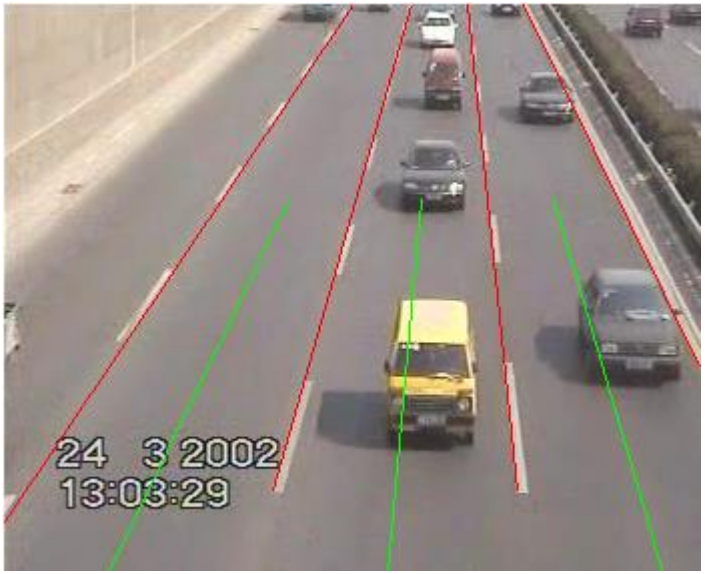
Finding lines – part 2

Lihi Zelnik-Manor, Computer Vision

Today

- Edge detection
  - Canny edge detector
  - Berkeley edge probability
- Line fitting
  - Hough transform
  - (Generalized) Hough transform application
  - RANSAC

Fitting a parametric model to data



parameters
parametric equation
parameters

Image credit Zhaozheng Yin @ wisc.edu

Image credit Yuan-Liang Tang on Mathworks

Fitting a parametric model to data

- Design questions:
  - What is a good model to represent our data?
  - Do we plan to fit multiple instances?
- Challenges:
  - Which features belong to the model? To which instance?
  - How many instances are there?
  - Computational complexity (typically we cannot examine all possible models).

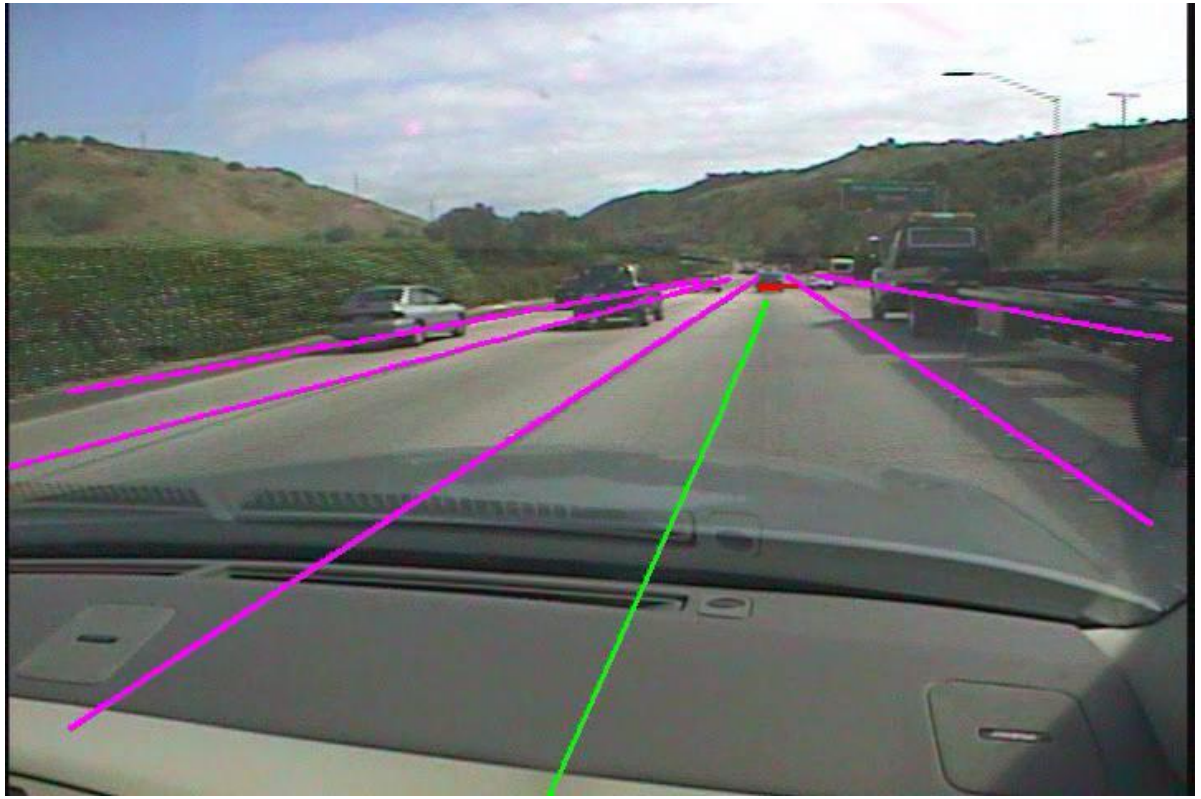
Line fitting applications

detection of power lines in helicopter navigation systems

Image credit: Horev et al. SIAM'15

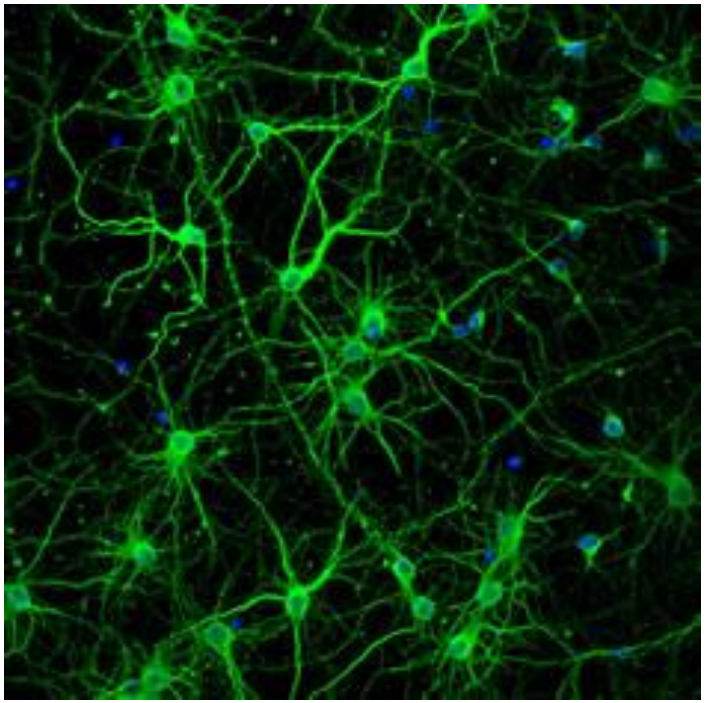
Line fitting applications

lane detection from car cameras in crashpreventing systems



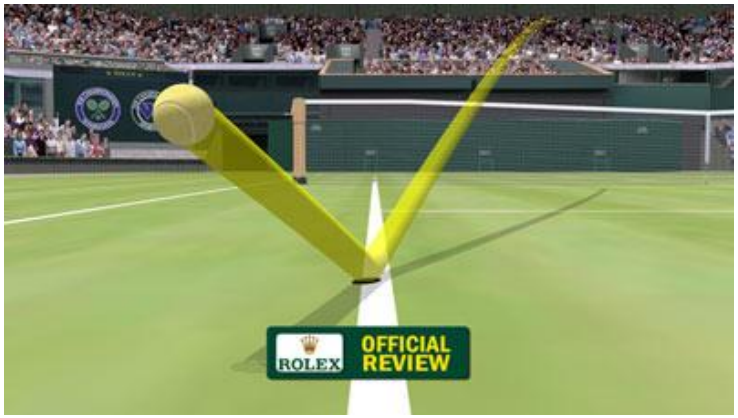
Line fitting applications

detection of long filaments in high-throughput biological imaging



Line fitting applications

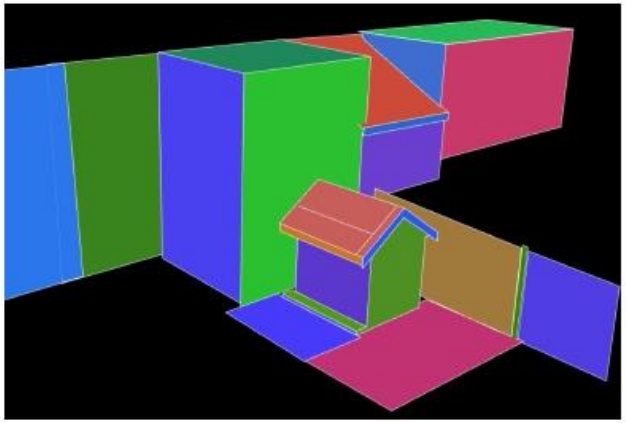
Sports



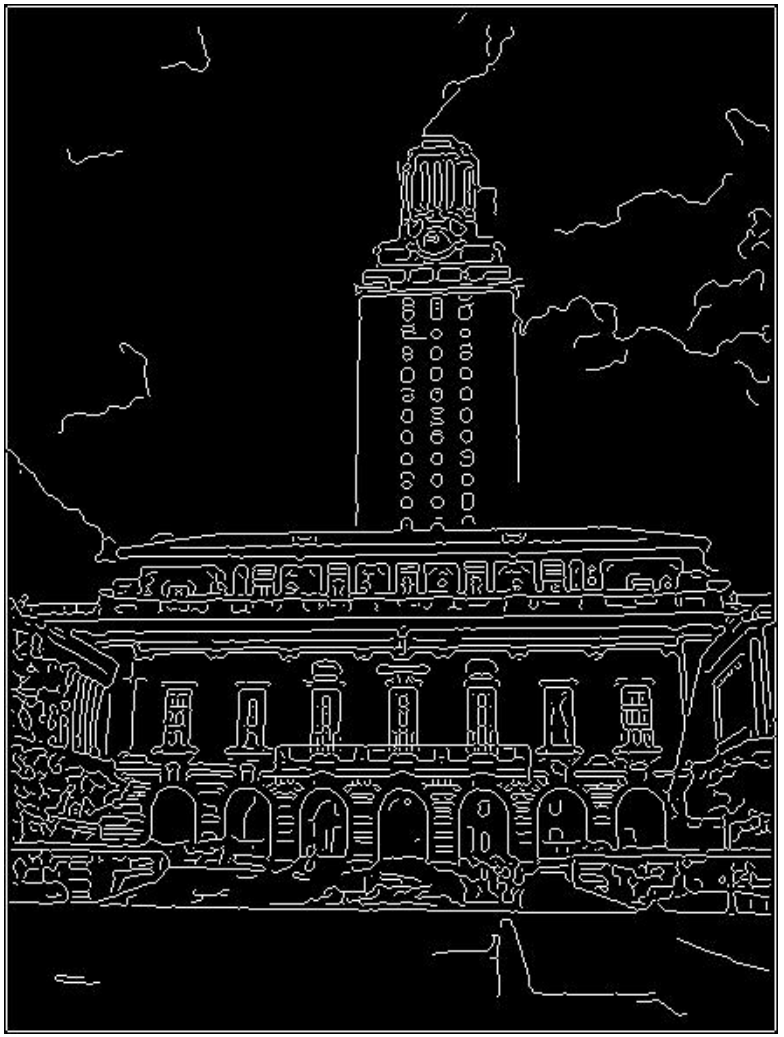
Line fitting applications

“Interactive 3D Architectural Modeling from Unordered Photo Collections” Sinha et al. 2008



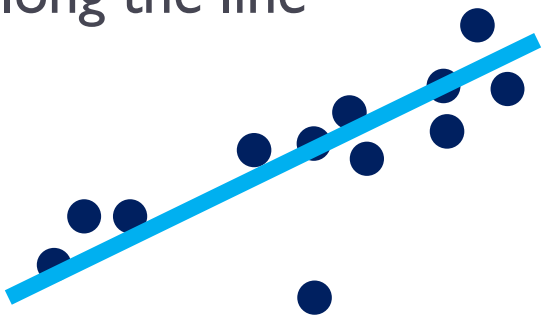


Challenges of line fitting

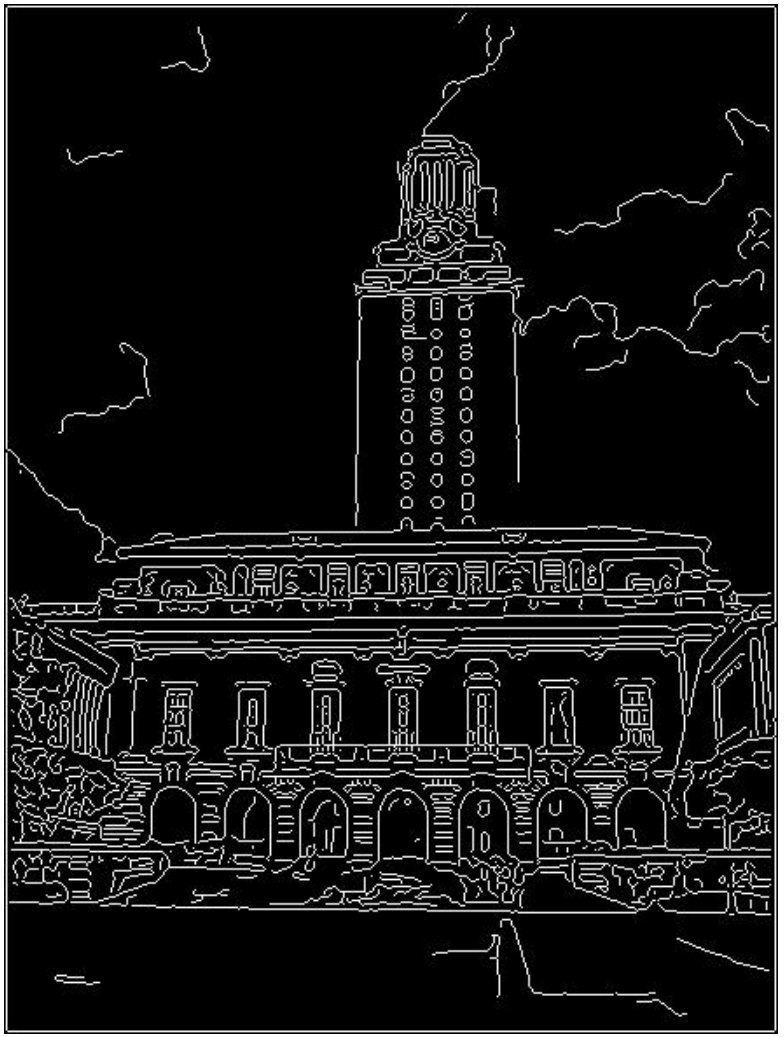


Challenges of line fitting

- Which points on which line?
- Noisy edge detection:
  - Clutter
  - Missed parts
  - Points are only approximately along the line

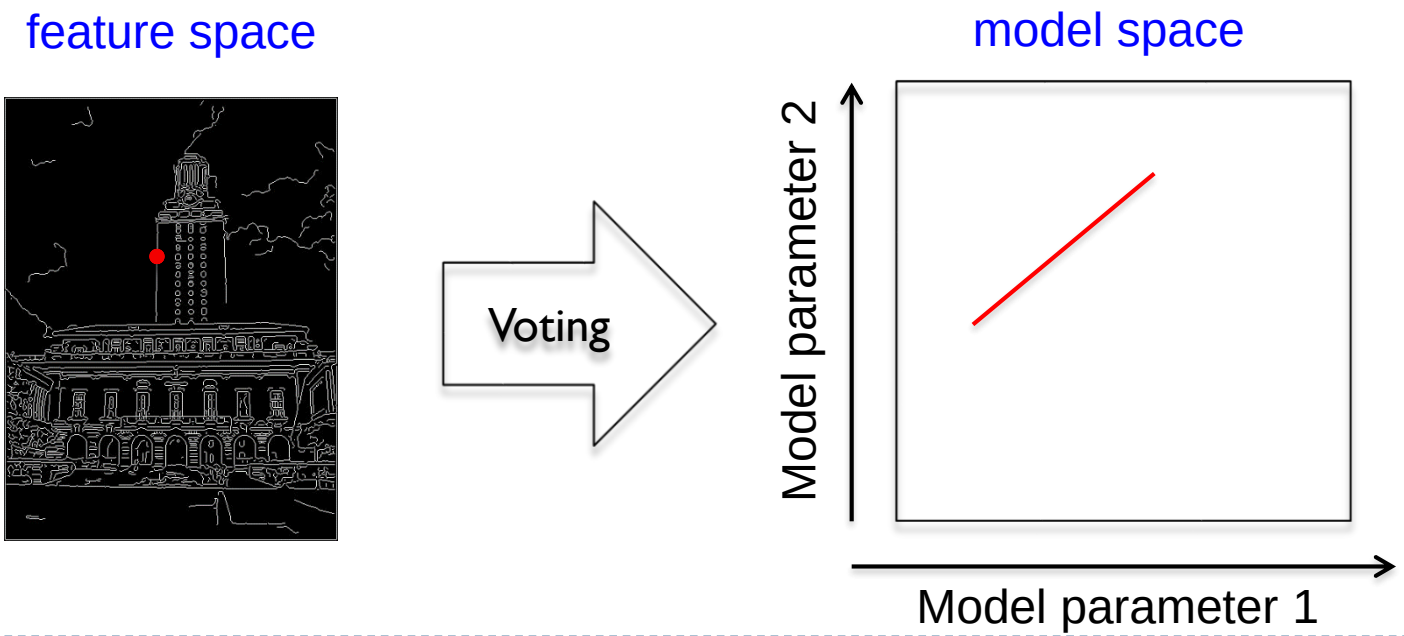


- Large search space.
- How many lines are there?



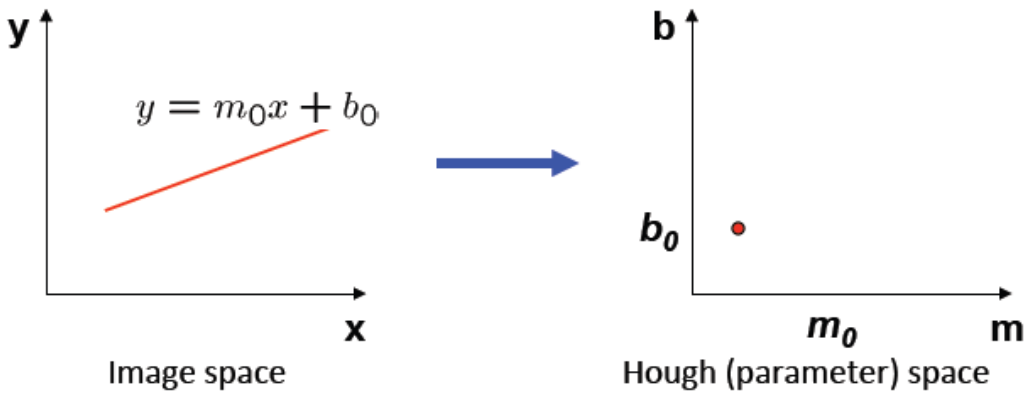
Voting

- Problem:
  - We cannot try all possible models
- Solution by voting:
  - Features (points) vote for models they are compatible with
  - Search for models with lots of votes



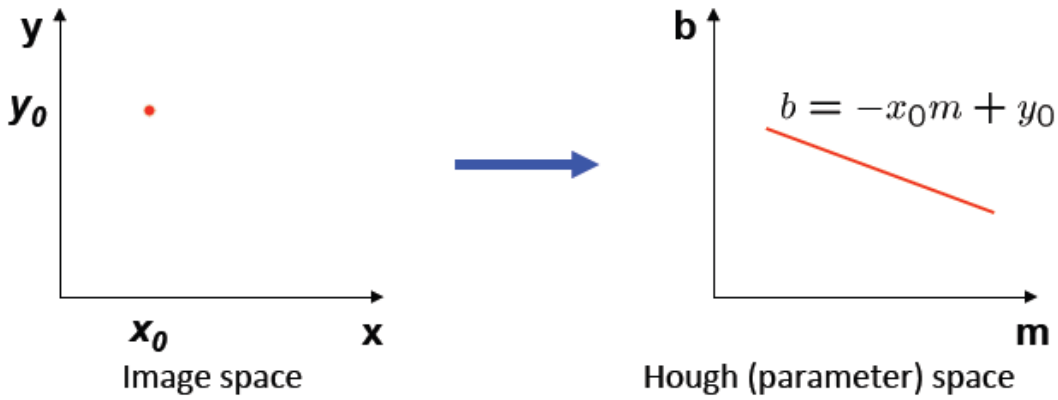
The Hough transform

- Transformation from image space (x, y) to Hough space (m, b)
- A line in the image corresponds to a point in Hough space
  - Image → Hough:
    Given a set of points (x, y) find all (m, b) such that $y = mx + b$



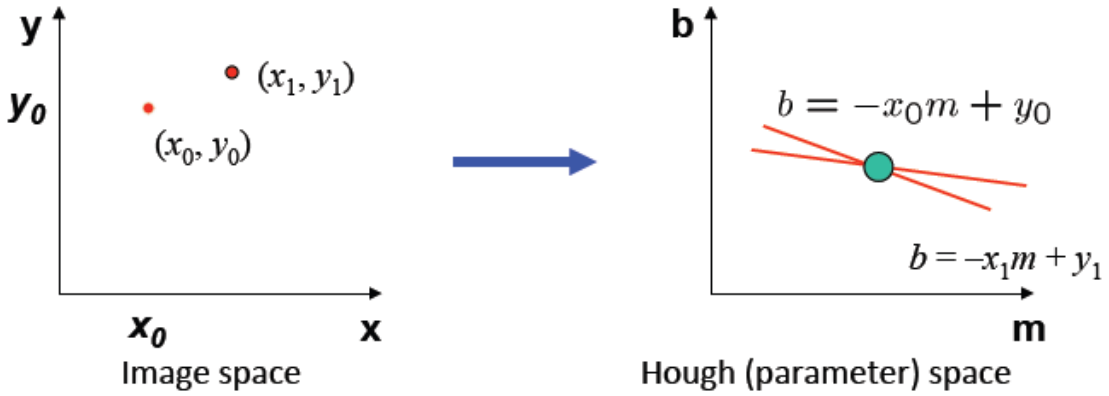
The Hough transform

- Transformation from image space (x, y) to Hough space (m, b)
- A line in the image corresponds to a point in Hough space
  - Image → Hough:
    Given a set of points (x, y) find all (m, b) such that $y = mx + b$
  - Hough → Image:
    A point (x_0, y_0) in the image is the solution of $b = -x_0 m + y_0$



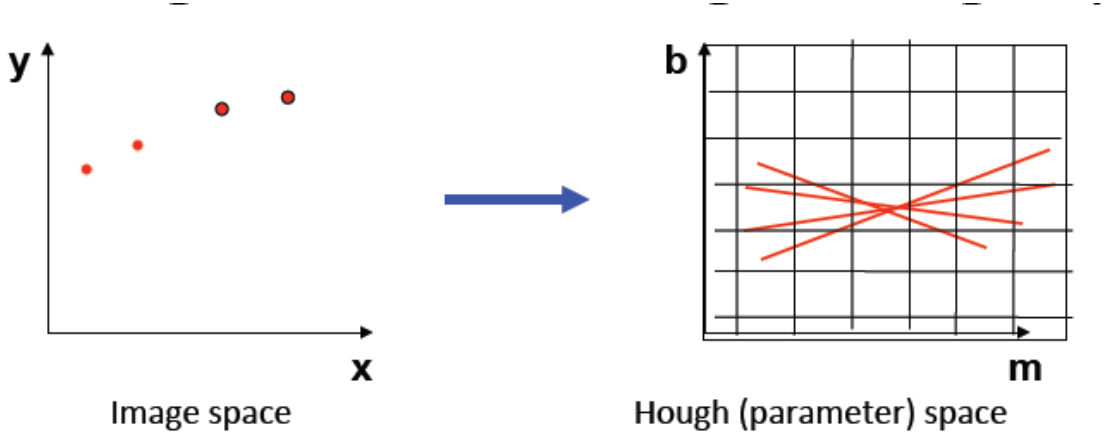
The Hough transform

- Transformation from image space (x, y) to Hough space (m, b)
- A line in the image corresponds to a point in Hough space
  - Image → Hough:
    Given a set of points (x, y) find all (m, b) such that $y = mx + b$
  - Hough → Image:
    A point (x_0, y_0) in the image is the solution of $b = -x_0 m + y_0$
- The line that contain both points (x_0, y_0) and (x_1, y_1) is the intersection of the lines $b = -x_0 m + y_0$ and $b = -x_1 m + y_1$



Finding lines with the Hough transform

- Discretize Hough space
- Each edge point votes for all possible parameters in Hough space
- Parameters with lots of votes indicate lines in image space



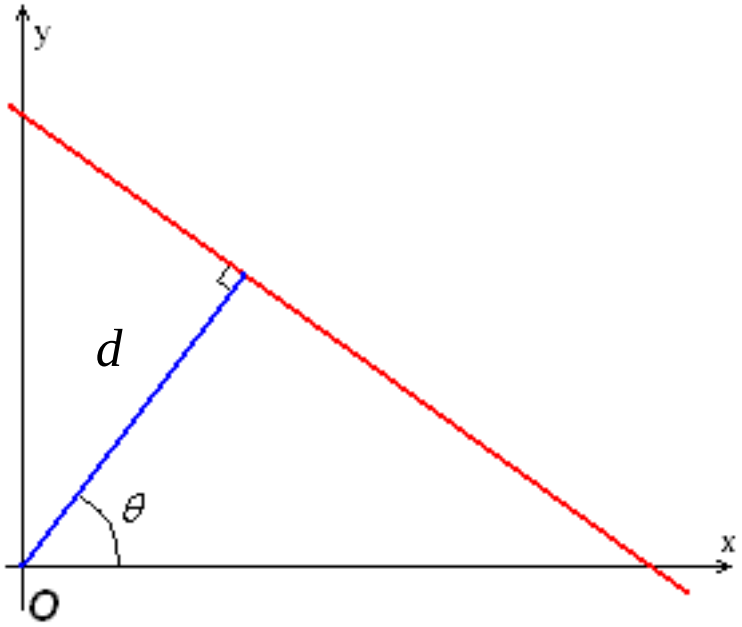
Polar representation for lines

- Problem:
  The familiar line equation $y = mx + b$ is problematic:
  - Can take infinite values
  - Undefined for vertical lines

- Solution:
  Use polar representation

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

$$\theta \in [0, \pi), \quad d \in R$$

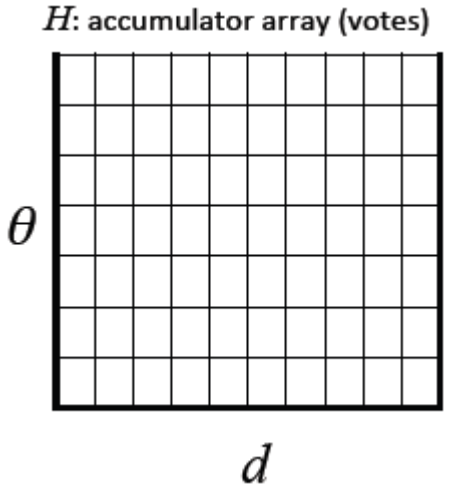


The Hough-transform algorithm

- Use polar representation

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

- Quantize Hough space



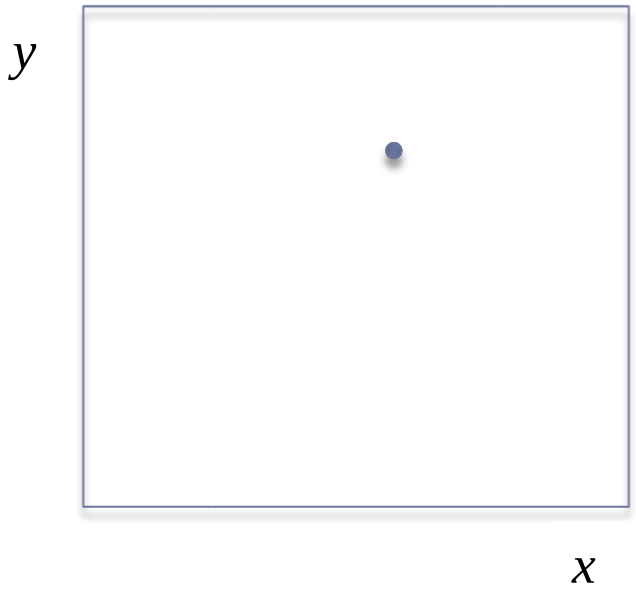
- Loop:

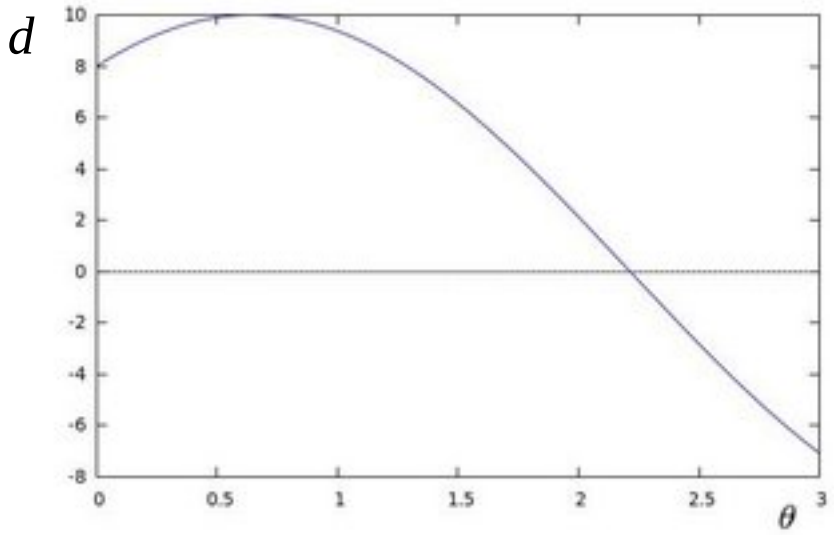
1. Initialize $H[d,\theta] = 0$
2. For each edge point (x,y) in the image $H[d,\theta]$ += 1 for all lines that go through it
3. Find bins $H[d,\theta]$ with maximum value

Example

For a given point (x_0,y_0)

$$d(q) = x_0 \cos q + y_0 \sin q$$

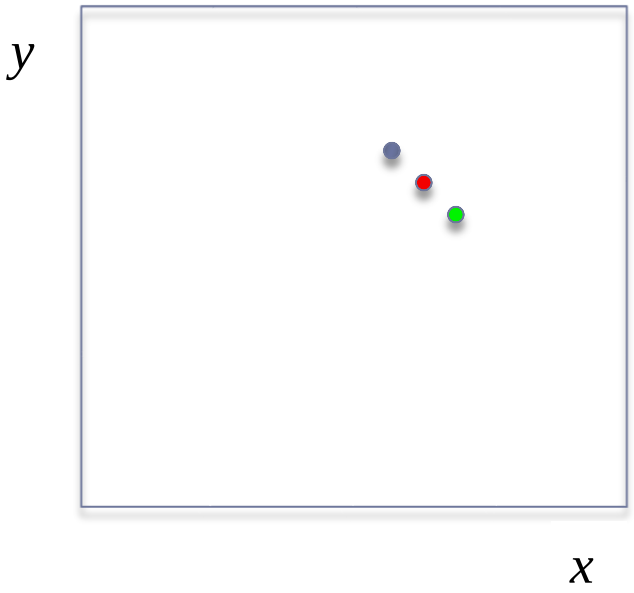


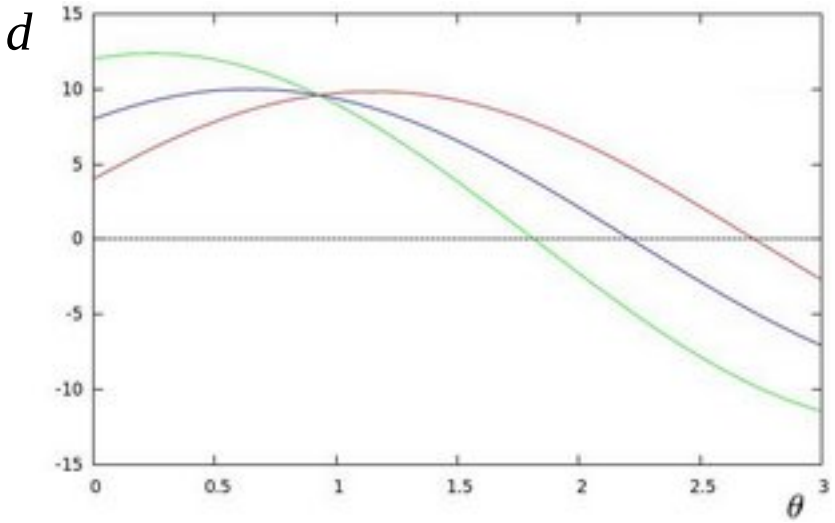


Example

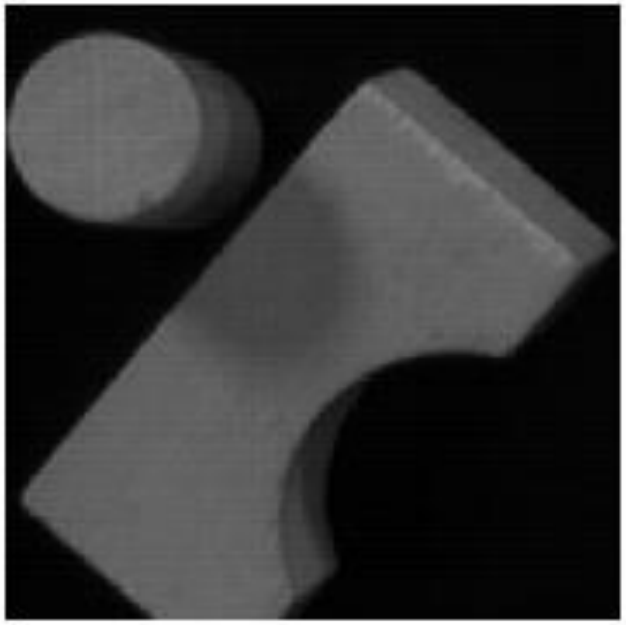
For a given point (x_0,y_0)

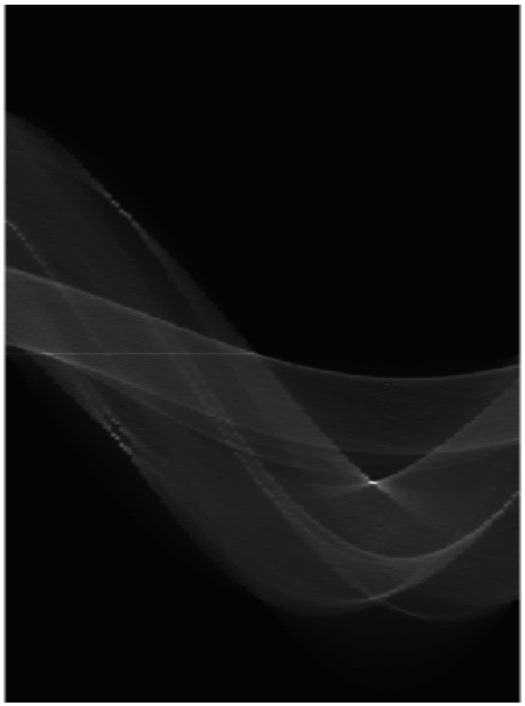
$$d(q) = x_0 \cos q + y_0 \sin q$$





Example: an image with straight lines

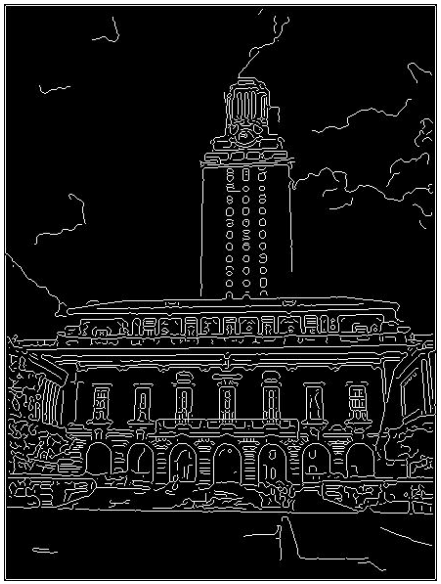


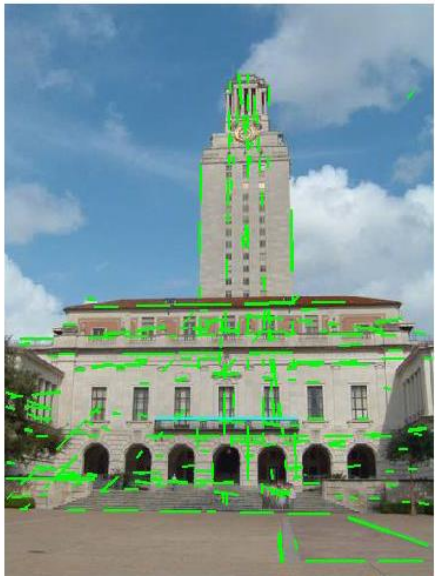


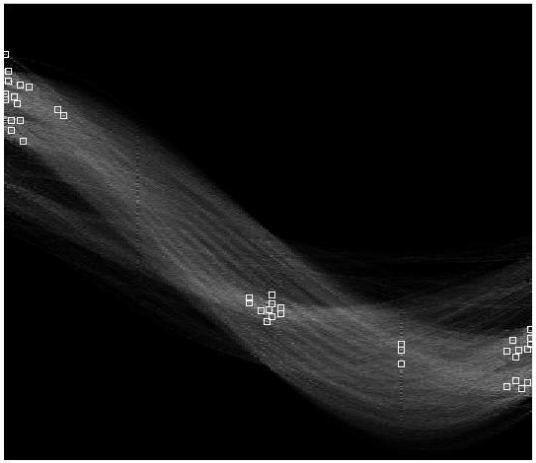
Properties

- Noise and clutter votes are inconsistent, so will not accumulate.
- Can handle occlusions if not all points are present as long as model gets enough votes.
- Efficient.

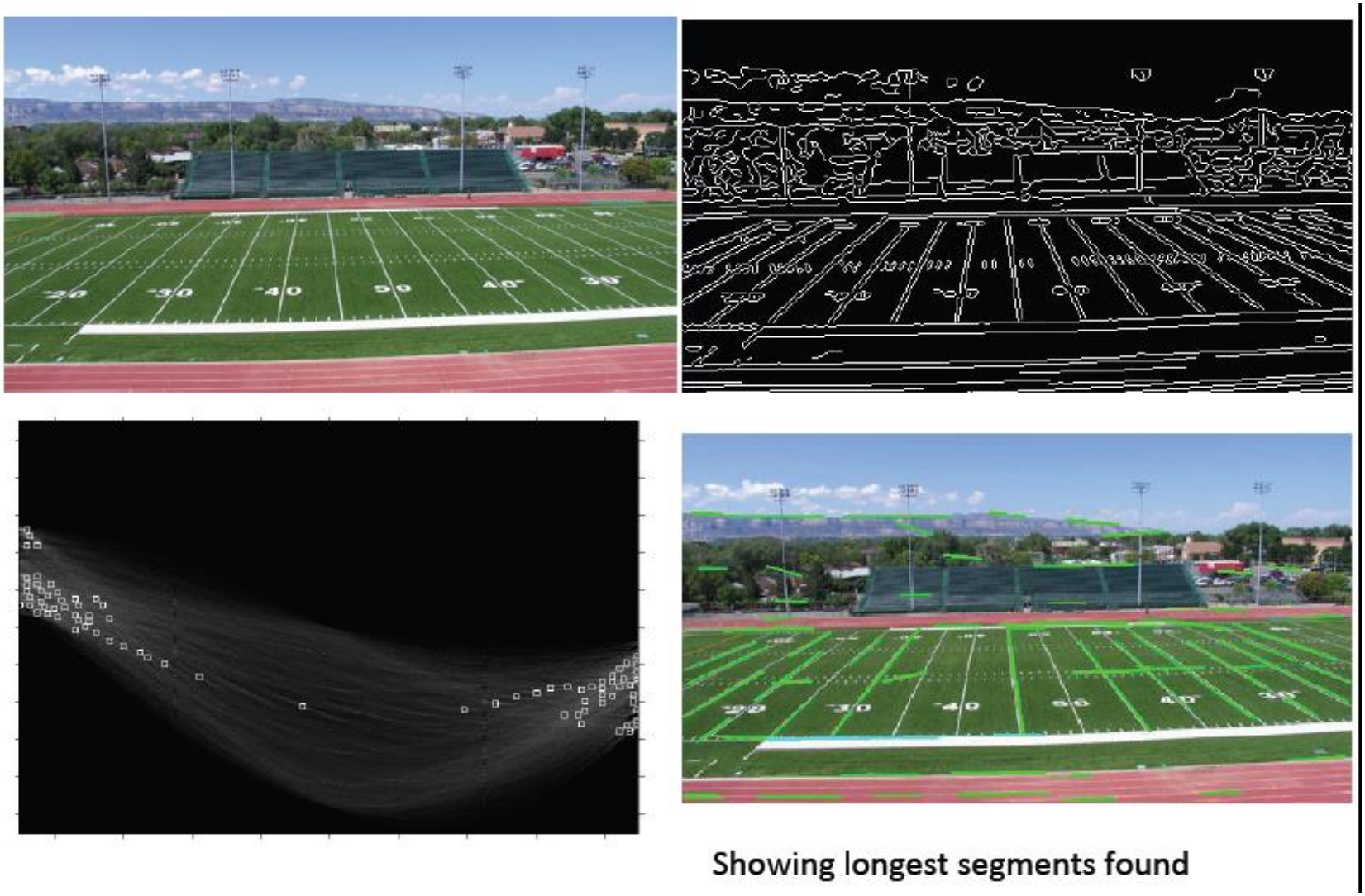
Example: a real image





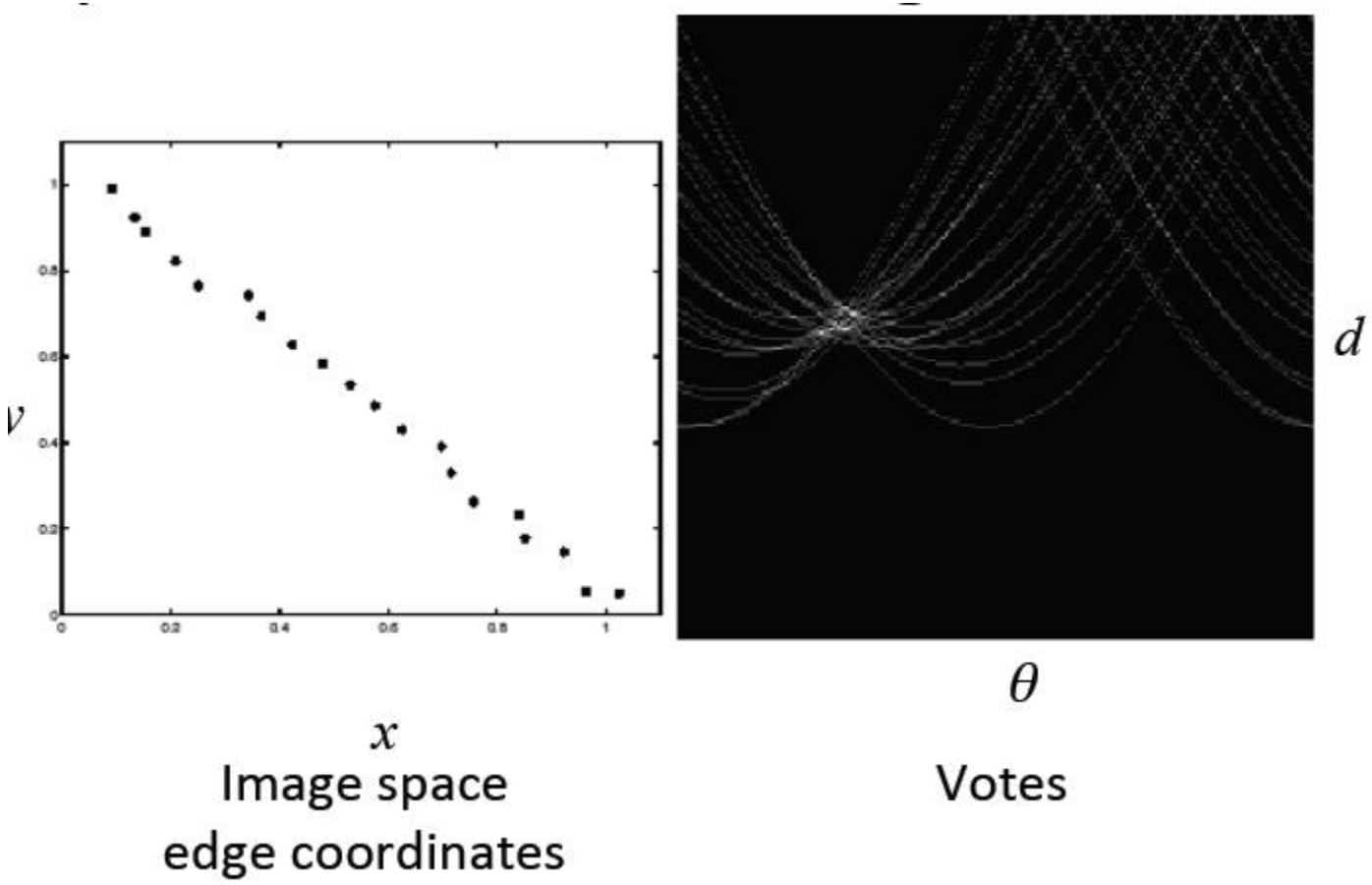


Example: a real image



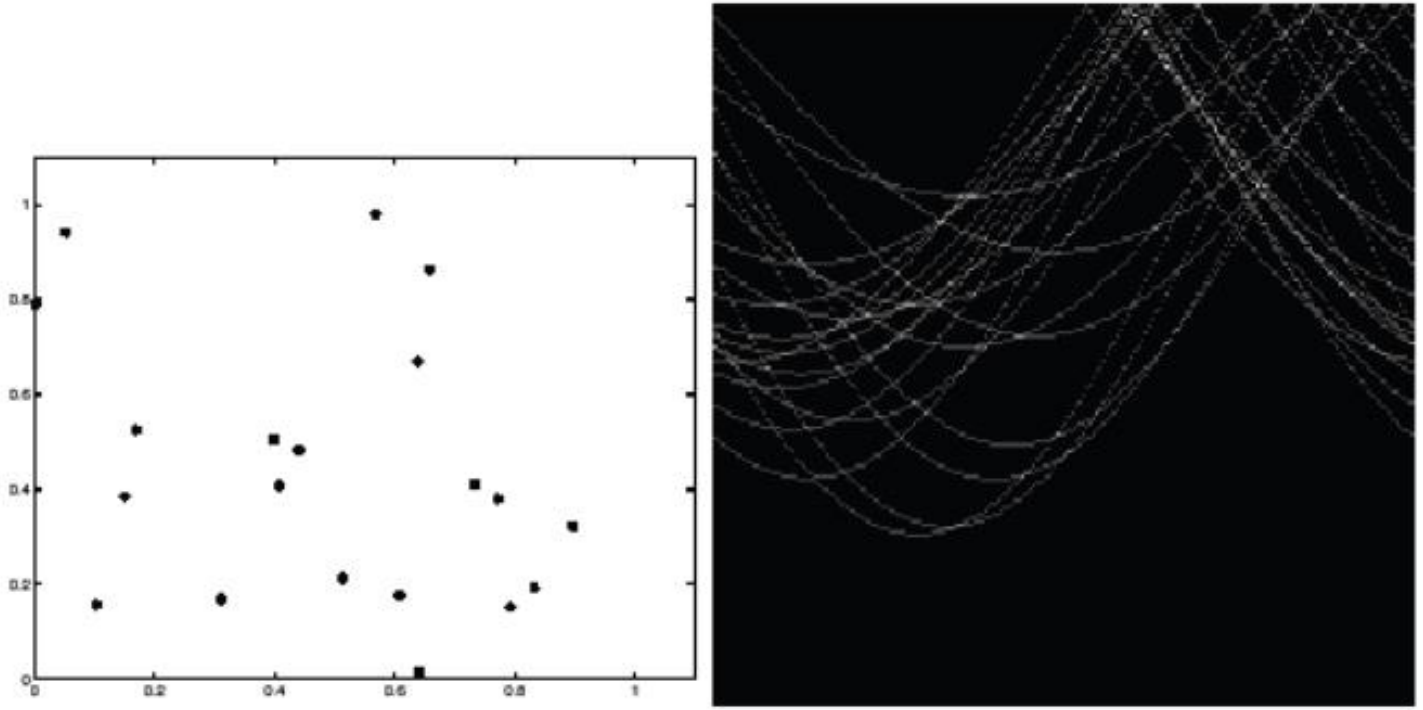
Impact of noise on Hough transform

- What difficulties does noise introduce?



Impact of noise on Hough transform

- Here everything is "noise" but we still see peaks in the vote space



Voting: Practical tips

- Use only trustworthy points
  - E.g., edges points with significant gradient magnitude (alternatively weight votes)
  - Szeliski suggests using edgels instead of points
- Choose a good quantization grid
  - Not too coarse – too many lines fall in the same bucket
  - Not too fine – collinear points vote for different lines
- Smooth the voting (vote also for neighbors)
- Non-maxima suppression
- Refit line using accumulated votes
- Reduce number of parameters, if possible

Hough transform summary

- Pros
  - Can handle occlusions
  - Some robustness to noise
  - Can detect multiple lines in a single pass over the image

- Cons
  - Clutter can produce spurious peaks in parameter space
  - Hard to select the right quantization

Generalized Hough Transform

- Can be extended to other parametric models such as: circles, ellipses, rectangles etc.
- Complexity increases exponentially with the number of parameters.
- Can be used to detect complex non-parametric models as described in Leibe et al. “Combined object categorization and segmentation with an implicit shape model”.

Today

- Edge detection
  - Canny edge detector
  - Berkeley edge probability
- Line fitting
  - Hough transform
  - RANSAC

RANdom SAmple Consensus [Fischler & Bolles 1981]

- Key ideas:
  - Look for "inliers" and use only them
  - If we fit a model to "outliers" we will not get a good fitting

RANSAC algorithm

Loop:

1. Randomly select a group of points
2. Fit a model to the selected group
3. Find the inliers of the computed model
4. If number of inliers is large enough re-compute model using only inliers
5. Compute number of inliers of updated model

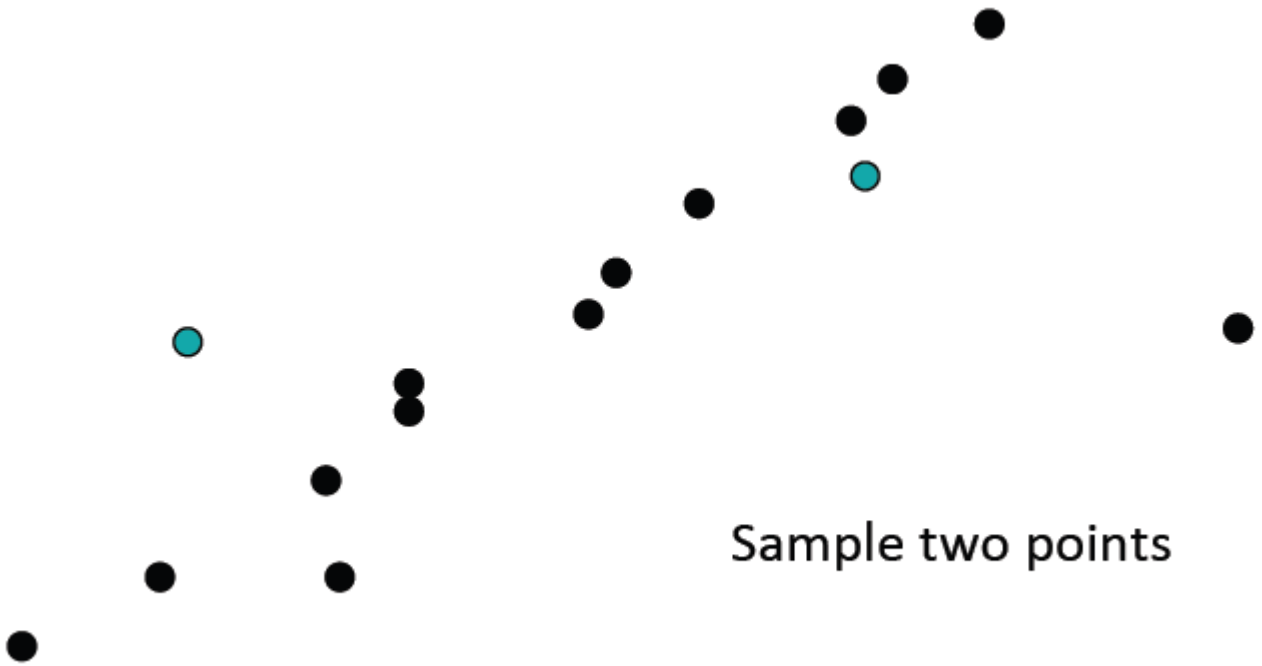
The winner: model with the largest number of inlier

Line fitting with RANSAC

- Input:
  A set of edge points

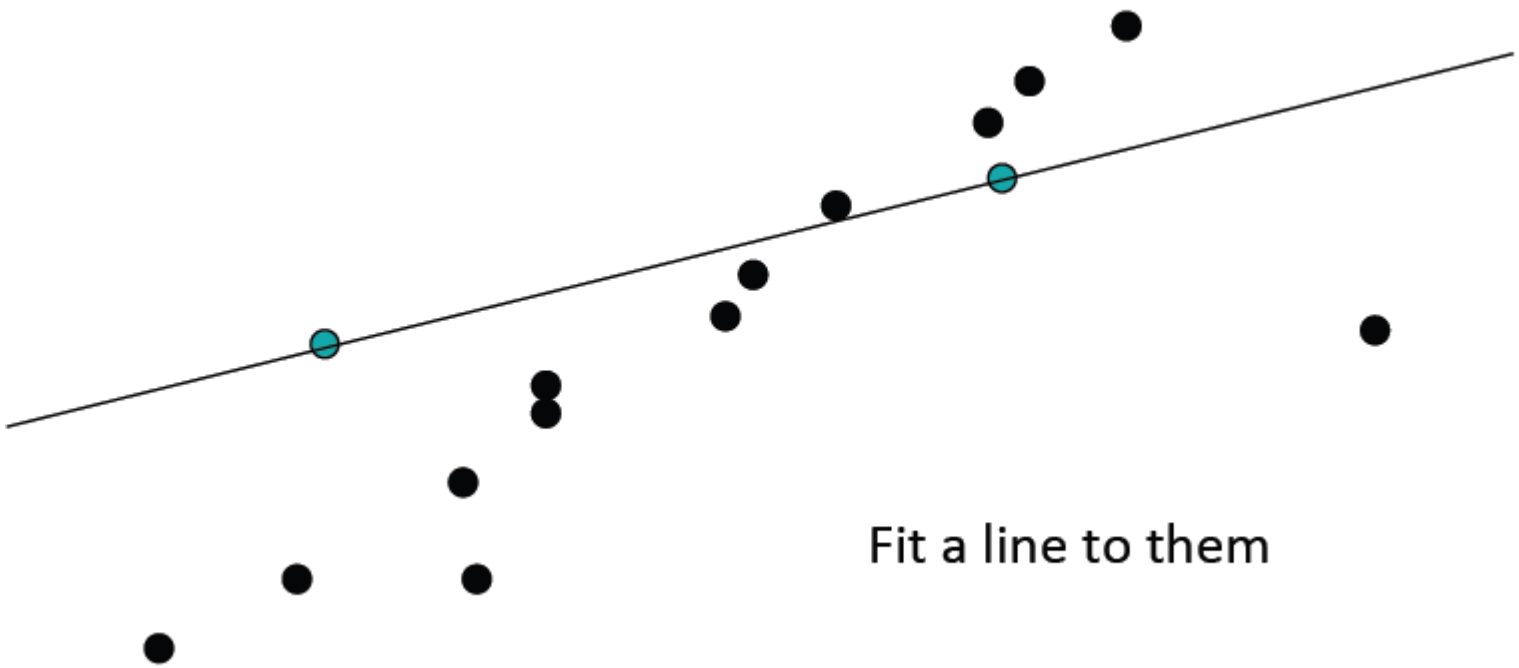
Line fitting with RANSAC

- Step 1:
  Select two points



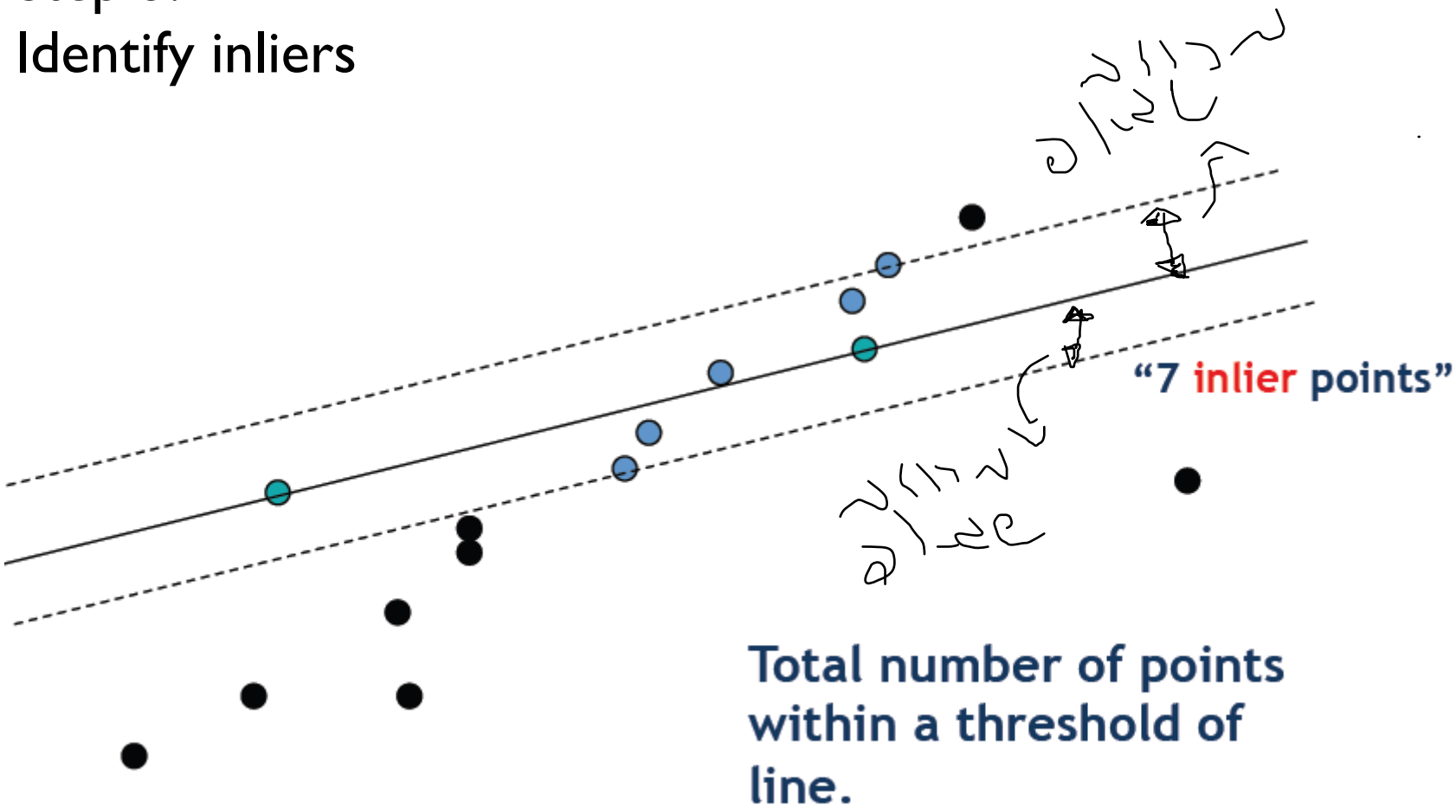
Line fitting with RANSAC

- Step 2:
  Fit a line to the selected points



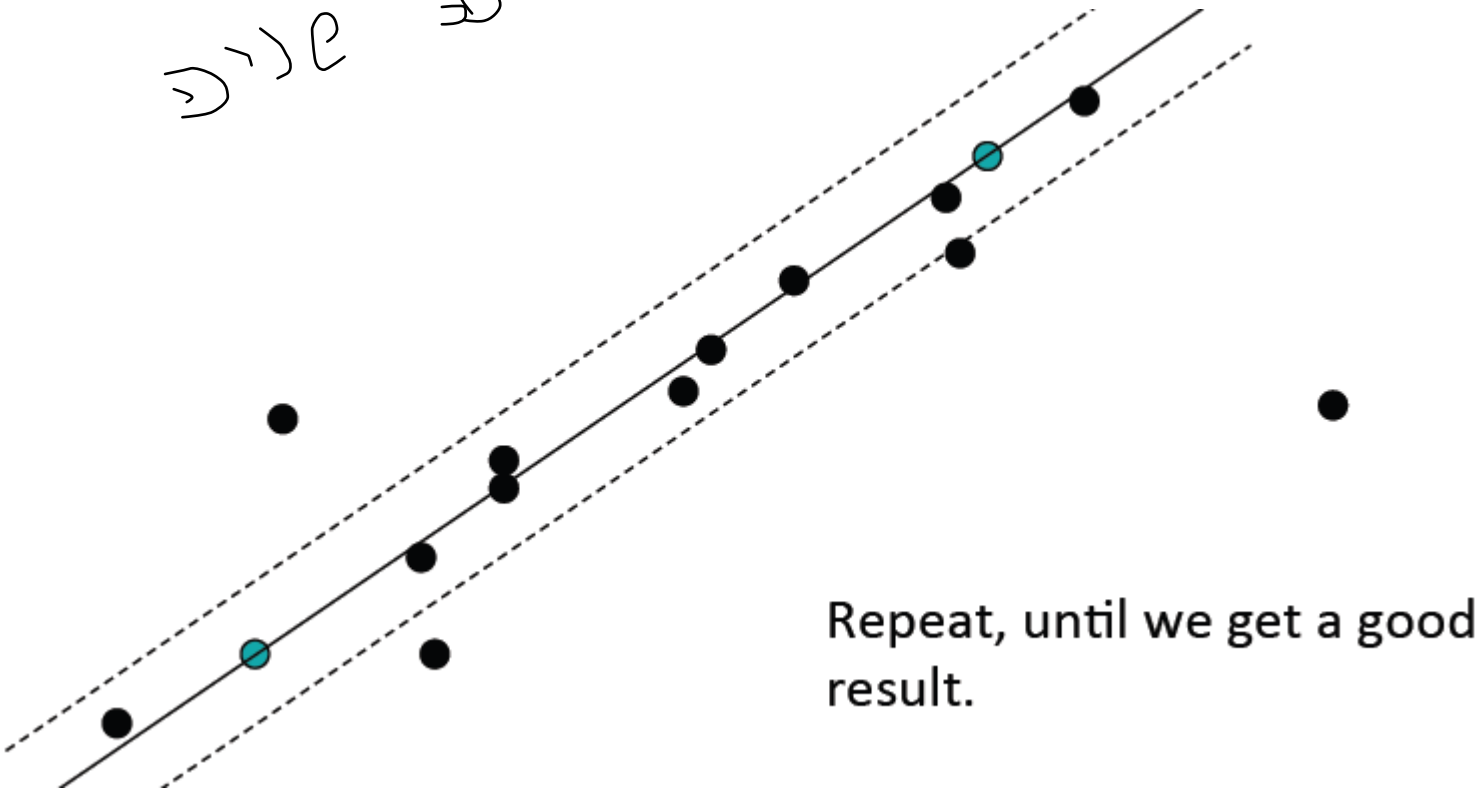
Line fitting with RANSAC

- Step 3:
  Identify inliers



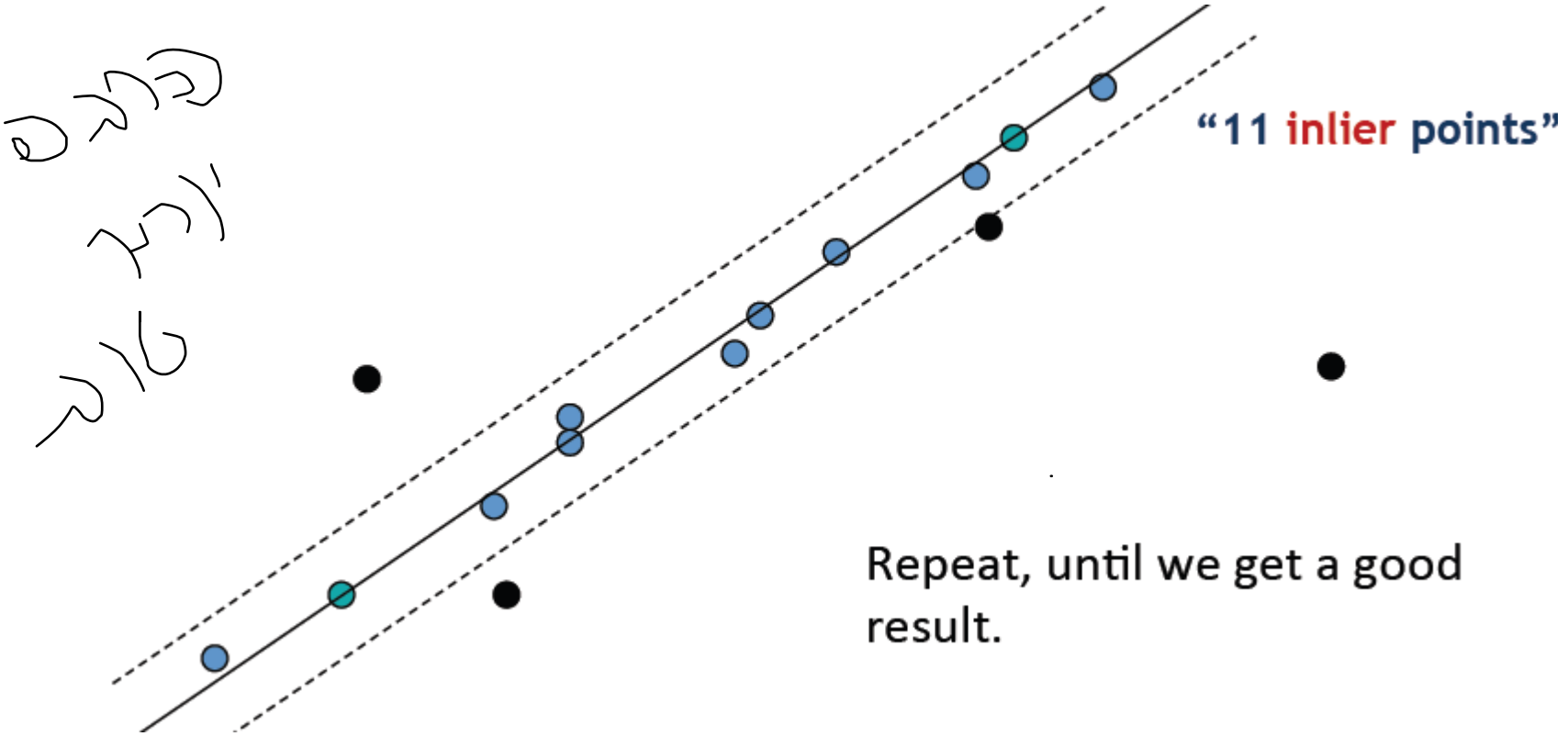
Line fitting with RANSAC

- Step 4:
  Fit line to inliers



Line fitting with RANSAC

- Step 5:
  Count number of new inliers



RANSAC – stopping criteria

- Option 1: when the model is good enough:
  - By number of inliers
  - By it fitting error

מספר הנקודות שצריך כדי לחשב את המודל
במקרה של קו ישר 2

- Option 2: according to probability
  - Let K be the number of iterations
  - Let n be the number of points needed to compute the model
  - Let f be the fraction of inliers of a model
  - Then the probability that a single sample is correct: f^n
  - The probability that all K samples fail is $(1-f^n)^K$
  - Choose K high enough to keep the failure rate low enough

הסתברות של אנלייר בתוך המודל הנכון

אפשר לעצור לפי אחוז מסויים של אנליירים מתוך ס"כ הנקודות

אפשר לעצור לפי השגאה כל אנלייר ואוטלייר יש להם שגאה, ס"כ יש להם שגאה ממוצעת, למשל אם ירדתי בשגאה הממוצעת מתחת לסף מסויים אני יכול לעצור

RANSAC – for multiple models?

- How can we use RANSAC to compute multiple models?

איך לחשב מספר מודלים, למשל לא ידע אם המודל שלי הוא קוו ישר או מעגל, לכן אפשר לנסות
... מקוון שאני לא ידע אז אני צריך להגדיר מספר מודלים שאני מאמין שהם נכונים למשל אקספ, קוו ישר או מעגל

עובדת די טוב, קל למימוש

חסרון, כאשר אחוז האוטלייר מאד גבוהה וזה
TIME CONSUMING

לכן בכל פעם שאני מריץ את ה
RANSAC
אני בוחר במספר הנקודות המנימלי שמתאים לכל המודלים

למשל אם אני רוצה לדעת אם זה קוו או עגול אז אני אבחר 3 נקודות
ואז על 3 נקודות האלה אגדיר מודל עגול ומודל קוו

אחד יהיה המנצח , הנצחון לפי מה שהגדרנו למעלה

הנקודות עם
DATAPOINTS

נשתמש ברנסאק למצוא מודל מתאים בין שתי תמונות שאחת עברה טרנספורמציה לעומת השניה

למשל תמונה שעברה הזהה ביחס לשניה, ואני רוצה לדעת מה הזזה הזאת
למשל אני בוחר מודל הזהה, למשל הזזה של 20 פיקסלים, ומריצים את האלגוריתם

אם הזזה לא 20, ננסה יותר
זה נקבע לפי האנליירים, וגם לפי מה שהגדרנו למעלה

RANSAC - summary

- Pros
  - General method that works well for lots of model fitting problems
  - Easy to implement

- Cons
  - When the percentage of outliers is high too many iterations are needed and failure rate increases

End – finding lines

Now you know how it works