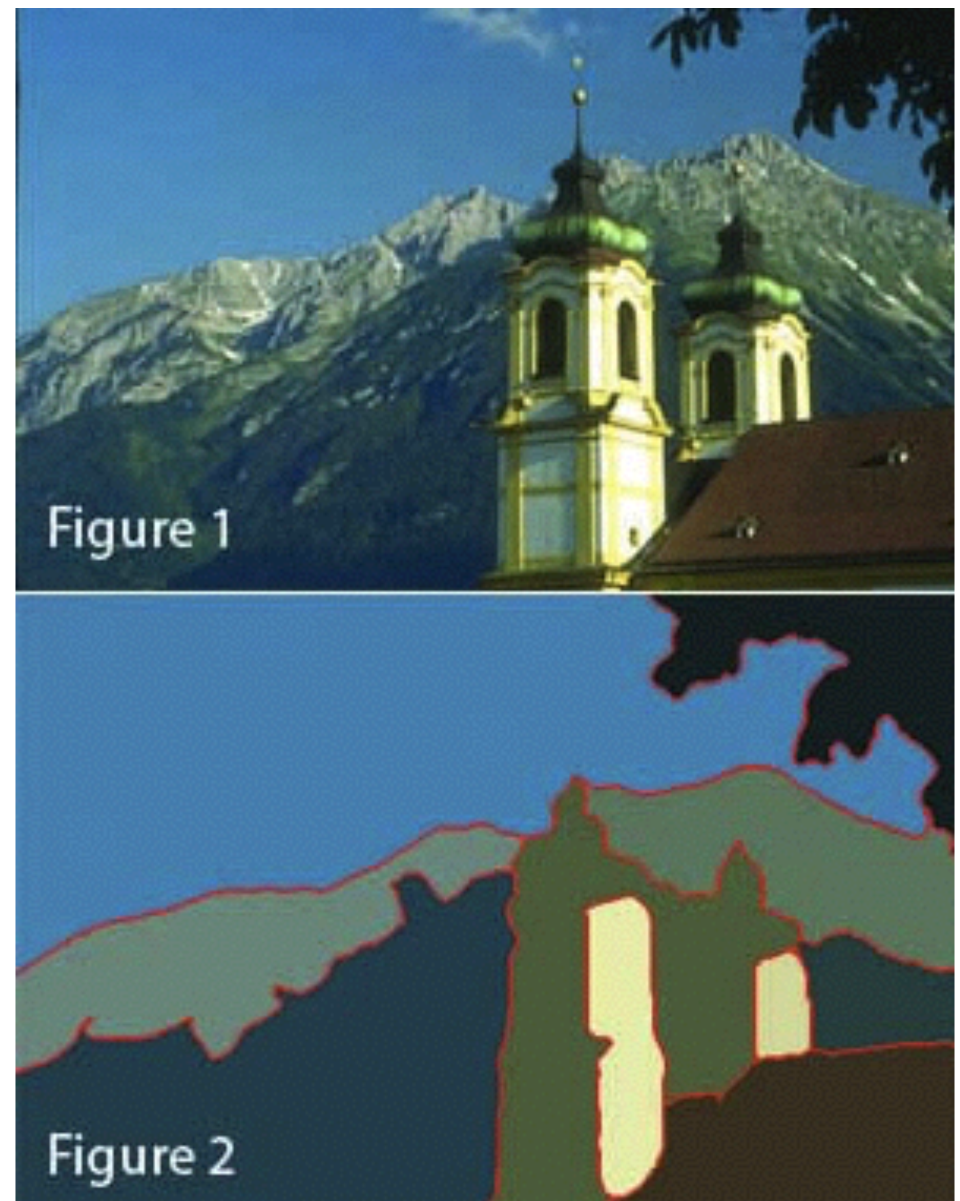
Tanaya Guha
Aug - Nov 2017

Image segmentation

- Segmentation is the process of
    - separating objects from background
    - partitioning an image into coherent regions
- In general, it is a difficult task
- Often a first step to image analysis in applications such as, object recognition.

Segmentation examples





source: http://csl.illinois.edu/

Types of Segmentation

- Partition based on predefined criteria
  - Partition based on discontinuity (isolated points, lines, edges)
  - Partition based on similarity (in color, structure, shape)
- Partition with or without manual intervention

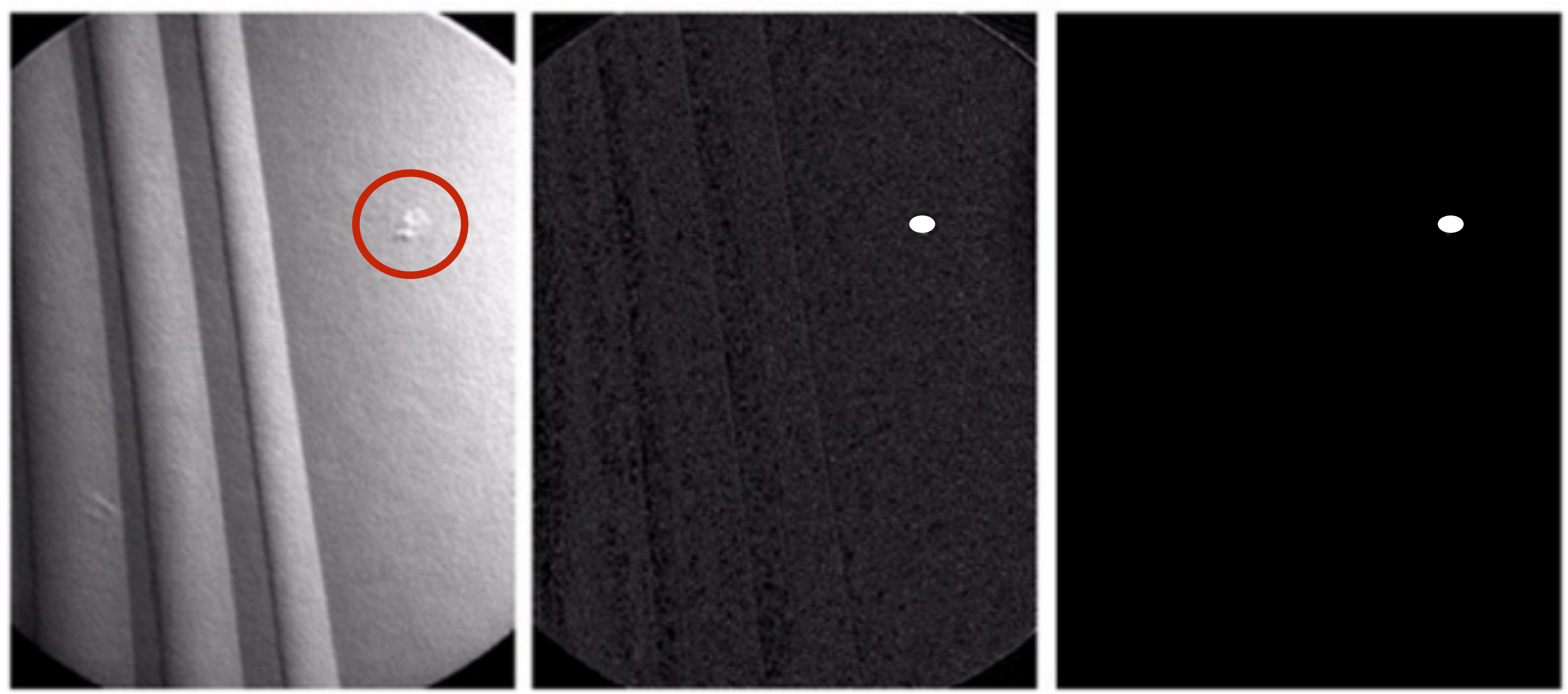
Machine Learning Perspective

- Can be seen as a supervised or unsupervised learning problem
  - Foreground-background segmentation can be seen as a binary classification of each pixel (supervised, weakly-supervised)
- Can be seen as a data clustering problem (unsupervised)
  - image segments = clusters in a suitable feature space

Basics

- Detect gray-level discontinuities
    - isolated points, lines, edges
- **Isolated point detection:**

−1	−1	−1
−1	8	−1
−1	−1	−1



Basics

- Detect gray-level discontinuities
    - isolated points, lines, edges
- **Line detection:**

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+45°

−1	2	−1
−1	2	−1
−1	2	−1

Vertical

2	−1	−1
−1	2	−1
−1	−1	2

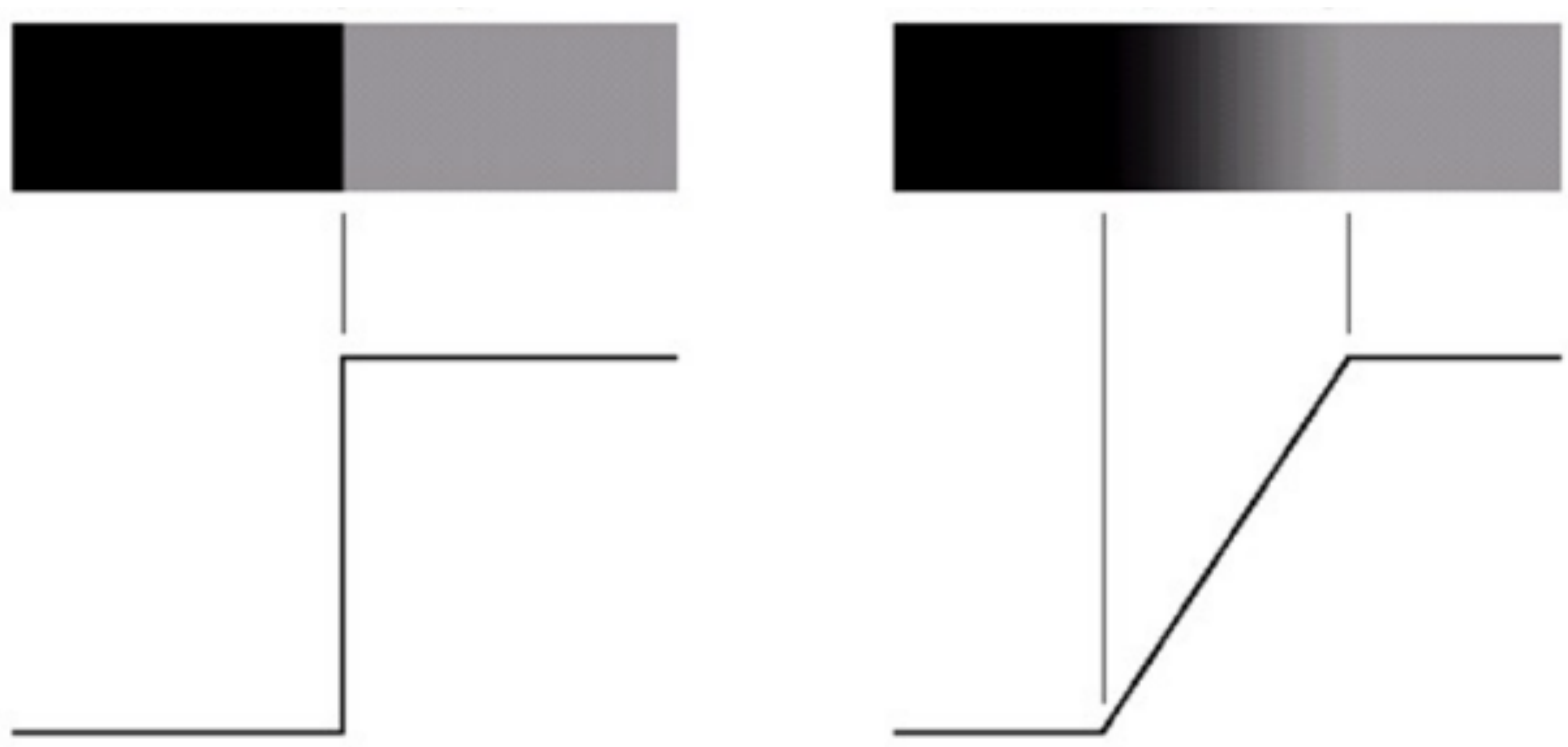
−45°

Basics

- Detect gray-level discontinuities
  - isolated points, lines, edges
- **Line detection:**

Basics

- Detect gray-level discontinuities
    - isolated points, lines, edges
- **Edge detection:**

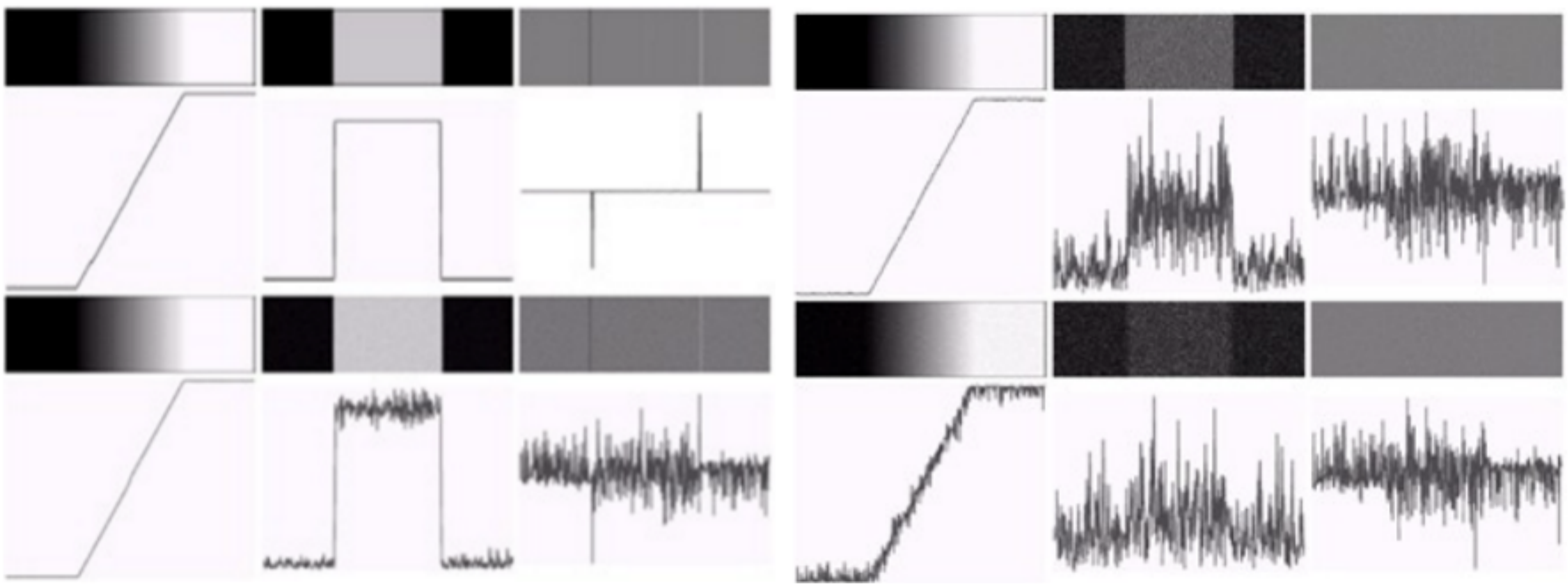


gray-level profile

gray-level profile

Basics

- Detect gray-level discontinuities
    - isolated points, lines, edges
- **Edge detection:**



Hough transform

- Detects geometric shapes, such as lines and circles, in images, when the parametric equation is known.
- Introduced in 1962 by [Hough 1962], and was first used to find lines in images in [Duda 1972].
- Robust detection under noise and partial occlusion
- Requires edge detection as a preprocessing step

Hough transform

Can we achieve these?

- Find the n most prominent lines
- Find lines in a particular orientation

Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform

Before applying Hough transform

- Apply edge detector (any derivative operator)
- Compute a magnitude edge map
- Binarize the edge map by thresholding

Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform

Edge map as input image to Hough transform

Before applying Hough transform

- Apply edge detector (any derivative operator)
- Compute a magnitude edge map
- Binarize the edge map by thresholding

Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform

- Assume that we have performed edge detection and binarization.
- We have n pixels (white) in the image that may partially describe edges and boundaries.
- We wish to a find subset of pixels that form straight lines.
- Consider a point (x_i, y_i) in the image which lies in a line $y_i = ax_i + b$
  - Infinitely many lines pass through (x_i, y_i).
  - They satisfy the equation for different values of $(a\ b)$.

Hough transform

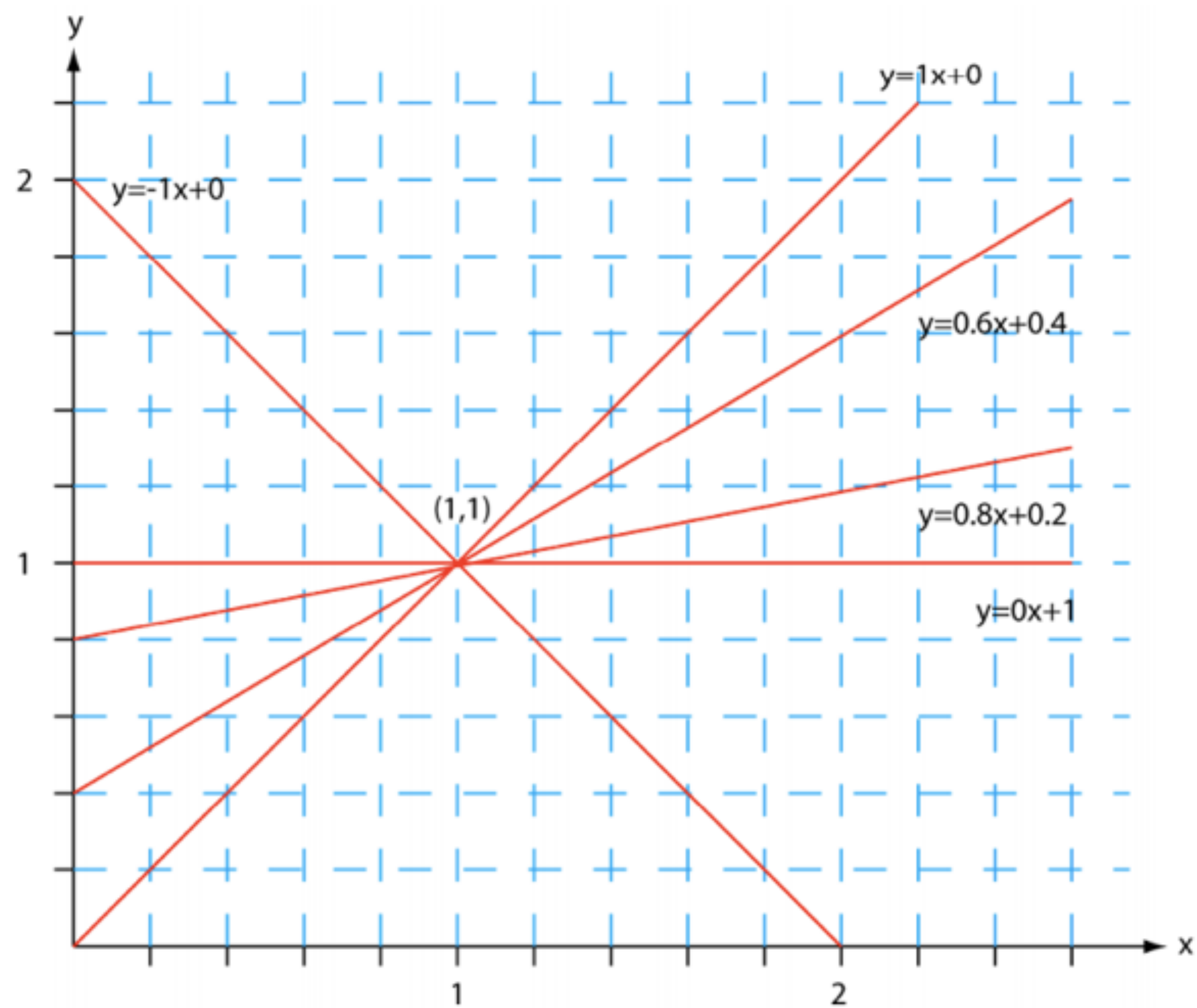


Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform

- The equation can obviously be rewritten as $b = -ax_i + y_i$
- Now consider x and y as parameters and a and b as variables.
- This is a line in ab-space parameterized by x and y.
- A single point in xy-space corresponds to a line in ab-space.
- Another point in xy-space will give rise to another line in ab-space.

Hough transform

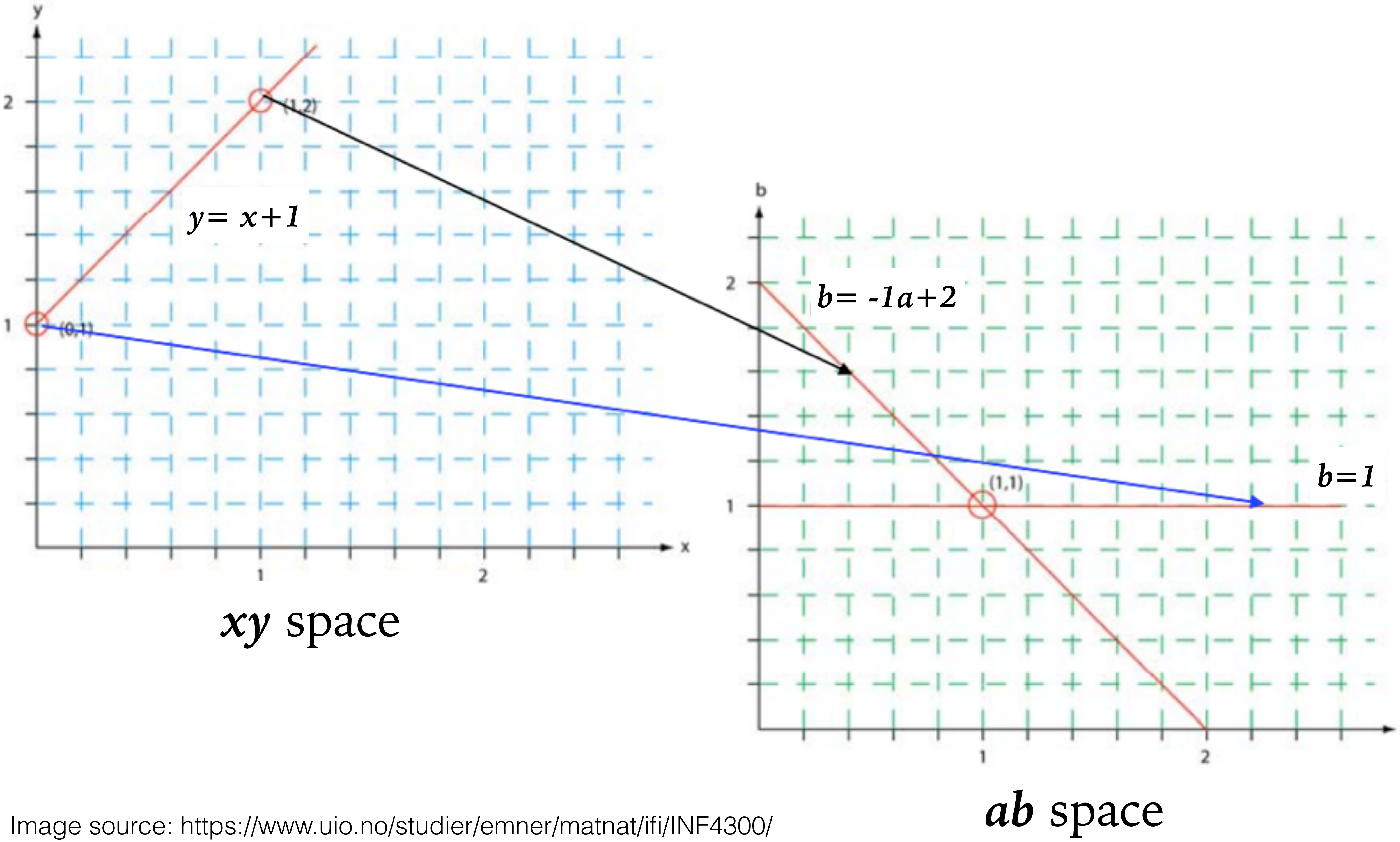
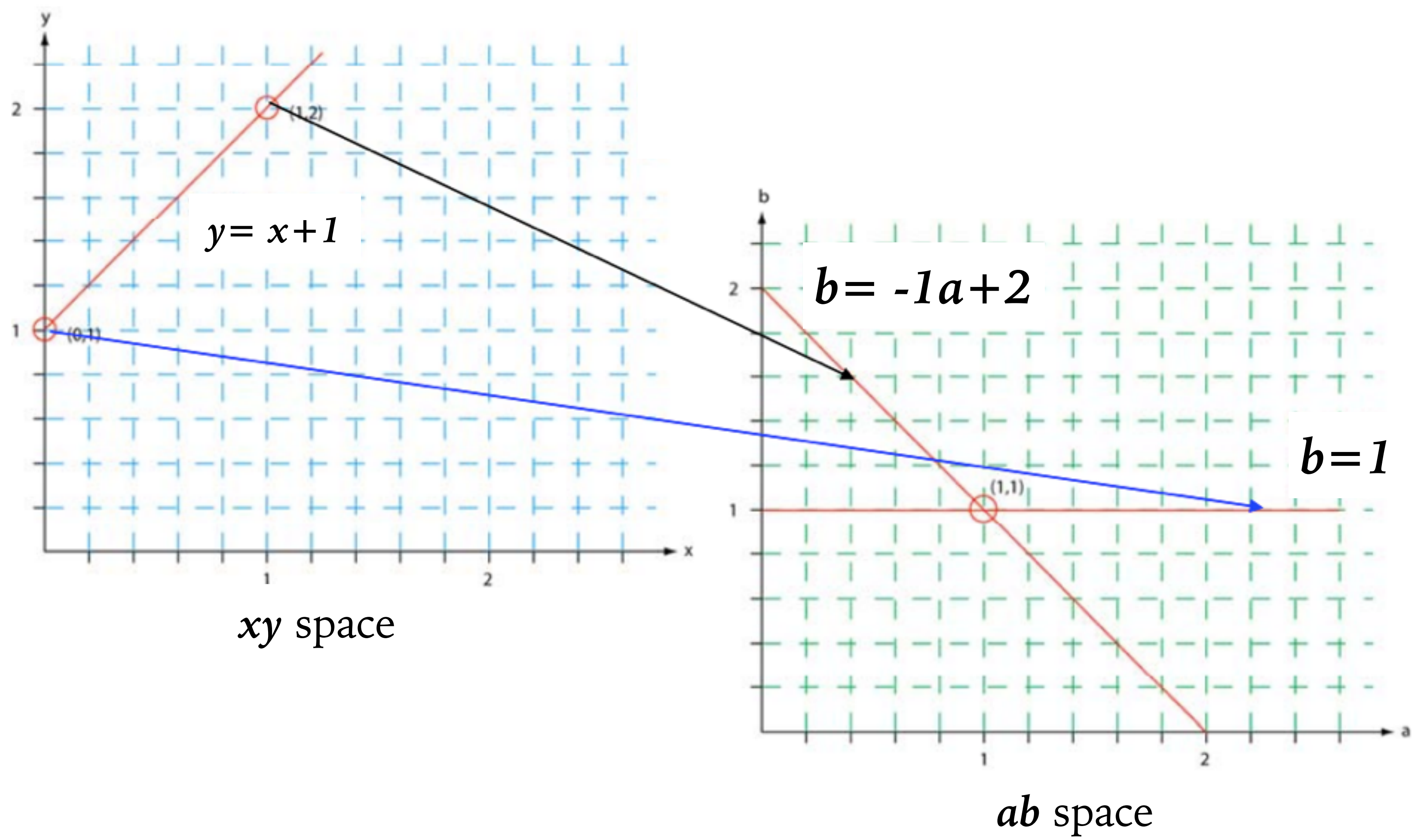


Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform



- All points on $y=x+1$ will reflect as intersecting lines at $(1,1)$ in ab-space
- What if we count the number of intersections at each (a,b) location?

Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform

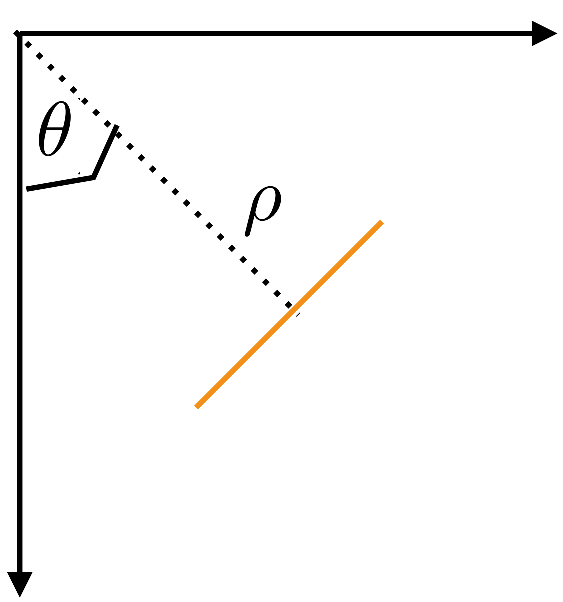
Key idea: Move from pixel space to parameter space

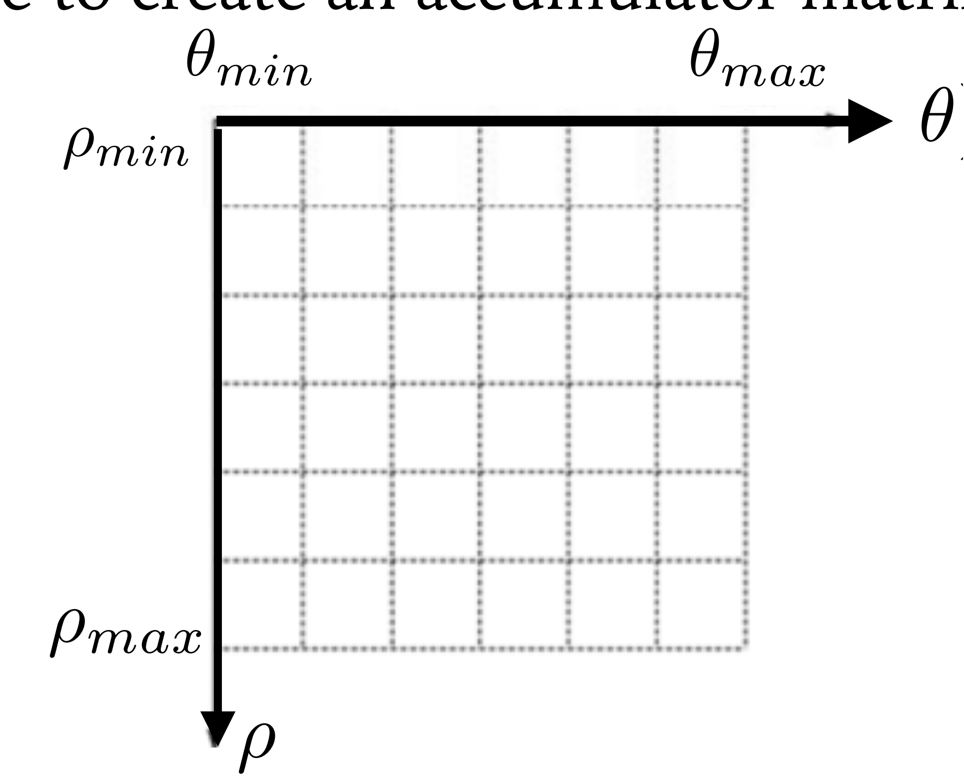
- In practice, we use the polar representation of lines

$$x_i \cos\theta + y_i \sin\theta = \rho$$

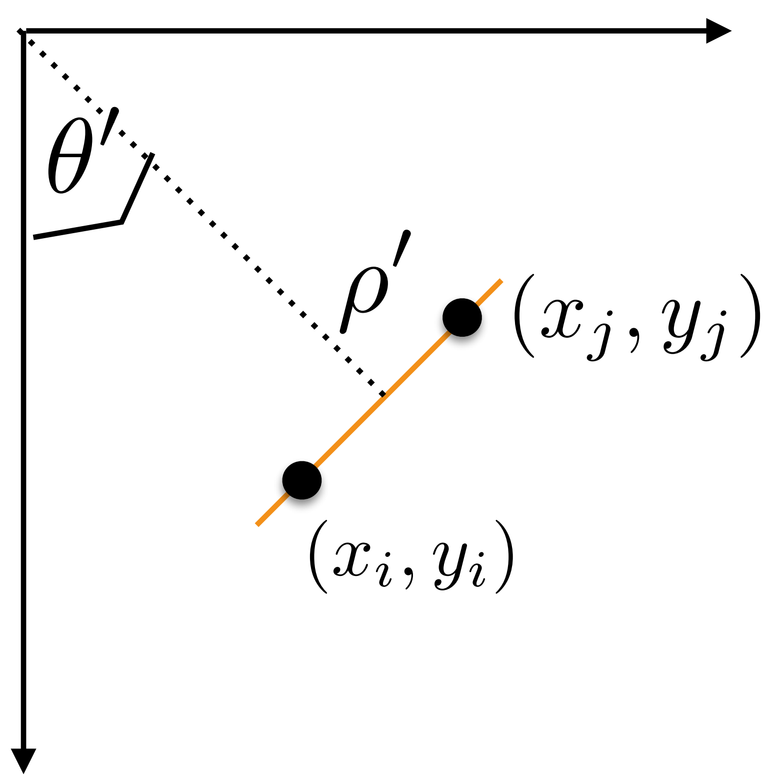
- Instead of ***ab***-space, we thus have a (ρ, θ) space (Hough space)

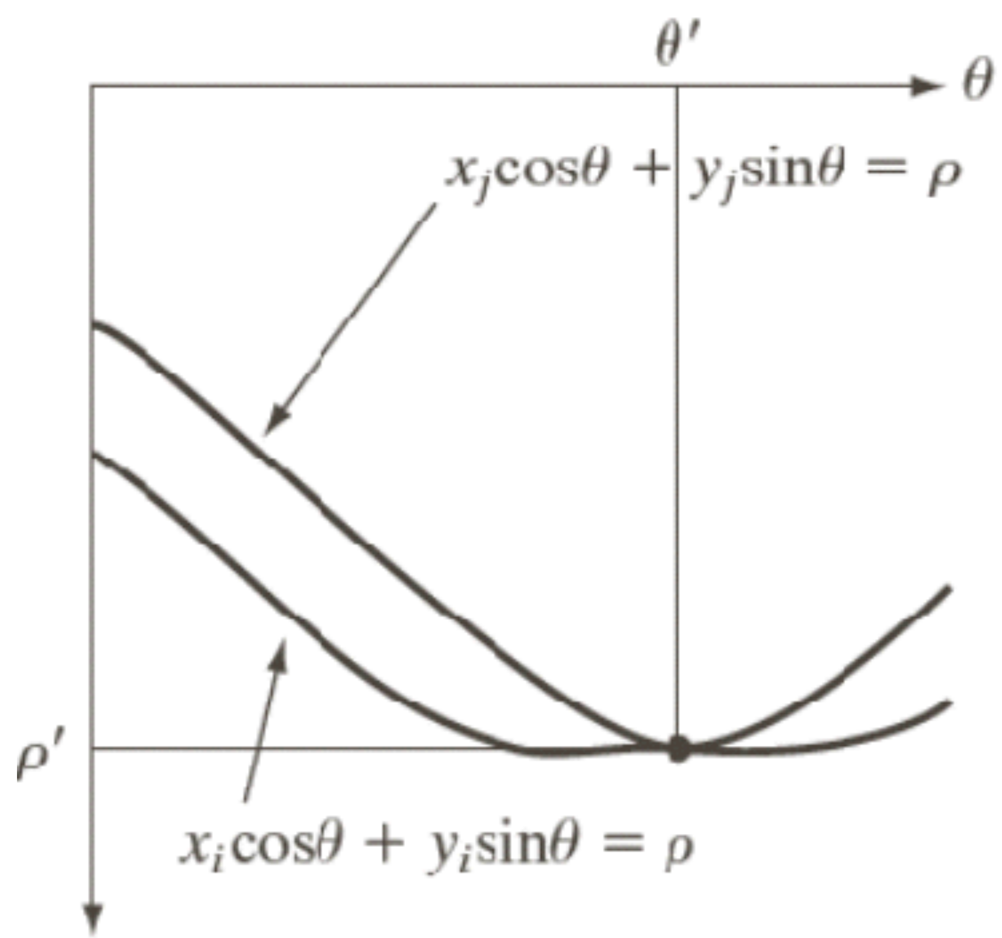
- We will discretize the (ρ, θ) space to create an accumulator matrix ***A***





Hough transform

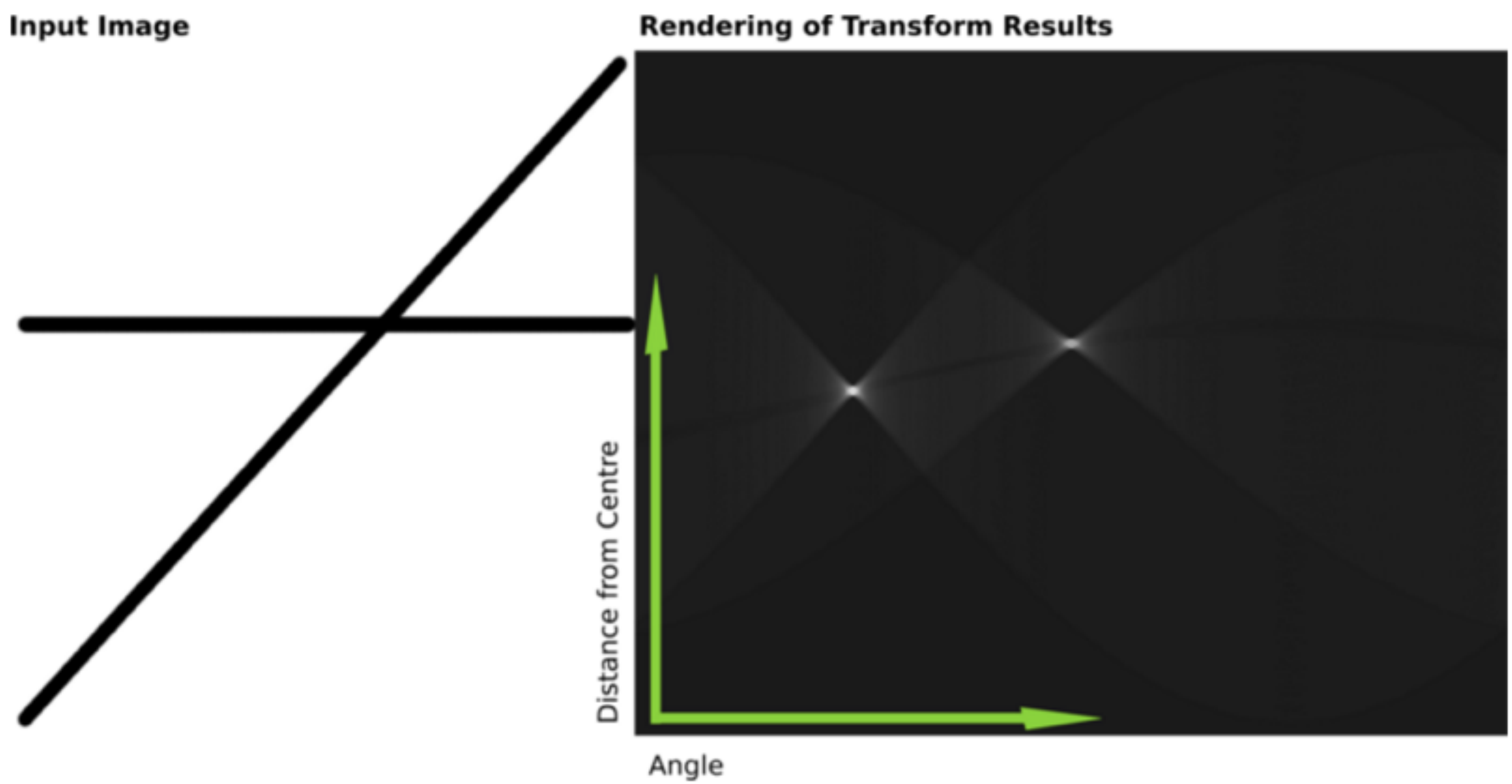




Hough transform

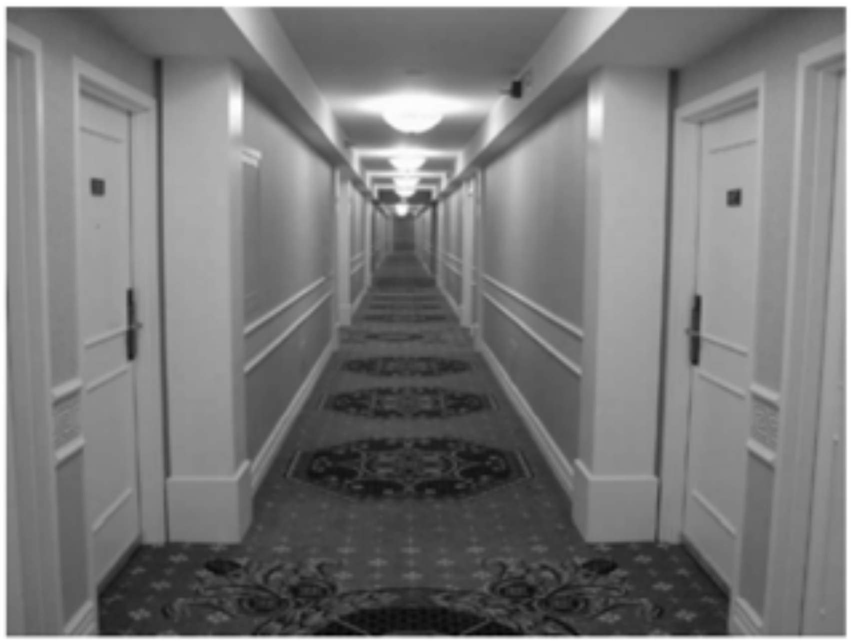
- Discretize (ρ, θ) space to create accumulator matrix ***A,*** where ***A(i,j)*** is a cell corresponding to ***i***-th ρ value and ***j***-th θ value
  - range of ρ ?
  - range of θ ?
- Initialize: ***A(i,j)*** $= 0$ for all ***i*** and ***j***
- For each pixel ***(x,y):***
  - vary θ through all possible values and and compute ρ
  - For each (ρ_j, θ_j) pair, increment cell ***A(i,j)***
- Find the largest values in Hough space

Hough transform



source: wikipedia

Hough transform



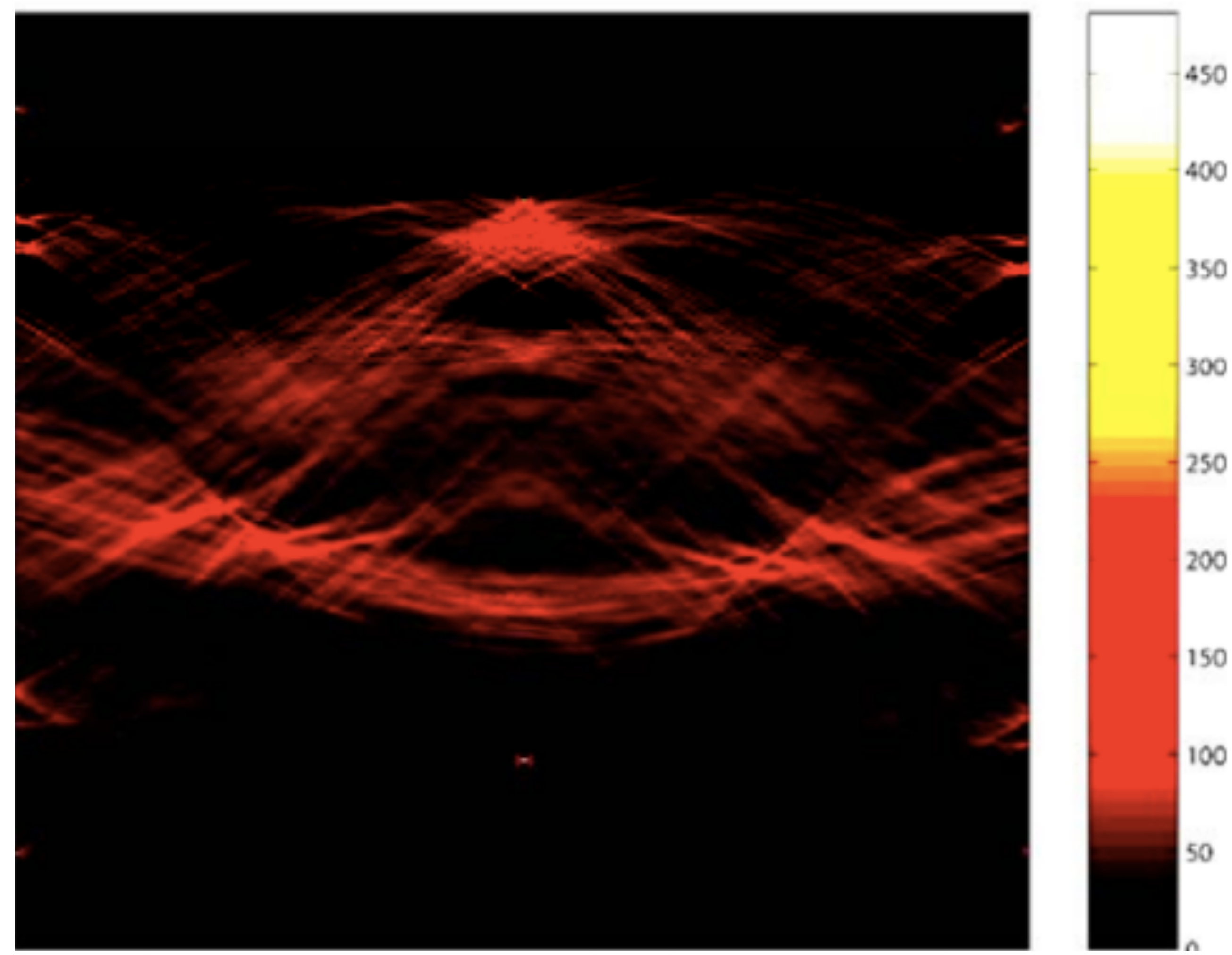


Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough transform

Image source: https://www.uio.no/studier/emner/matnat/ifi/INF4300/

Hough Transform

- Robust to occlusion, noise, missing data
- Simple implementation
- Can be extended to find other parametric shapes
- Complex shapes will give rise to high-dimensional Hough space - computationally expensive.
- Looks for only one type of shape at a time
- Co-linear line segments can not be separated