Fitting & Alignment

Tae Hyun Kim

Fitting and Alignment: Methods

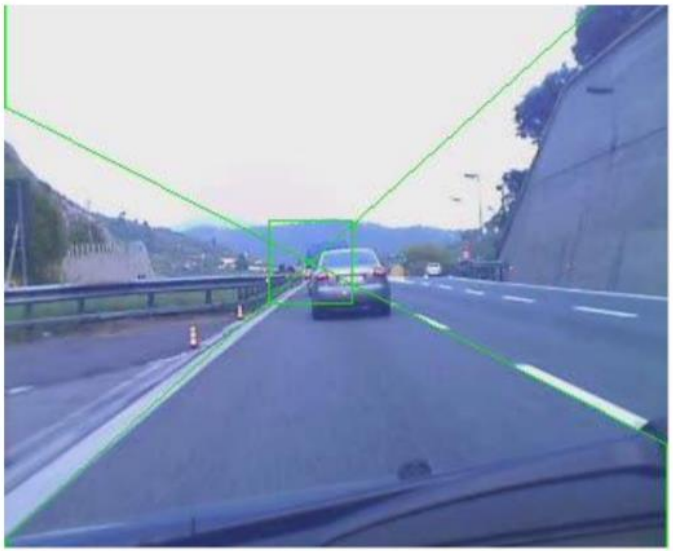
- Global optimization / Search for parameters
  - Least squares, total least squares
  - Robust least squares
  - Iterative closest point (ICP)

- Hypothesize and test
  - RANSAC
  - Generalized Hough transform

Fitting: Issues

- Noise in the measured feature locations
- Extraneous data: clutter (outliers), multiple lines
- Missing data: occlusions

- Case study: Lane detection



Hypothesize and test
Recall : RANSAC

1. Propose parameters
   - Try possible models as many as possible
   - Each point votes for consistent parameters
   - Repeatedly sample enough points to solve for parameters

2. Score the given parameters
   - Number of consistent points (inliers)

3. Choose best parameters among the sets of parameters
   - Global or local maximum of scores

4. Possibly refine parameters using inliers

Voting schemes

- Let each feature (point) vote for all the models that are compatible with it

Voting에 기반한 방법들이 가지는 최대 장점은 outlier들을 쉽게 제거할수 있다
최대 단점은 오래걸림

- Hopefully, the noisy features will not vote consistently for any single model

- Missing data doesn't matter if there are enough features to agree on a good model

Hough Transform: Outline

P.V.C. Hough, *Machine Analysis of Bubble Chamber Pictures,* Proc. Int. Conf. High Energy Accelerators and Instrumentation, 1959

- An early type of voting scheme
- General outline
  - Discretize parameter space into bins
  - For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
  - Find bins that have the most votes

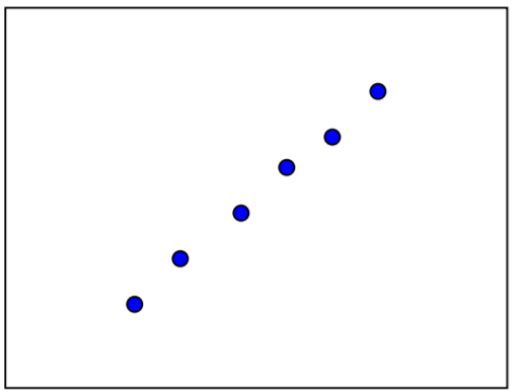
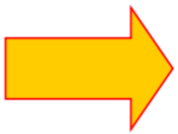
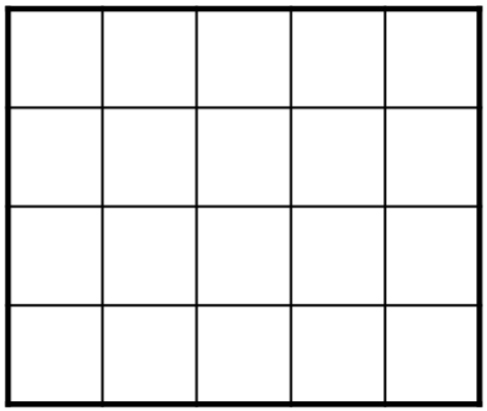


Image space

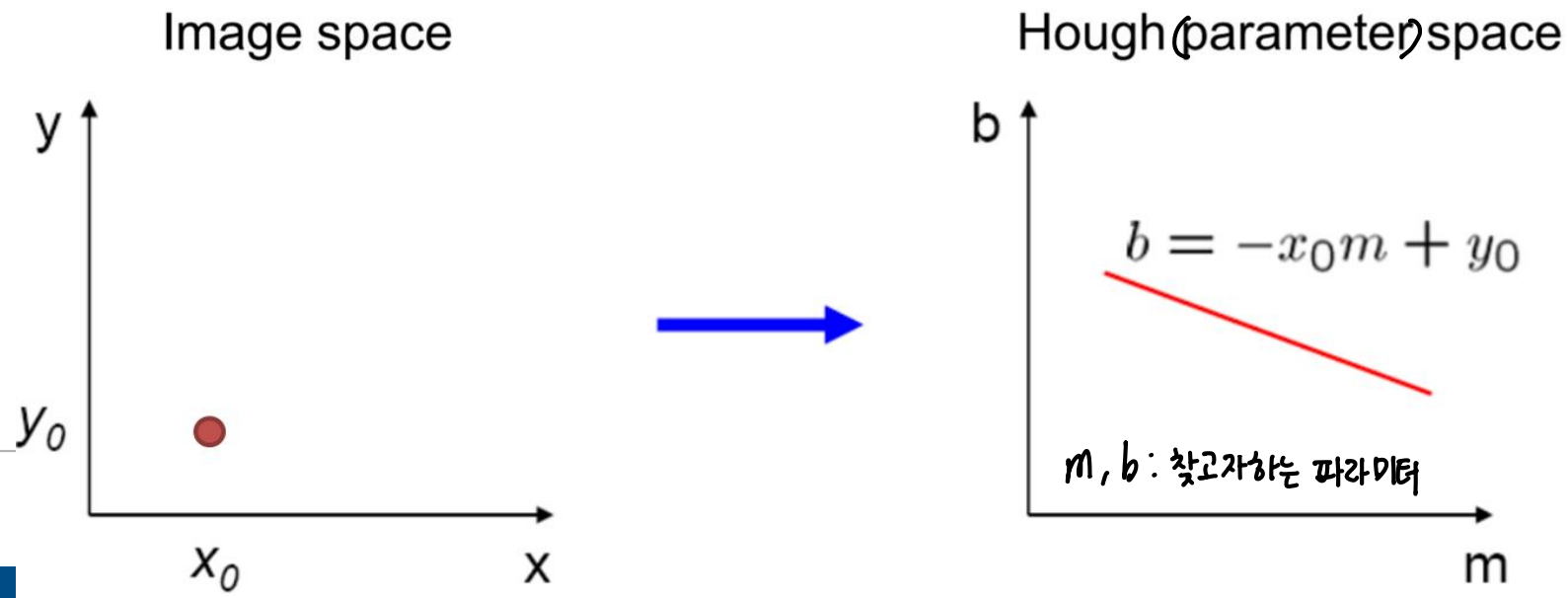




Hough parameter space

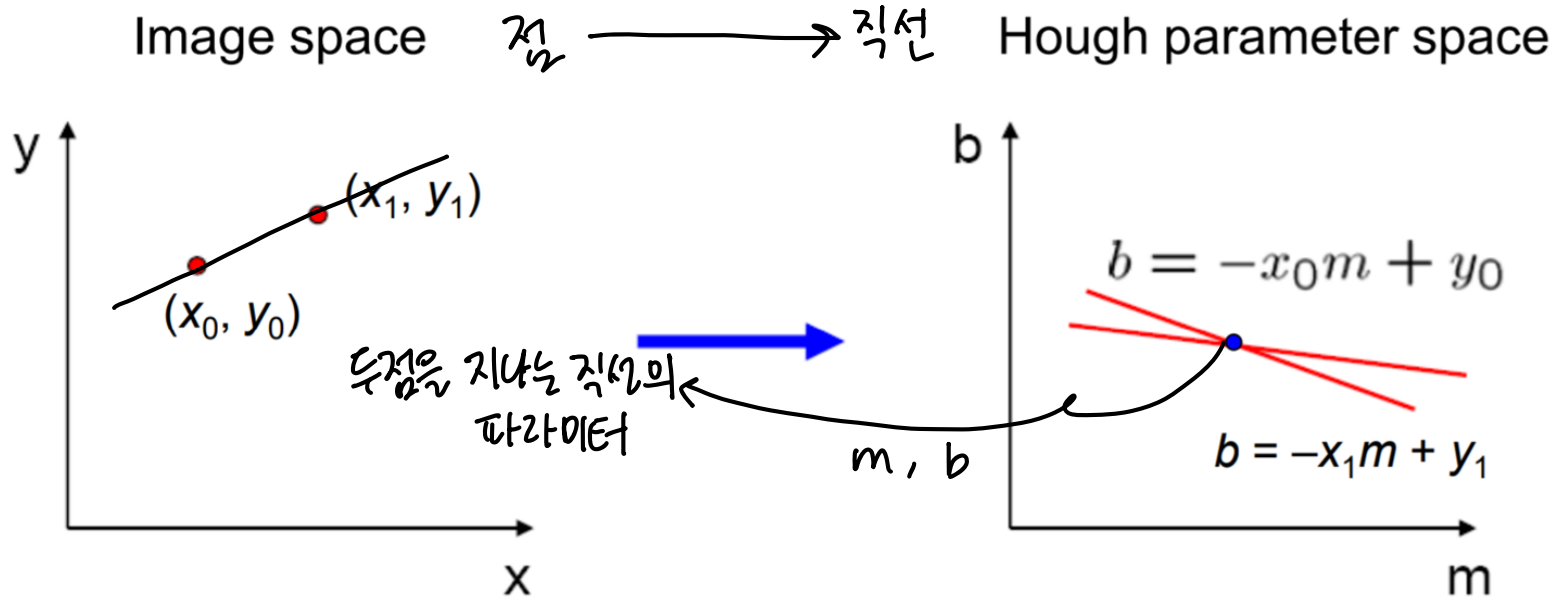
Parameter space representation

- What does a point in the image space map to in the Hough space?
  - Mapped to a single line in the Hough space
    - A point $(x_0, y_0) \rightarrow$ A line $b = -x_0 m + y_0$
    - Hough space: space of parameter we want to estimate



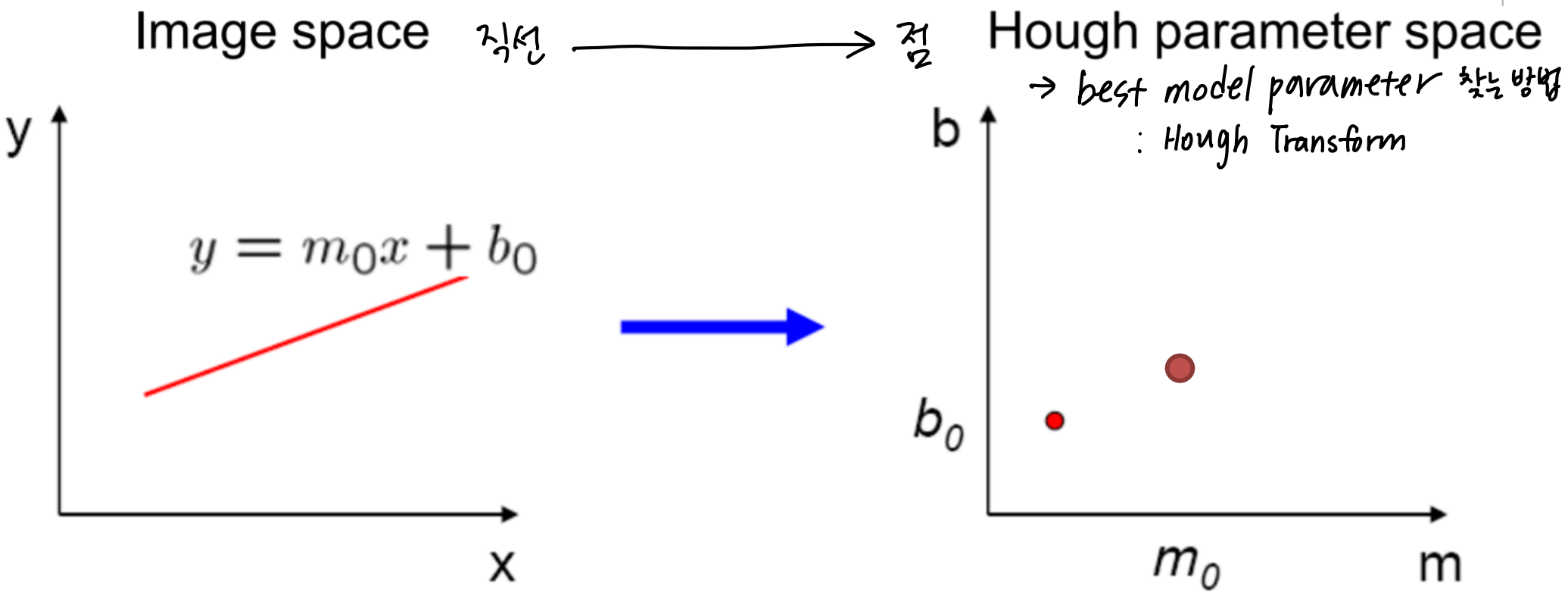
Parameter space representation

- Mapping multiple points in the image space
  - Two points (x_0, y_0), (x_1, y_1) → two lines $b = -x_0 m + y_0$, $b = -x_1 m + y_1$
- Intersection of two lines in the Hough space
  - Compatible parameters for the two points



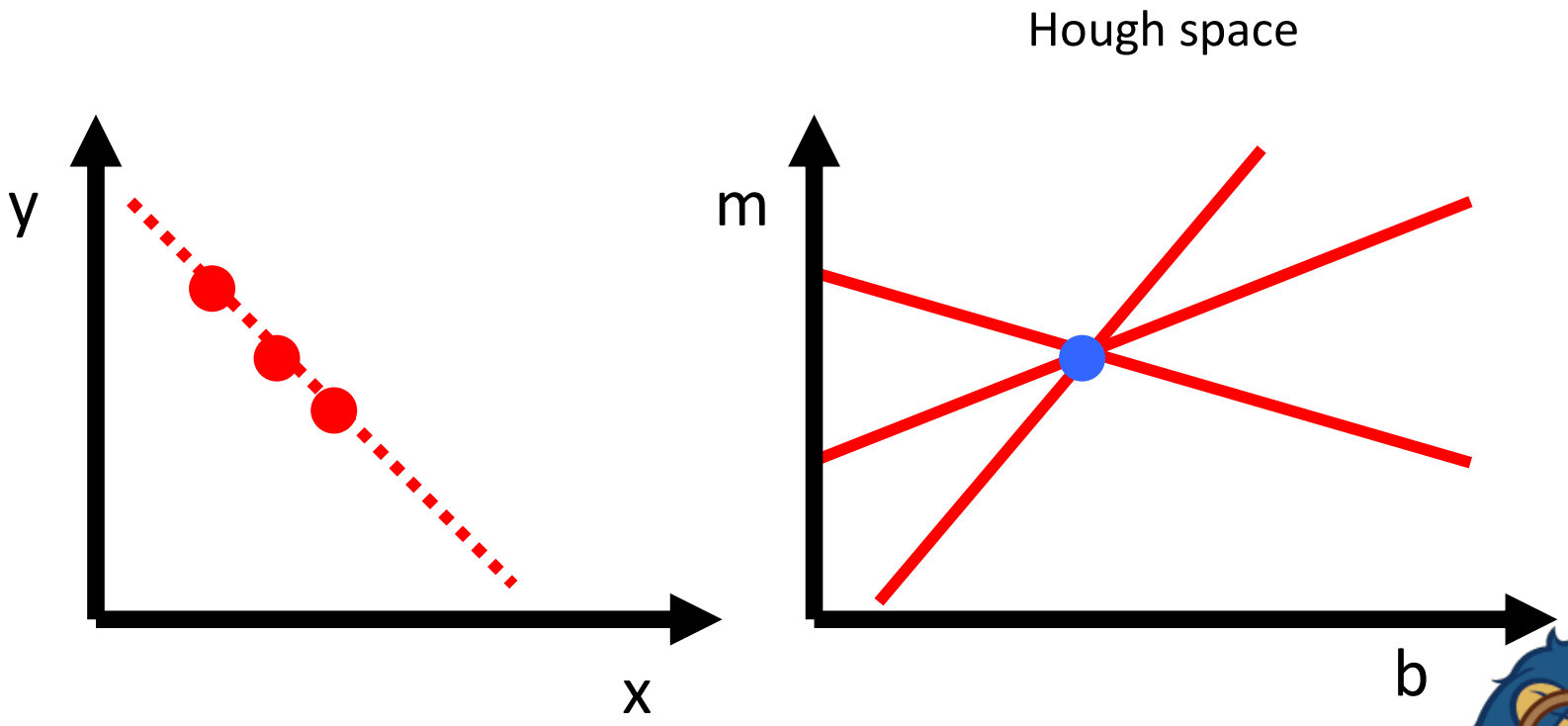
Parameter space representation

- A line (infinitely many points) in the image
  - A point in the Hough space



Hough transform

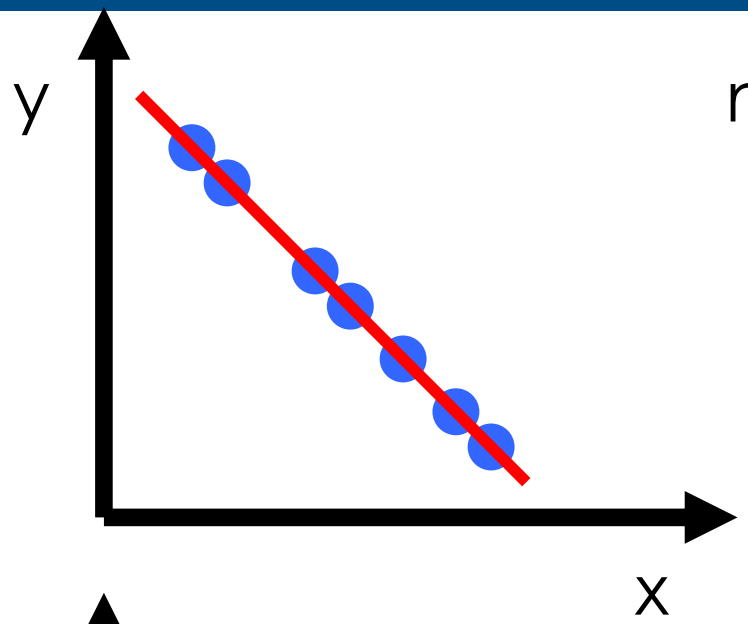
- Given a set of points, find the parameters of a line that explains the data points best

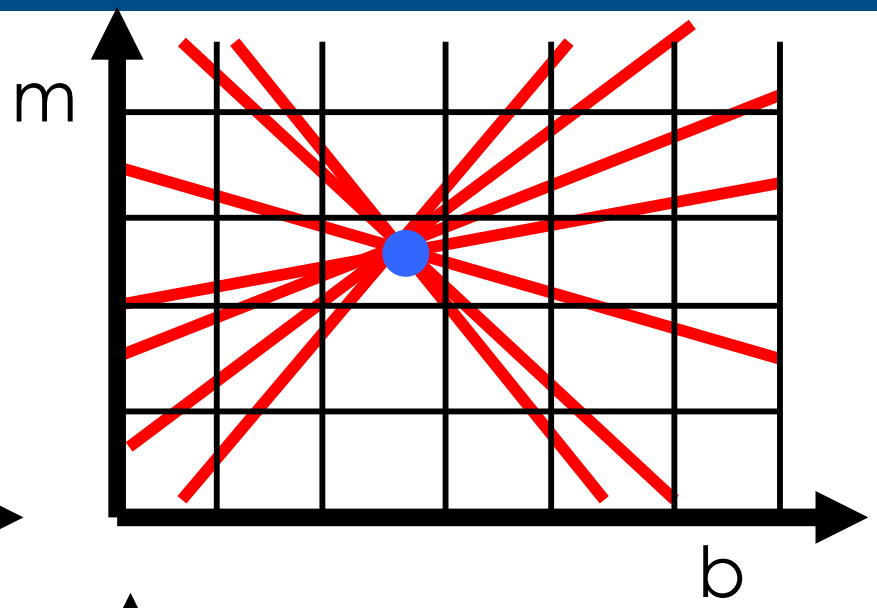


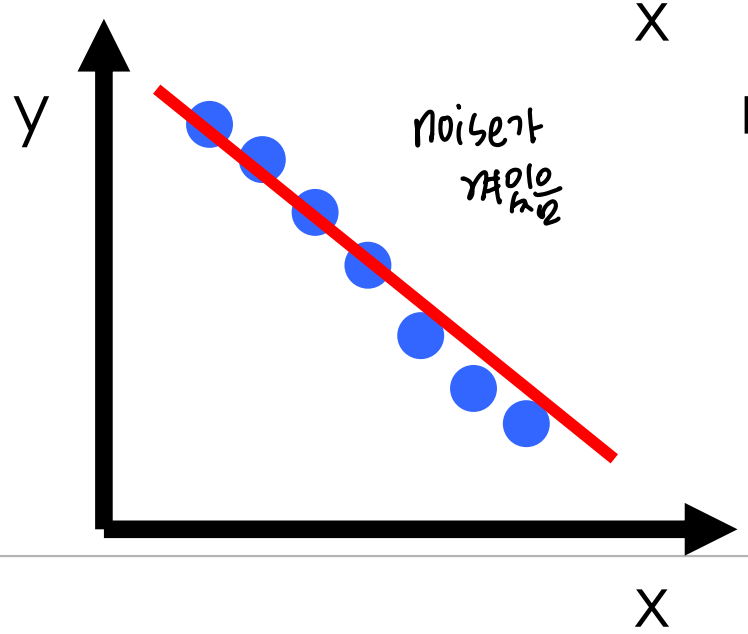
$$y = mx + b$$

Slide from S. Savarese

Hough transform



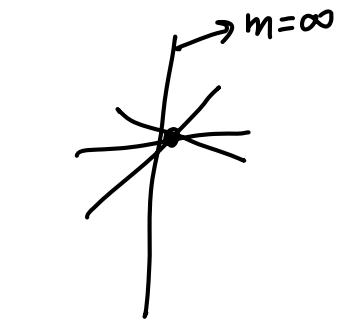




Slide from S. Savarese

Polar representation for lines

- Problems in the Hough space (m,b):
  - Unbounded parameter domain
    - **m**=[- inf, …, 0, …, inf]
    - Vertical lines require infinite **m**
  - Require infinitely large number of bins for voting

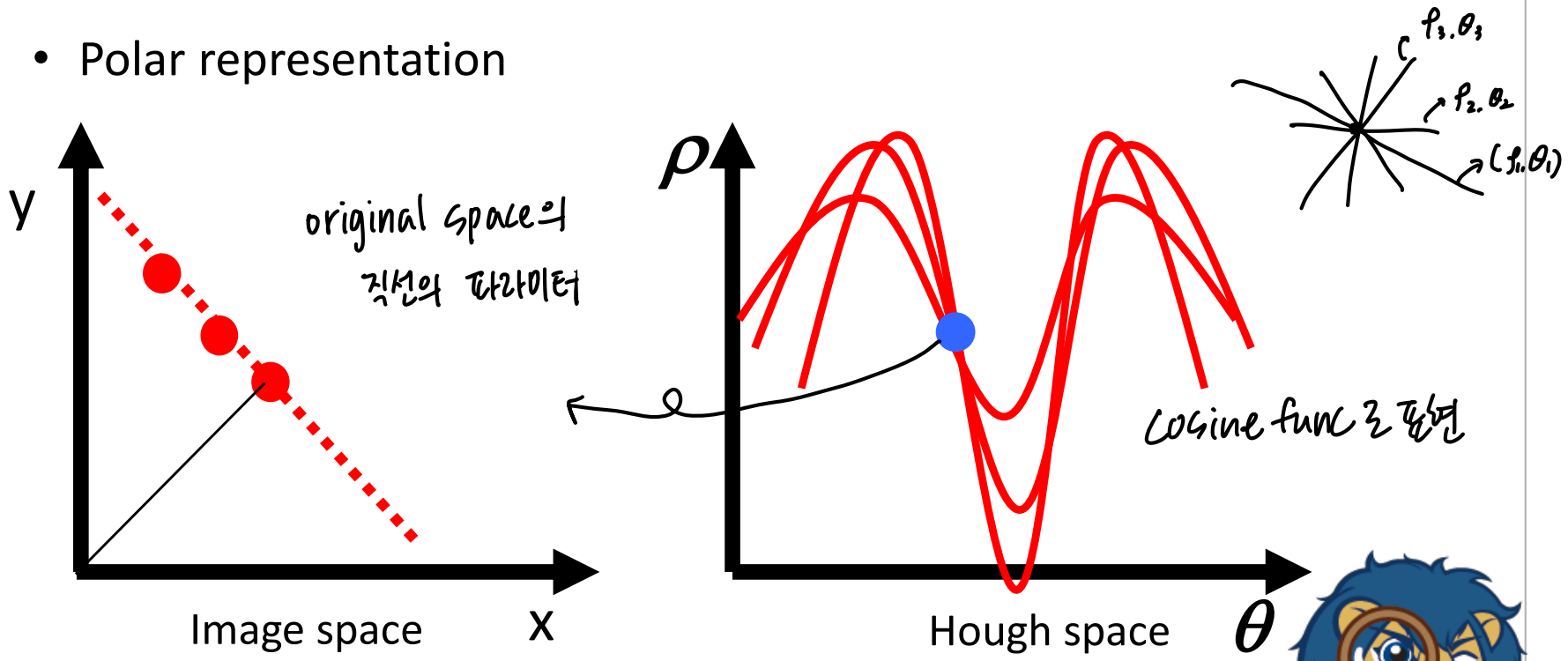


parameter space에 무한히 많은 bin이 필요

parameter space 를 양산: hough transform

Polar representation for lines

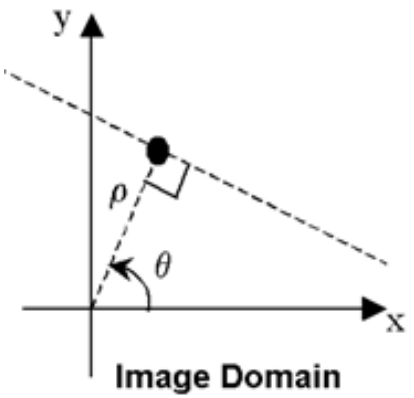
- Alternative
  - Map a point in the image to a single sinusoidal line in the Hough space
    - Polar representation



Polar representation

- Any line through (x,y) in image space can be parametrized in polar representation
  - $\rho = x \cdot \cos\theta + y \cdot \sin\theta$

    ρ, θ → 직선 표현 가능 → 극좌표계 polar representation



- Image point (x,y) → parameters (ρ, θ)
- What are ρ & θ ?
  - $\rho = A \cdot \cos(\theta - \delta)\,, A = \sqrt{x^2 + y^2}, \delta = \tan^{-1}(y/x)$
  - Result
    - Given (x,y) in image spcae→ **A cosine function** in the parameter (Hough) space

Algorithm outline

- Initialize all bins (H(θ, ρ) = 0)
- For each point (x,y) in the image

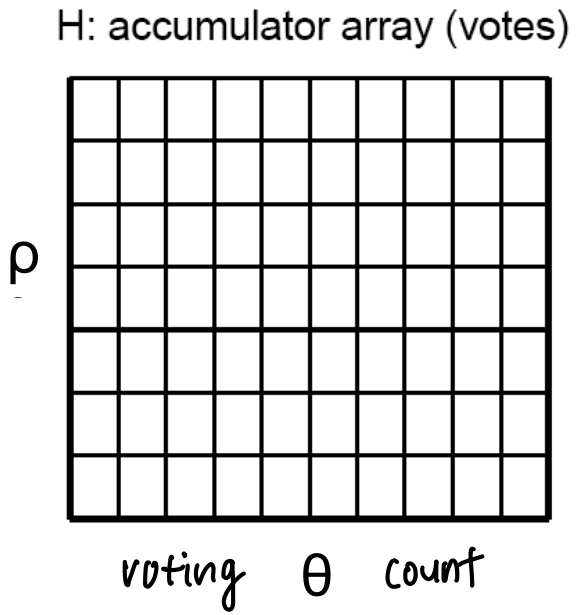
  For θ = 0 to 180

  ρ = x cos θ + y sin θ

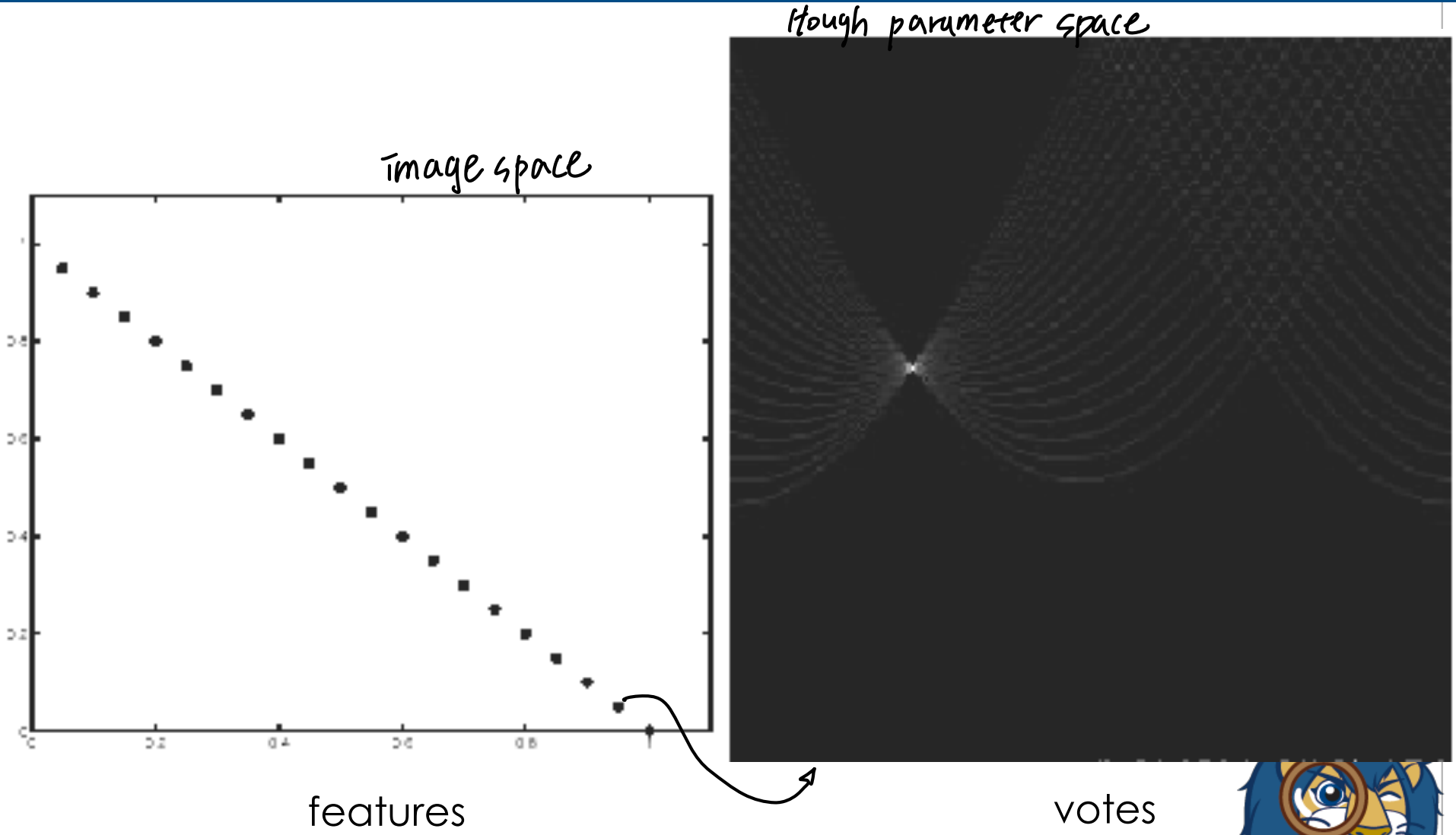
  H(θ, ρ) = H(θ, ρ) + 1

  end

  end
- Find the θ^*, ρ^* where H(θ^*, ρ^*) is the maximum
  - The detected line in the image is given by,
    - ρ^* = x cos θ^* + y sin θ^*

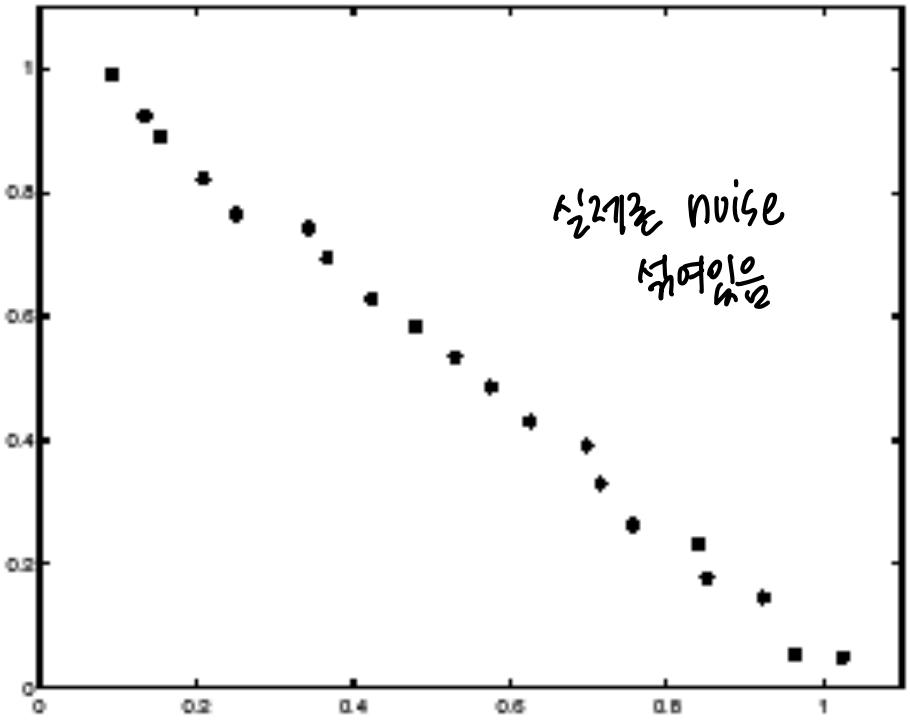


Basic illustration

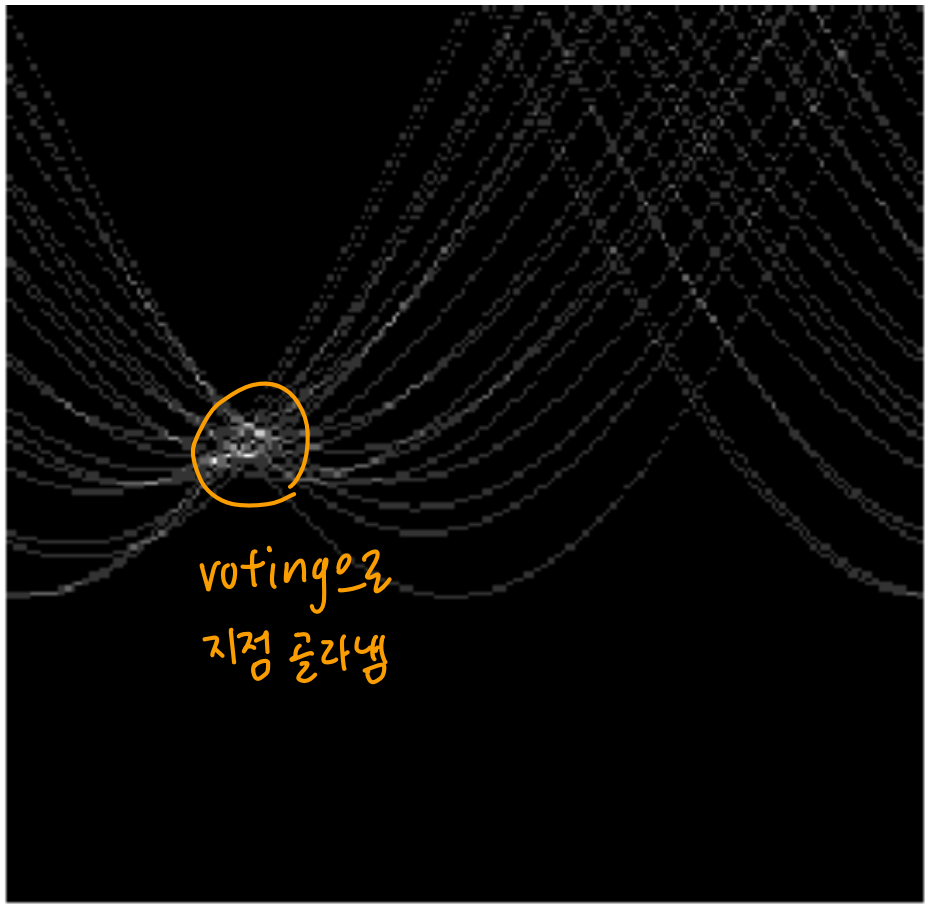


Slide from S. Savarese

Effect of noise



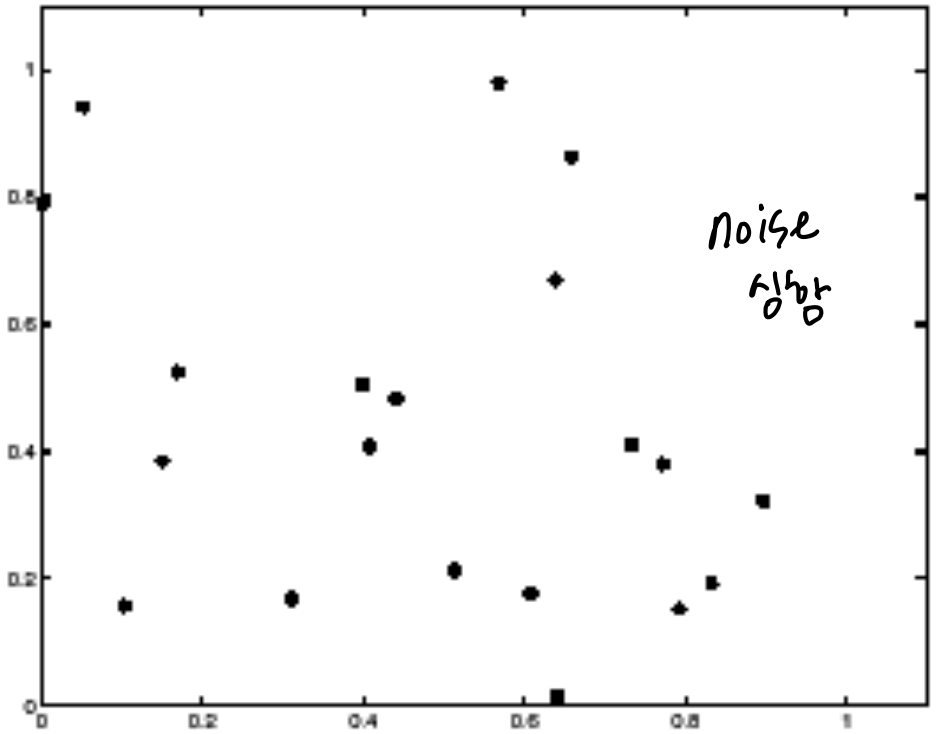
features



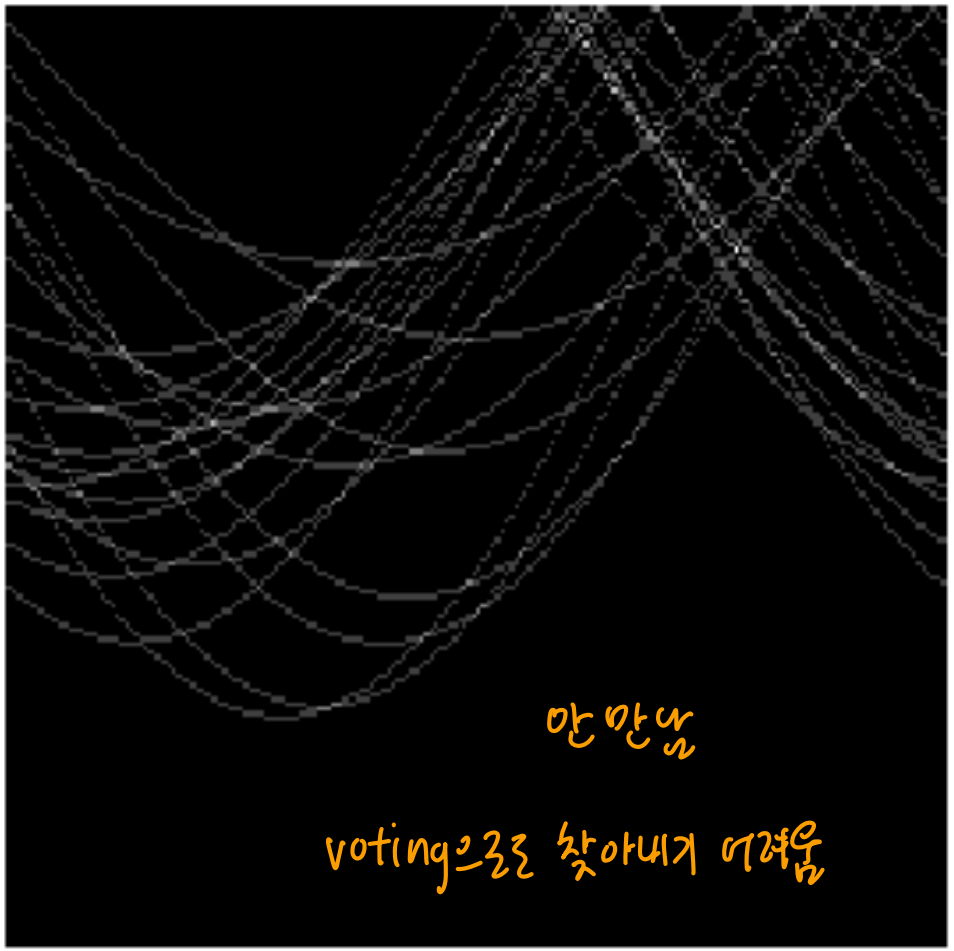
votes

- Peak gets fuzzy and hard to locate

Effect of large noise



features



votes

- Issue: spurious peaks due to uniform noise

1. Image → Edge (with Canny edge detector)

→ dominant lines 찾기 → edge detection

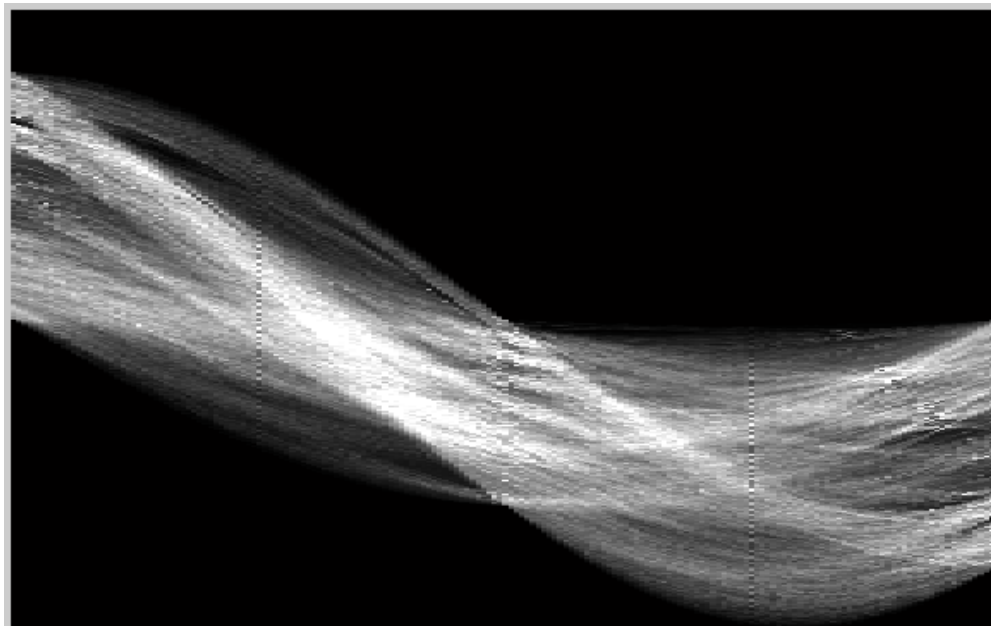
2. Canny → Hough votes

모든 line들을 hough space에서 표현

grid로 count

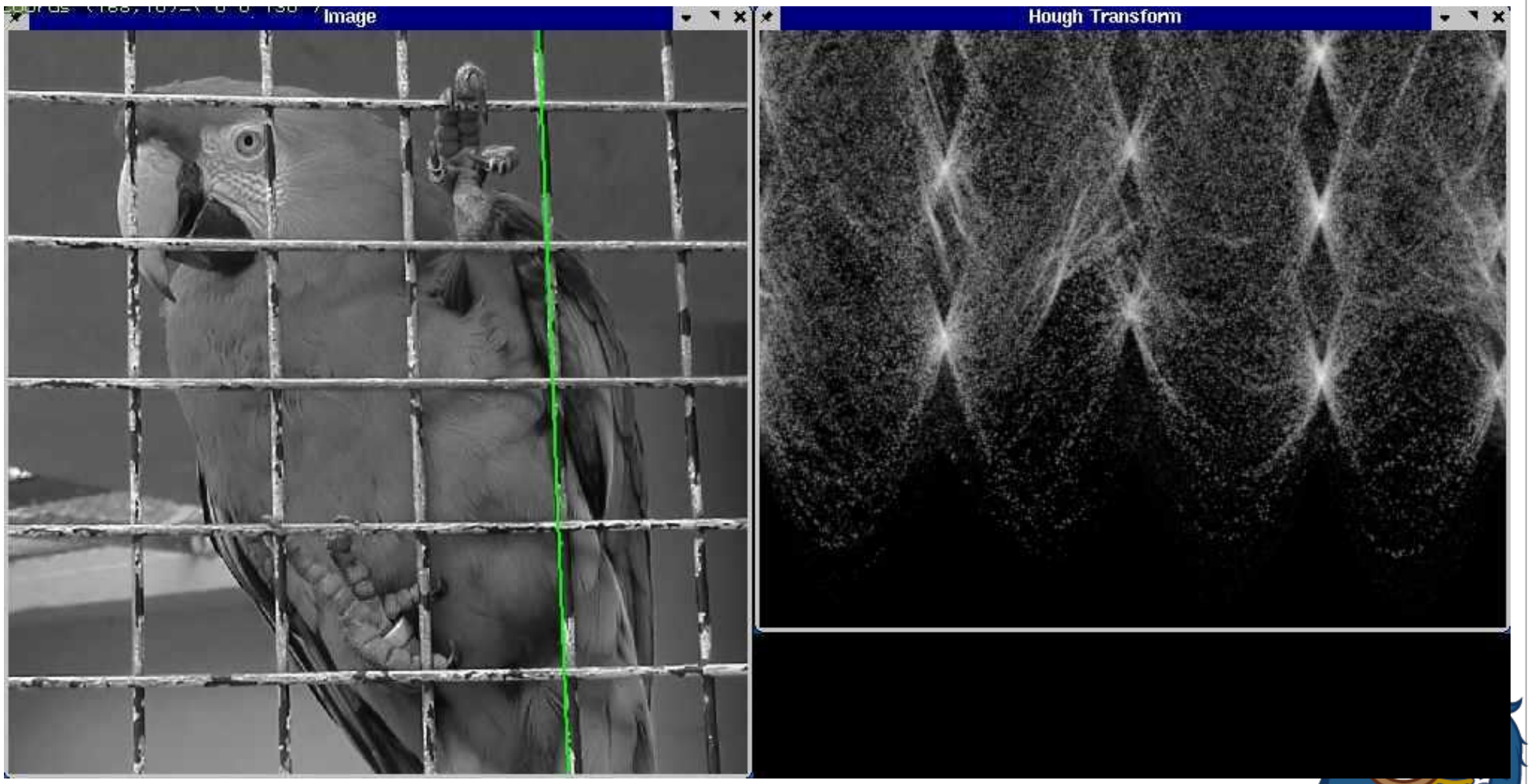
3. Hough votes → Edges

- Find peaks and post-process



cell안에 많이 들어온 dominant한 라인 찾음 →

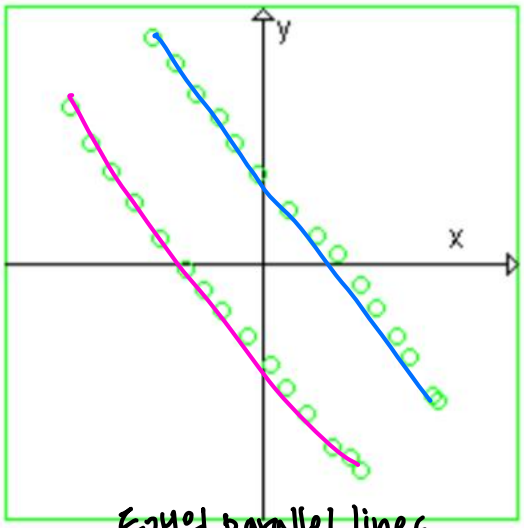
Hough transform example



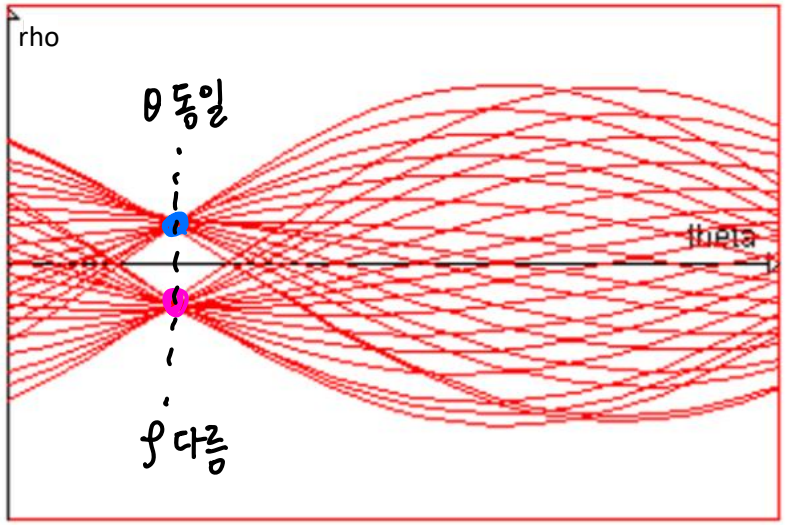
edge map → 밝은 부분을 레이더로 → wave로 돌림 → 밝은것 고르면.. dominant line 찾기

Properties

- What happens with parallel lines?



두개의 parallel lines
기울기는 같고 y절편만 다름

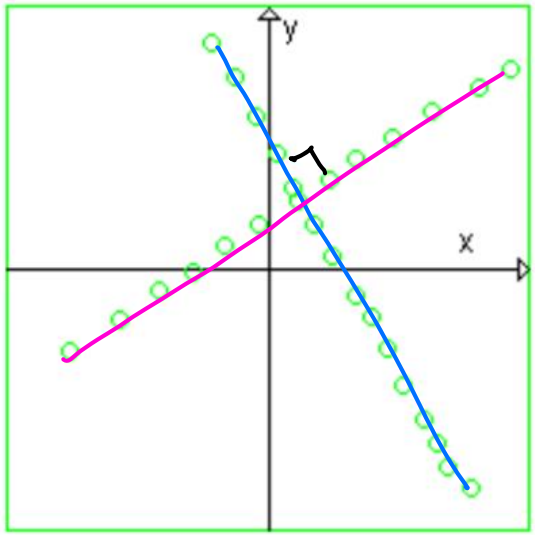


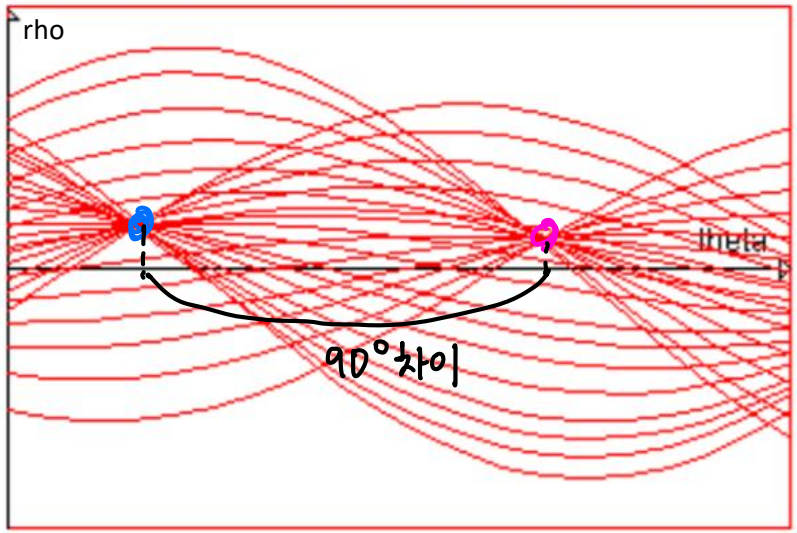
Same angles θ, different distances ρ

$$\rho = A \cdot \cos(\theta - \delta)\,, A = \sqrt{x^2 + y^2}, \delta = \tan^{-1}(y/x)$$

Properties

- What happens with perpendicular lines?





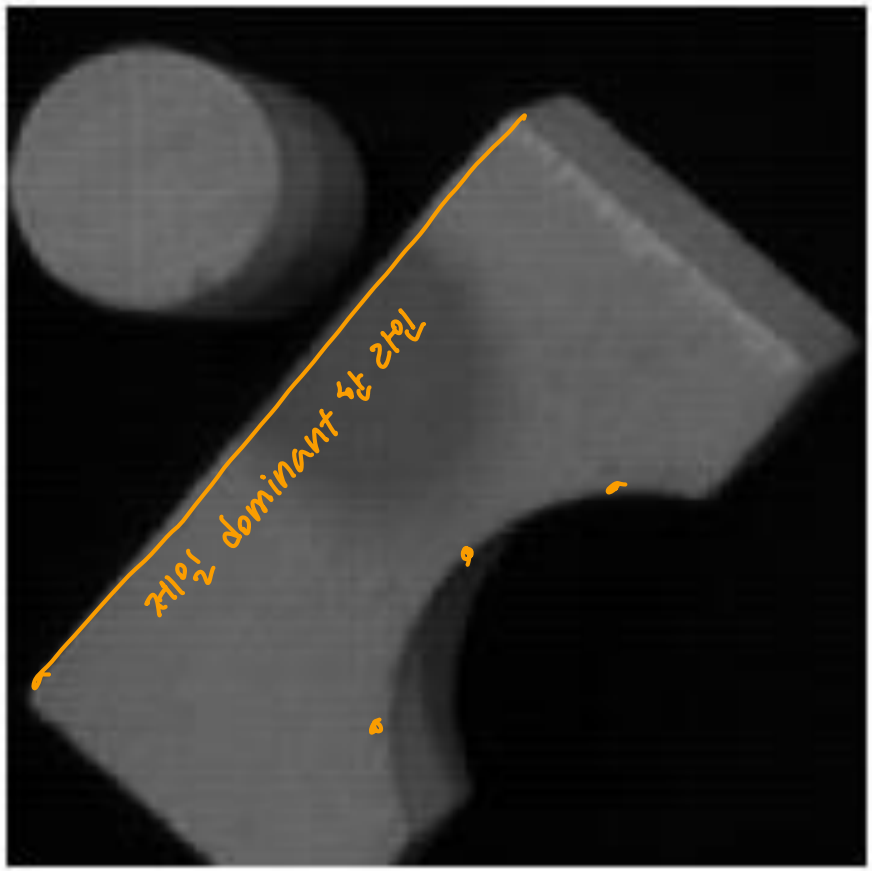
Angles θ are 90 deg apart, distances ρ are different

$$\rho = A \cdot \cos(\theta - \delta)\,, A = \sqrt{x^2 + y^2}, \delta = \tan^{-1}(y/x)$$

Parameter Space has Structure!

- Several lines

RANSAC / Hough transform
multiple fitting 찾을
(line 하나밖에 찾지못)
한번에 여러 모델 가질수 있음
병정 느낌



Practical details for line detection

- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude

- Choose a good grid / discretization
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket

    여러개의 라인이 들어오는 문제 발생 가능

  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets

    캐치가 안될수있음

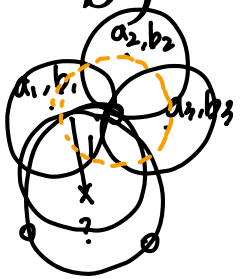
    → 실험적으로 grid 크기 적당히 설정 (Hyperparameter)

Hough transform for circles

라인 뿐만 아니라 여러 도형 가능

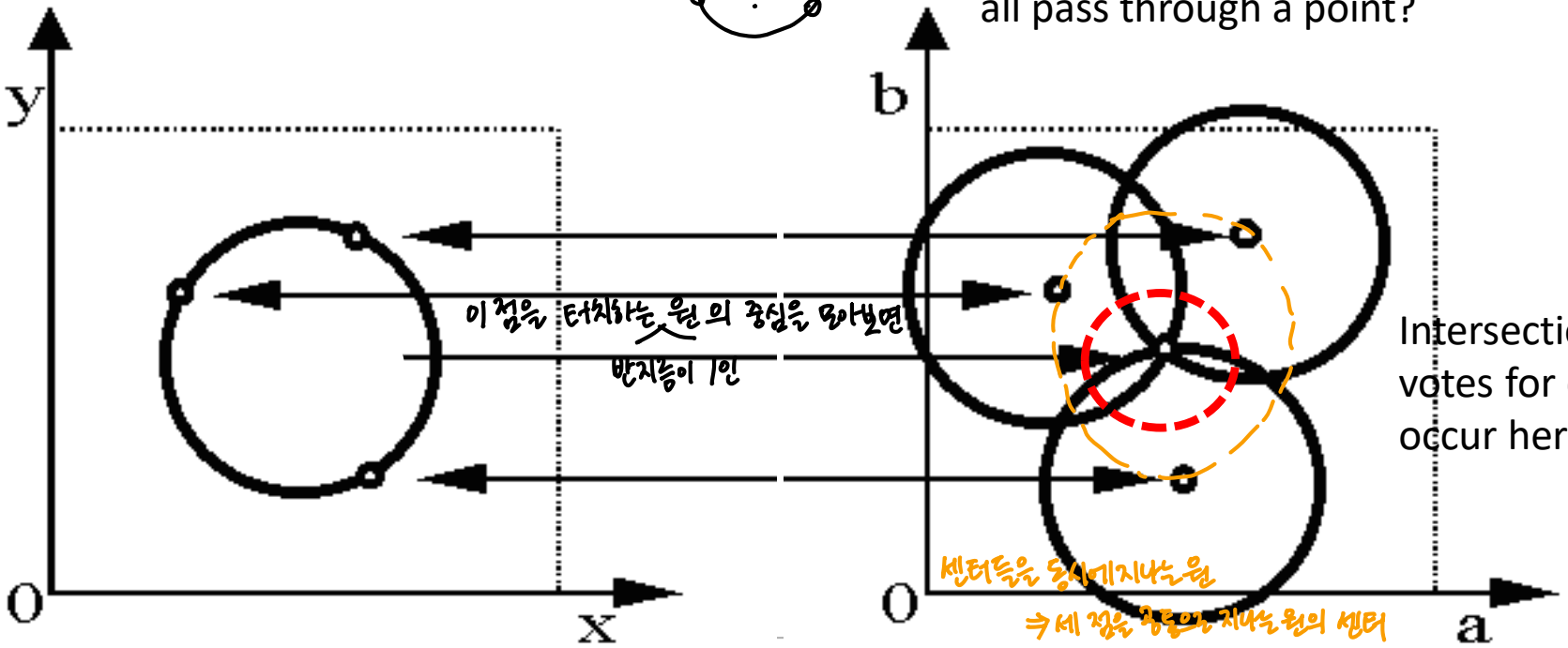
- Circle: center (a,b) and radius r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$



- For a fixed radius r

Equation of set of circles that all pass through a point?



Intersection: most votes for center occur here.

Image space

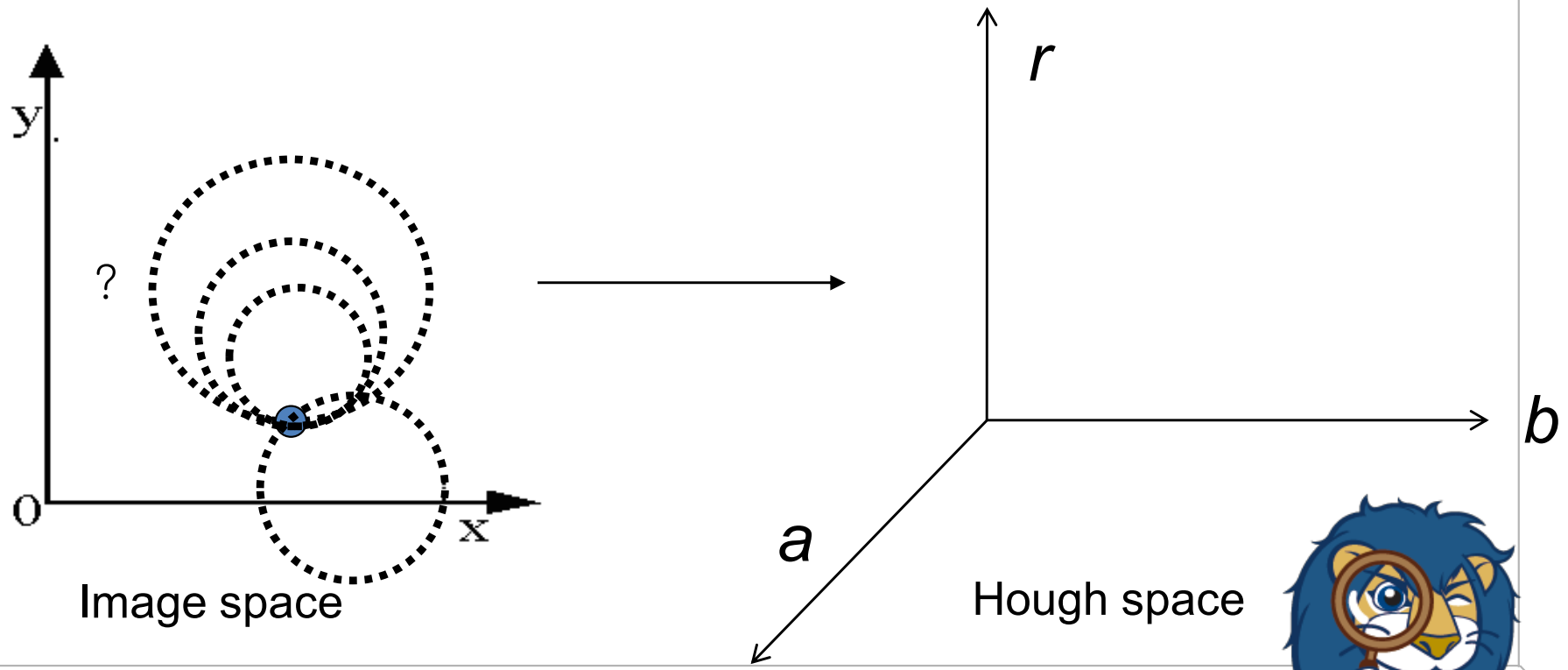
Hough space

Hough transform for circles

- Circle: center (a,b) and radius r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For an unknown radius r, → 변수 3개 필요



Hough transform for circles

- Circle: center (a,b) and radius r

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

- For an unknown radius r,

원선분 표면을 가장 많이 교차하는 부분이 반지름

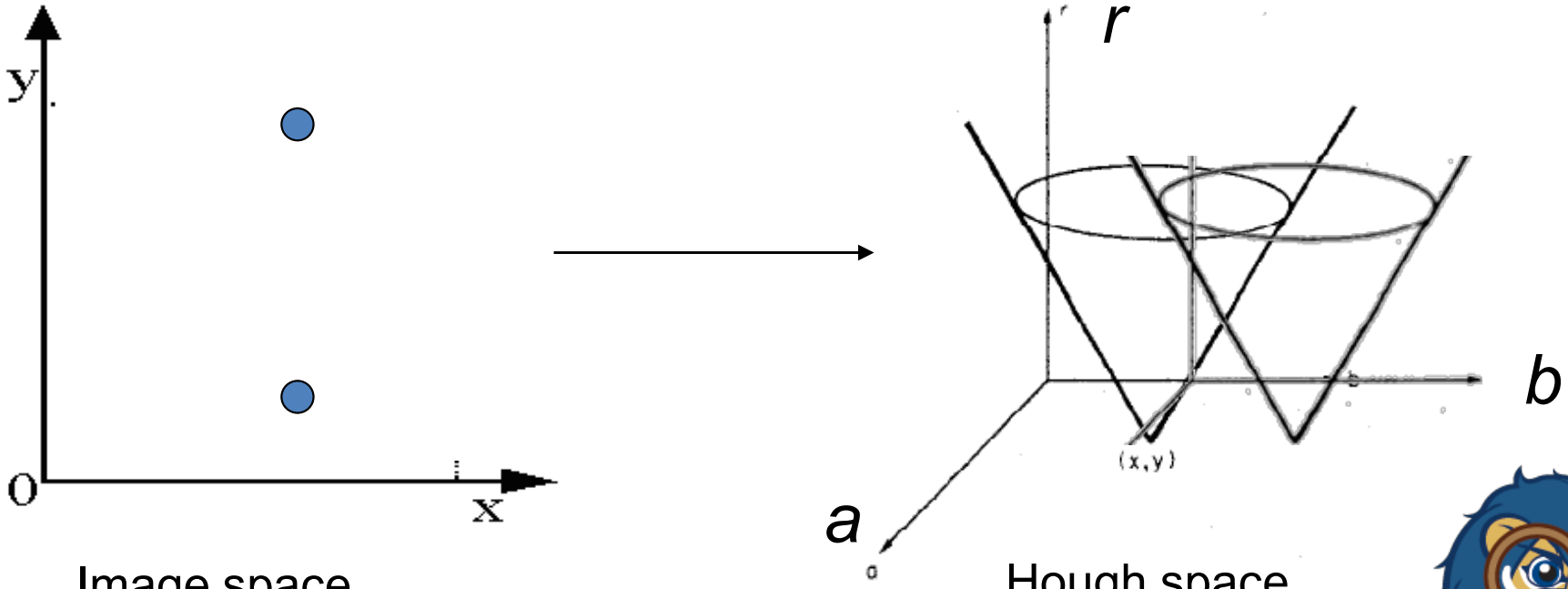
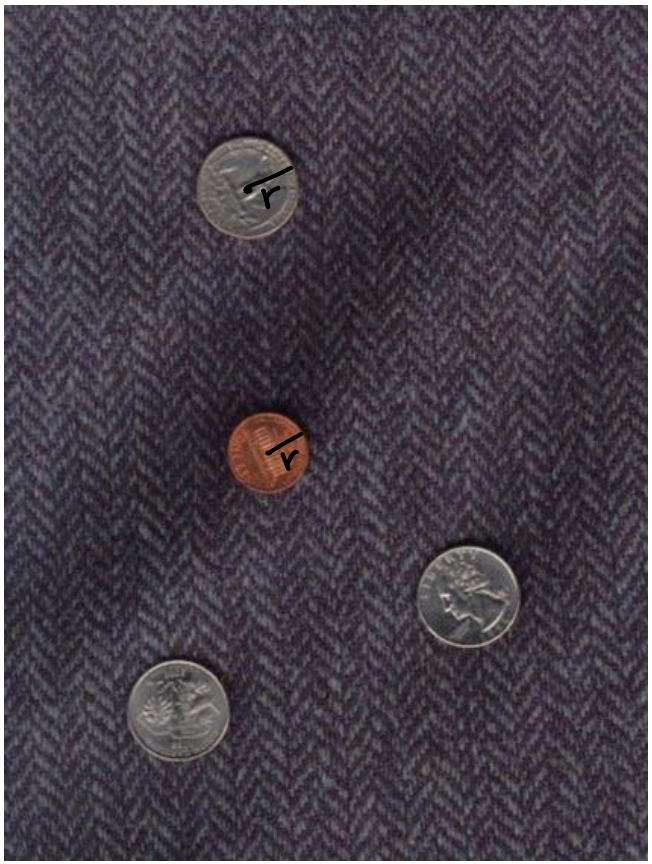


Image space

Hough space

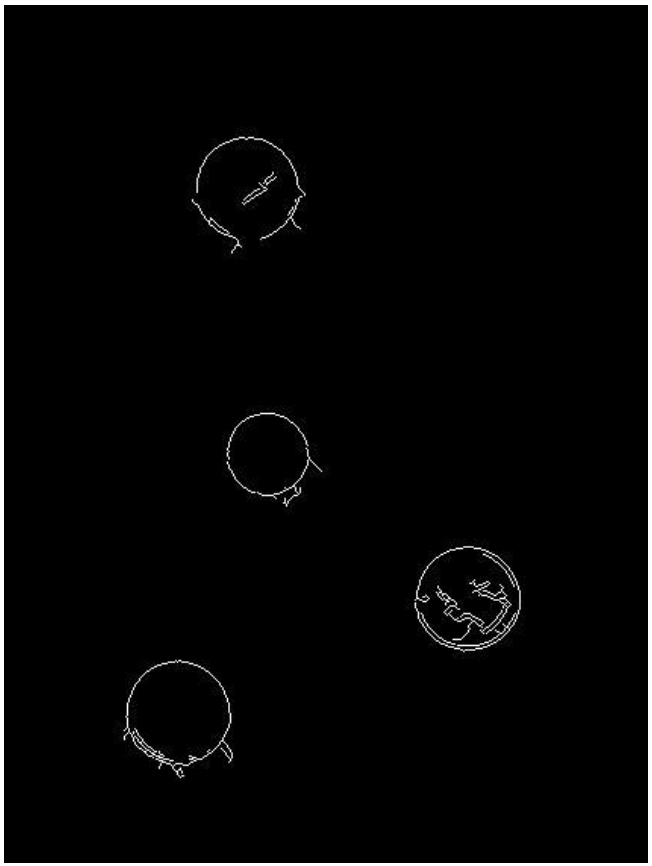
Example: detecting circles with Hough

Original



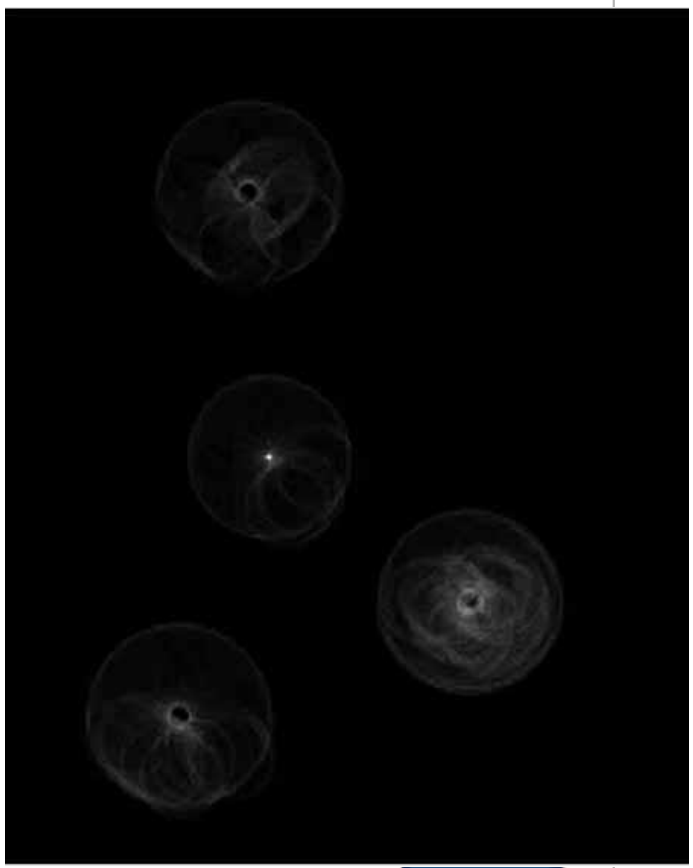
r 알려짐

Edges



H map 뜯어냄

Votes: Penny (small r)

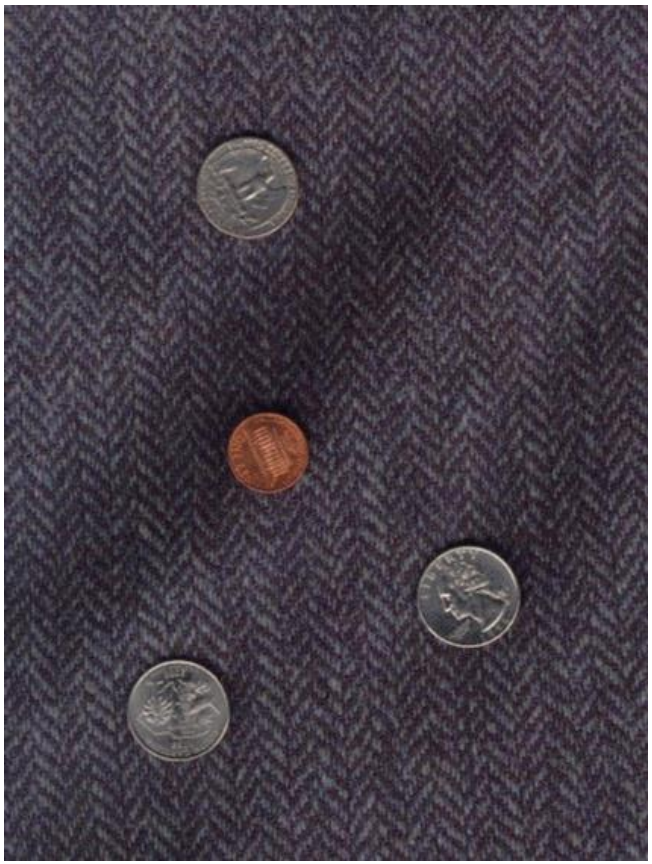


파라미터 (a, b) 찾기

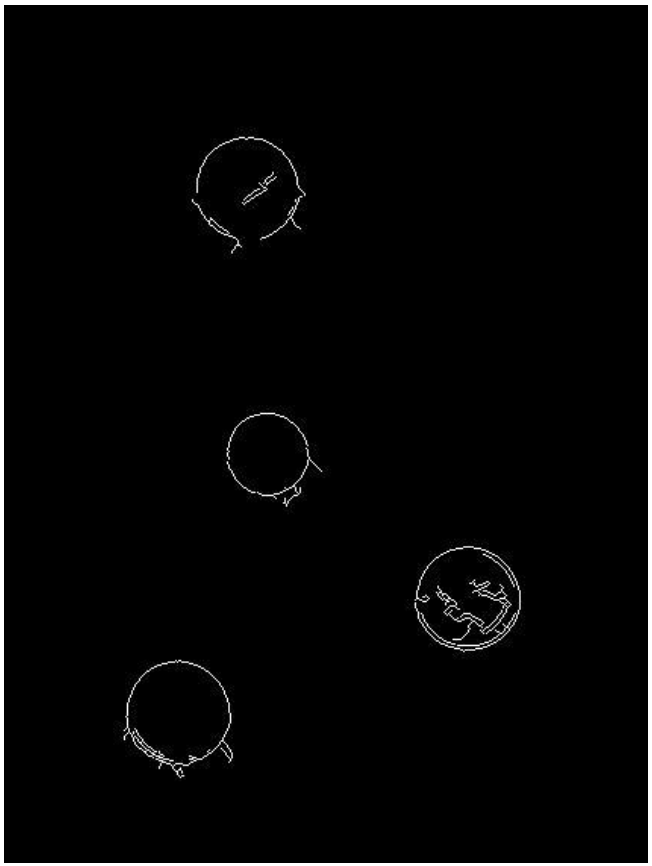
Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

Example: detecting circles with Hough

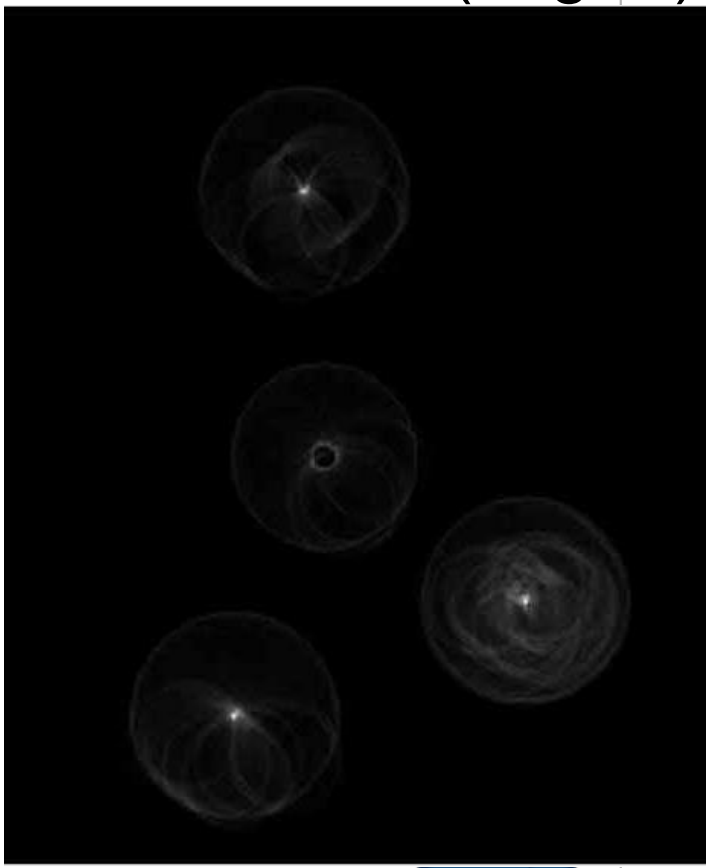
Original



Edges



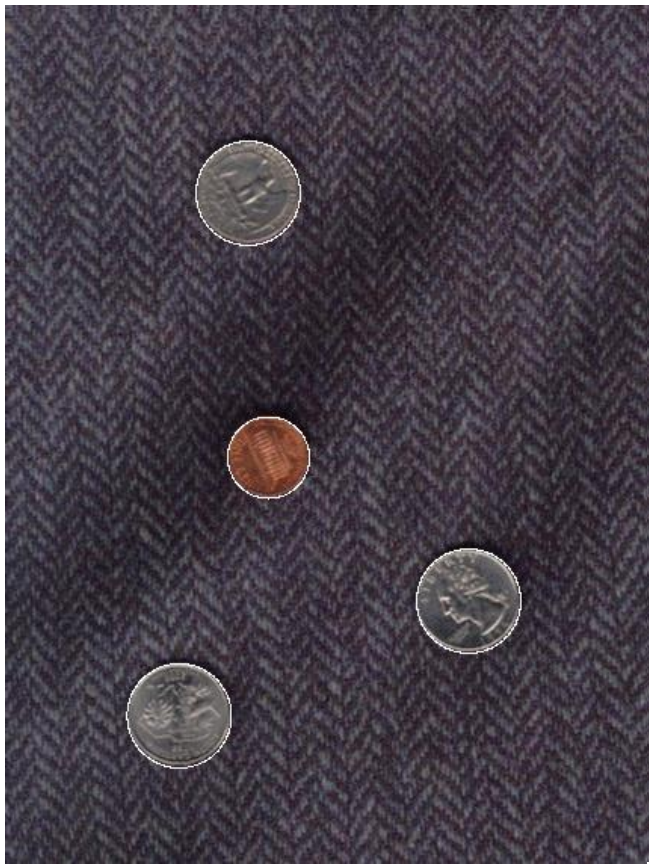
Votes: Quarter (large r)



Note: a different Hough transform (with separate accumulators) was used for each circle radius (quarters vs. penny).

Example: detecting circles with Hough

Combined detections



Summary

- Good
  - Robust to outliers: each point votes separately
  - Fairly efficient (much faster than trying all sets of parameters)
  - Provides multiple good fits

여러가지 모델을 한번에 찾을수 있음 multiple model fitting 가능

- Bad
  - Some sensitivity to noise
  - Bin size trades off between noise tolerance, precision, and speed/memory
    - Can be hard to find sweet spot
  - Not suitable for more than a few parameters
    - grid size grows exponentially

data 에 노이즈가 있으면 best parameter 찾기 어려워질

- Common applications
  - Line fitting (also circles, ellipses, etc.)
  - Object instance recognition (parameters are position/scale/orientation)
  - Object category recognition (parameters are position/scale)

파라미터 찾기 → 허프 트랜스폼을 젤 먼저 고려

Thank you!