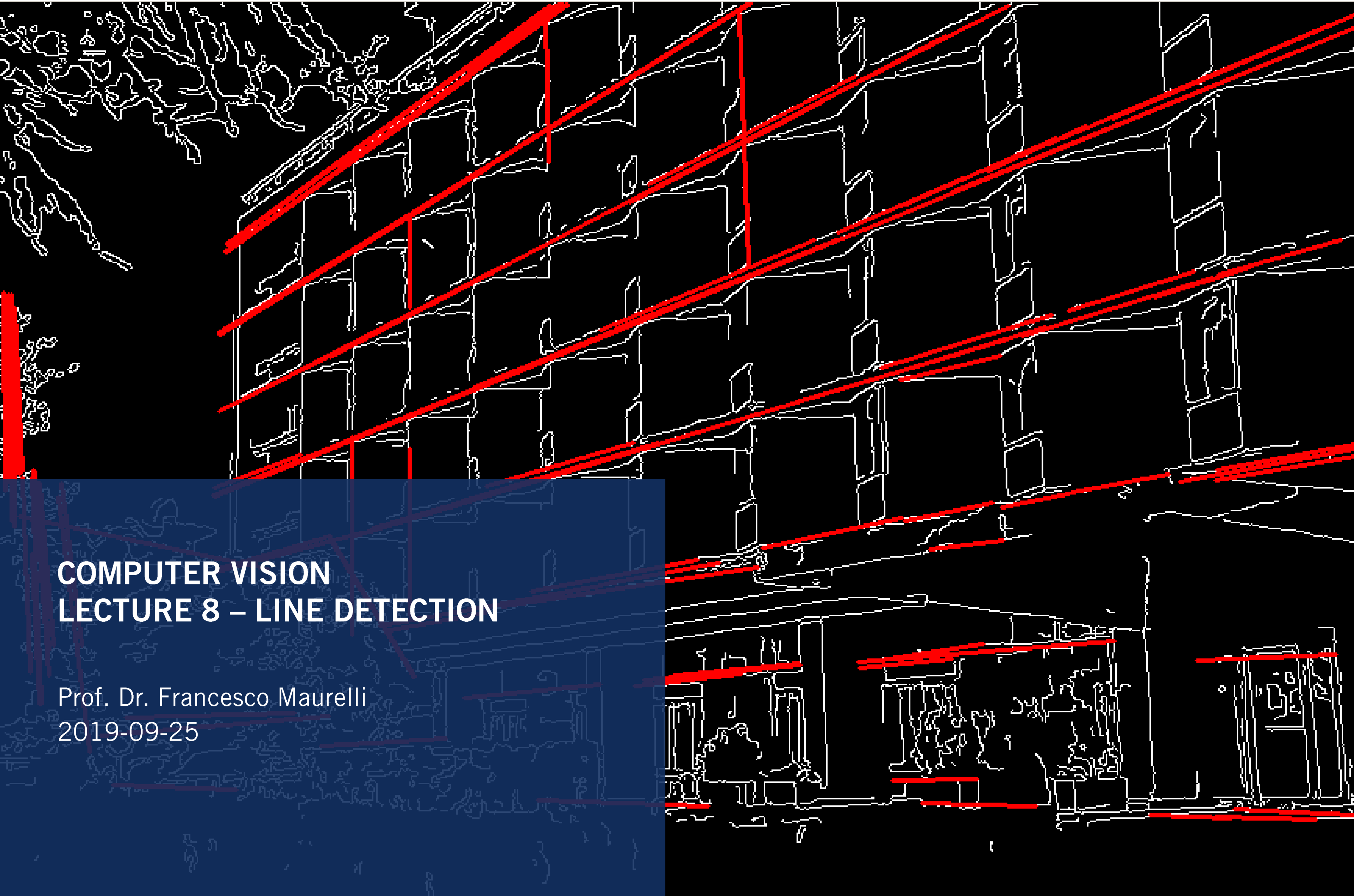


COMPUTER VISION
LECTURE 8 – LINE DETECTION

Prof. Dr. Francesco Maurelli

2019-09-25

WHAT WE WILL LEARN TODAY

1. Hough Transform
2. RANSAC

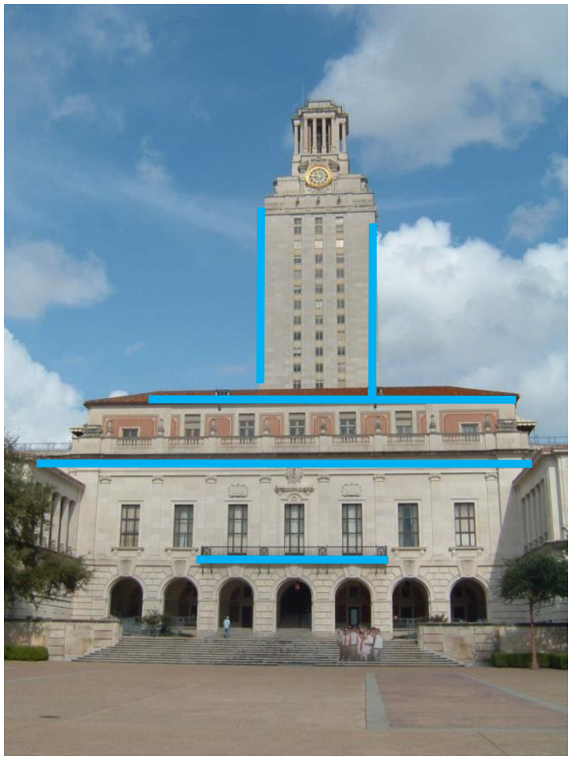
FITTING AS SEARCH IN PARAMETRIC SPACE

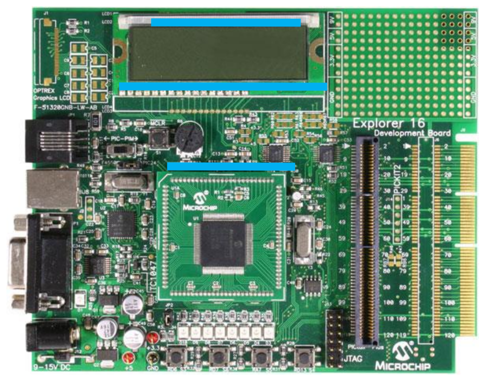
- Choose a parametric model to represent a set of features
- Membership criterion is not local
  - Can't tell whether a point belongs to a given model just by looking at that point.
- Three main questions:
  - What model represents this set of features best?
  - Which of several model instances gets which feature?
  - How many model instances are there?
- Computational complexity is important
  - It is infeasible to examine every possible set of parameters and every possible combination of features

LINE FITTING

Why?

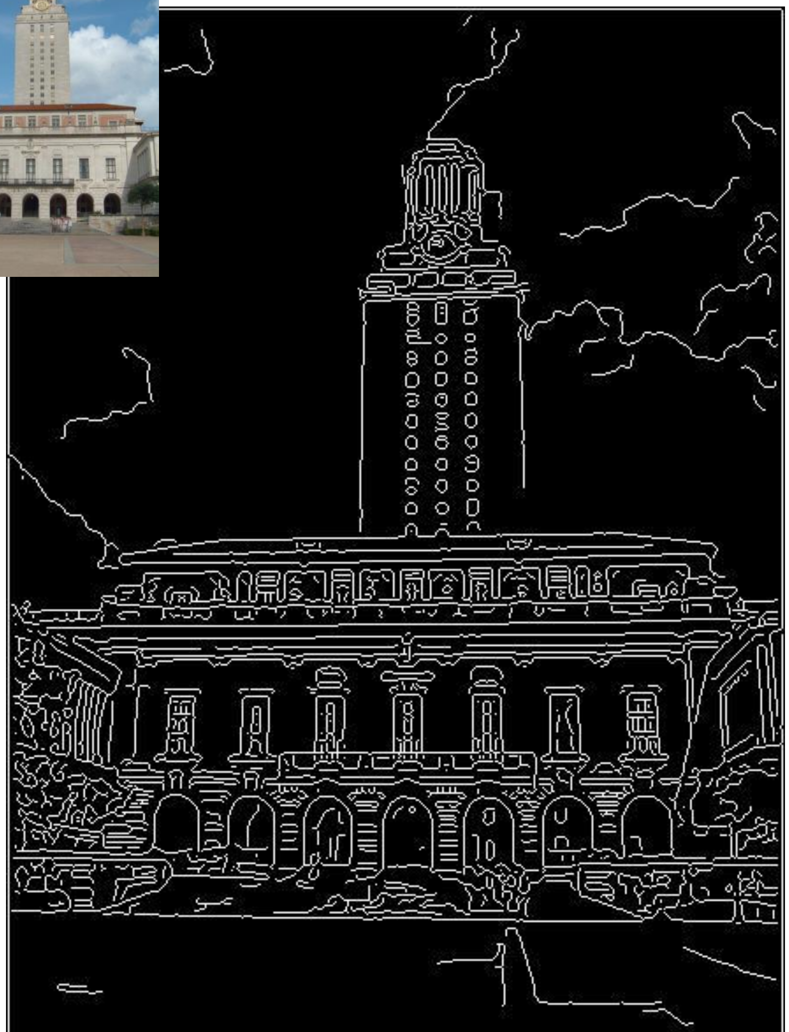
Many objects characterized by presence of straight lines

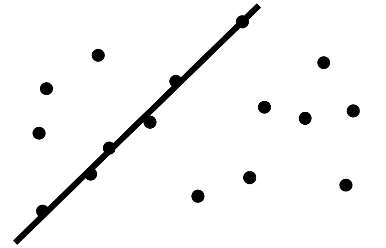




Wait, why aren't we done just by running edge detection?

DIFFICULTY OF LINE FITTING





- 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?

- The Hough transform (HT) can be used to detect lines.
- 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.
- Caveat: Hough transform can detect lines, circles and other structures ONLY if their parametric equation is known.
- It can give robust detection under noise and partial occlusion

PRIOR TO HOUGH TRANSFORM

Assume that we have performed some edge detection, and a thresholding of the edge magnitude image.

Thus, we have some pixels that may partially describe the boundary of some objects.

We wish to find sets of pixels that make up straight lines.

Consider a point of known coordinates $(x_i;y_i)$

- There are many lines passing through the point (x_i ,y_i).

Straight lines that pass that point have the form $y_i= a*x_i + b$

- Common to them is that they satisfy the equation for some set of parameters (a, b)

This equation can obviously be rewritten as follows:

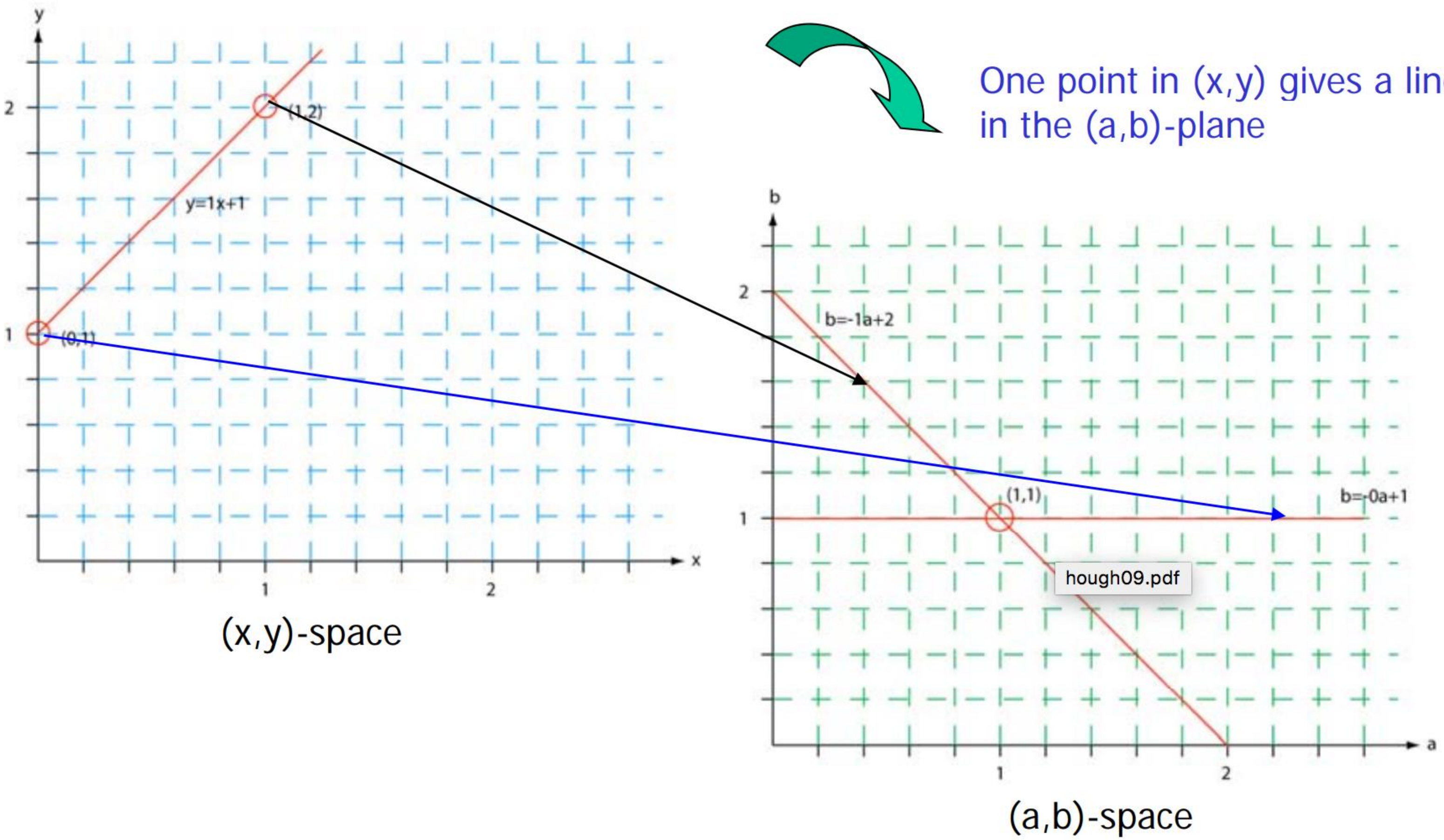
- $b = -a^*x_i + y_i$
- We can now consider x and y as parameters
- a and b as variables.

This is a line in (a, b) space parameterized by x and y.

- So: a single point in x_1,y_1-space gives a line in (a,b) space.
- Another point (x_2, y_2) will give rise to another line (a,b) space.

DETECTING LINES USING HOUGH TRANSFORM

One point in (x,y) gives a line in the (a,b)-plane



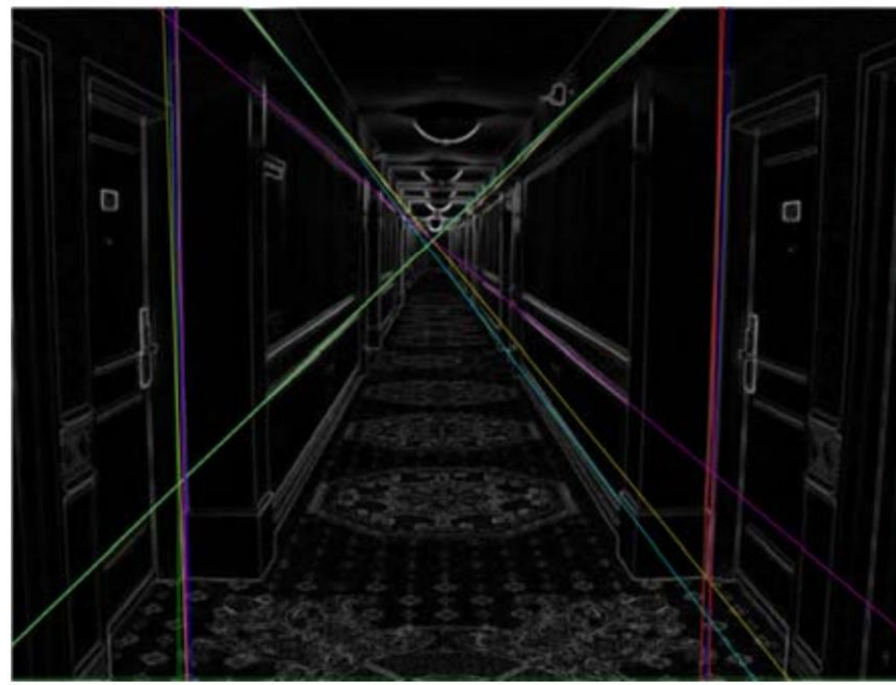
- Two points (x_1, y_1) and$(x_2\ y_2)$ define a line in the (x, y) plane.
- These two points give rise to two different lines in (a,b) space.
- In (a,b) space these lines will intersect in a point (a’ b’)
- All points on the line defined by (x_1, y_1) and $(x_2\ ,\ y_2)$ in (x, y) space will parameterize lines that intersect in (a’, b’) in (a,b) space.

ALGORITHM FOR HOUGH TRANSFORM

1. Quantize the parameter space (a b) by dividing it into cells
2. This quantized space is often referred to as the accumulator cells.
3. Count the number of times a line intersects a given cell.
   - For each pair of points (x_1, y_1) and (x_2, y_2) detected as an edge, find the intersection (a',b') in (a, b)space.
   - Increase the value of a cell in the range $[[a_{min}, a_{max}],[b_{min},b_{max}]]$ that (a', b') belongs to.
   - Cells receiving more than a certain number of counts (also called 'votes') are assumed to correspond to lines in (x,y) space.

OUTPUT OF HOUGH TRANSFORM

1. Here are the top 20 most voted lines in the image:



JACOBS UNIVERSITY

Advantages:

- Conceptually simple.
- Easy implementation
- Handles missing and occluded data very gracefully.
- Can be adapted to many types of forms, not just lines

Disadvantages:

- Computationally complex for objects with many parameters.
- Looks for only one single type of object
- Can be “fooled” by “apparent lines”.
- The length and the position of a line segment cannot be determined.
- Co-linear line segments cannot be separated.

RANSAC [FISCHLER & BOLLES 1981]

1. RANdom SAmple Consensus

2. Approach: we want to avoid the impact of outliers, so let’s look for “inliers”, and use only those.

3. Intuition: if an outlier is chosen to compute the current fit, then the resulting line won’t have much support from rest of the points.

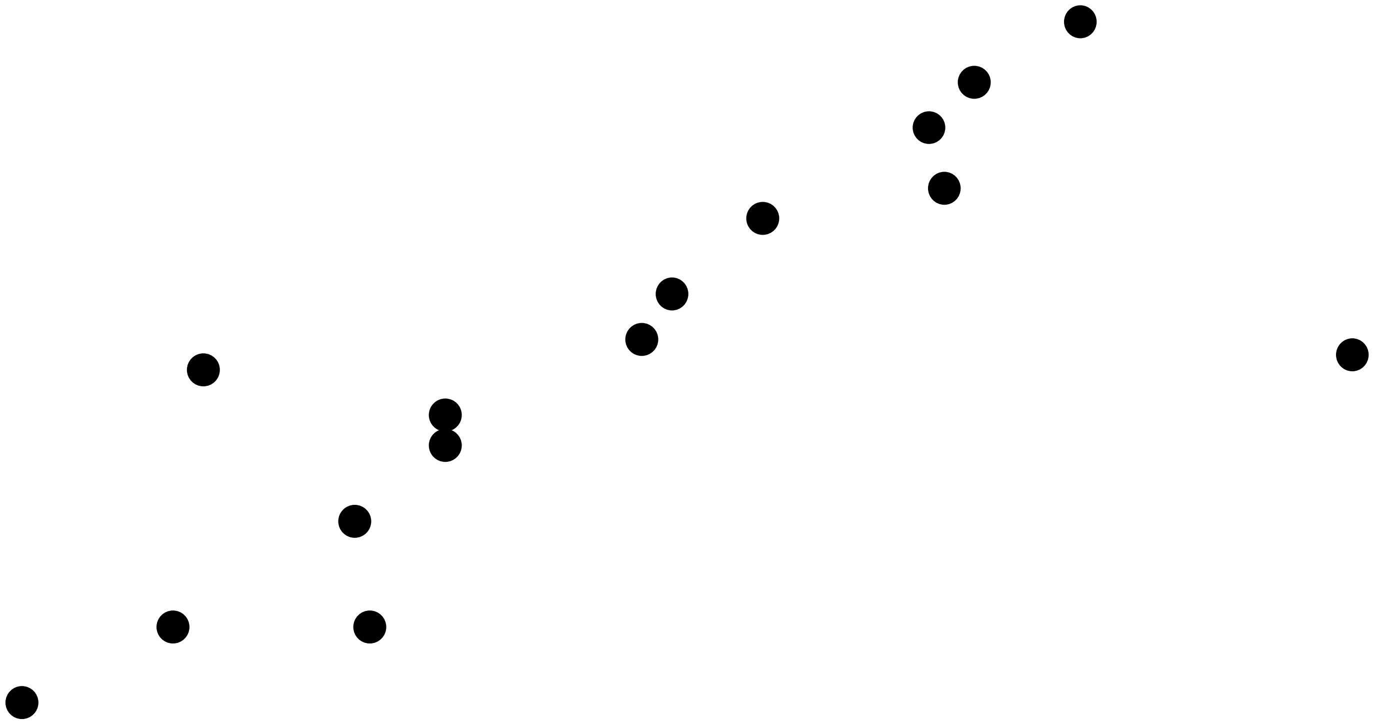
RANSAC [FISCHLER & BOLLES 1981]

RANSAC loop:

1. Randomly select a *seed group* of points
2. Compute transformation from seed group
3. Find *inliers* to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers
5. Keep the transformation with the largest number of inliers

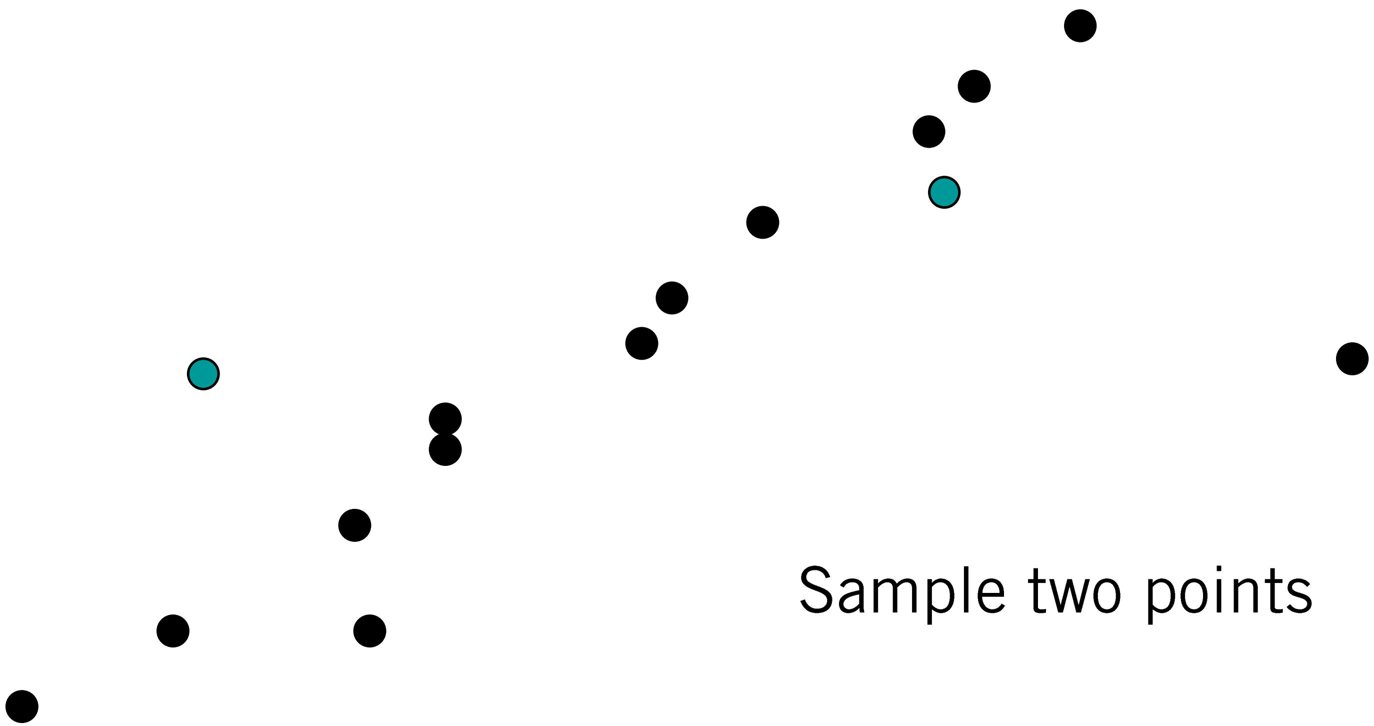
RANSAC LINE FITTING EXAMPLE

1. Task: Estimate the best line
   - *How many points do we need to estimate the line?*



RANSAC LINE FITTING EXAMPLE

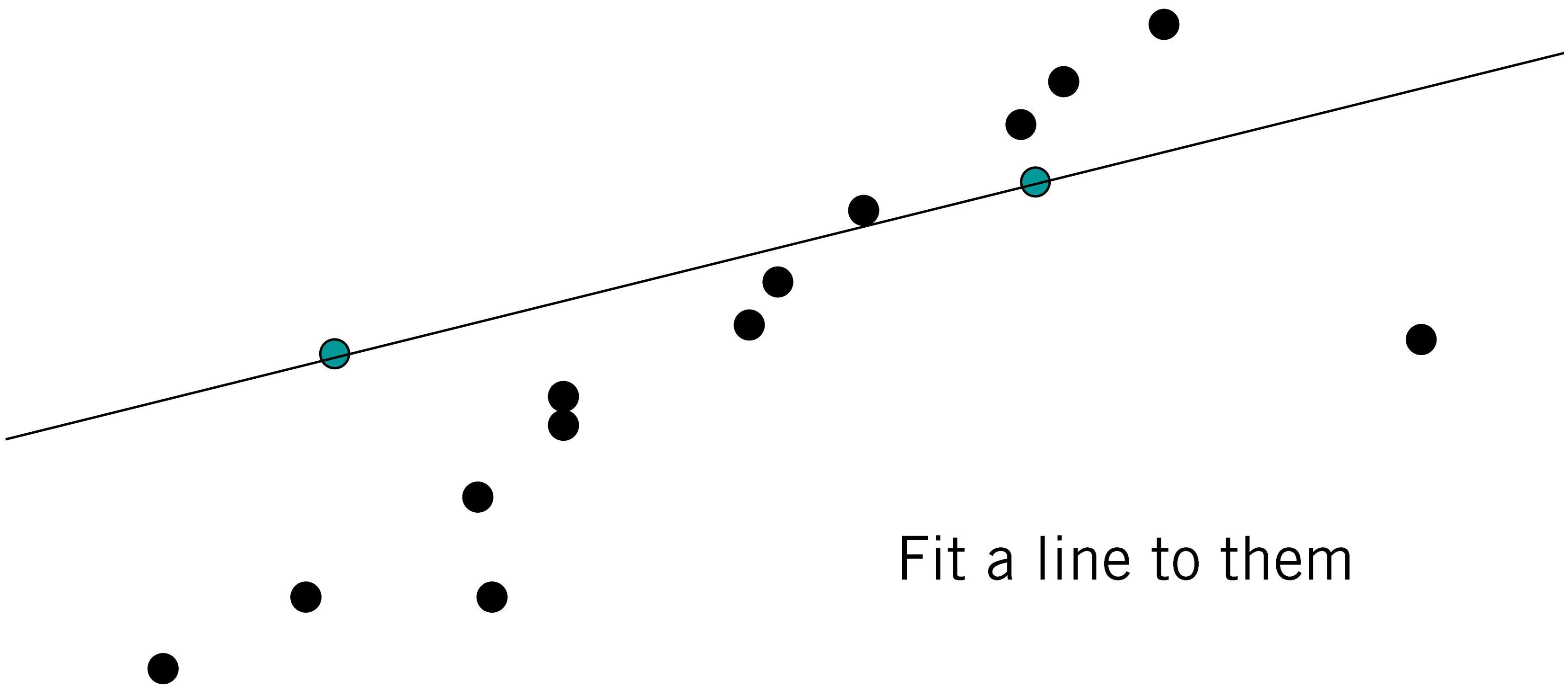
1. Task: Estimate the best line



Sample two points

RANSAC LINE FITTING EXAMPLE

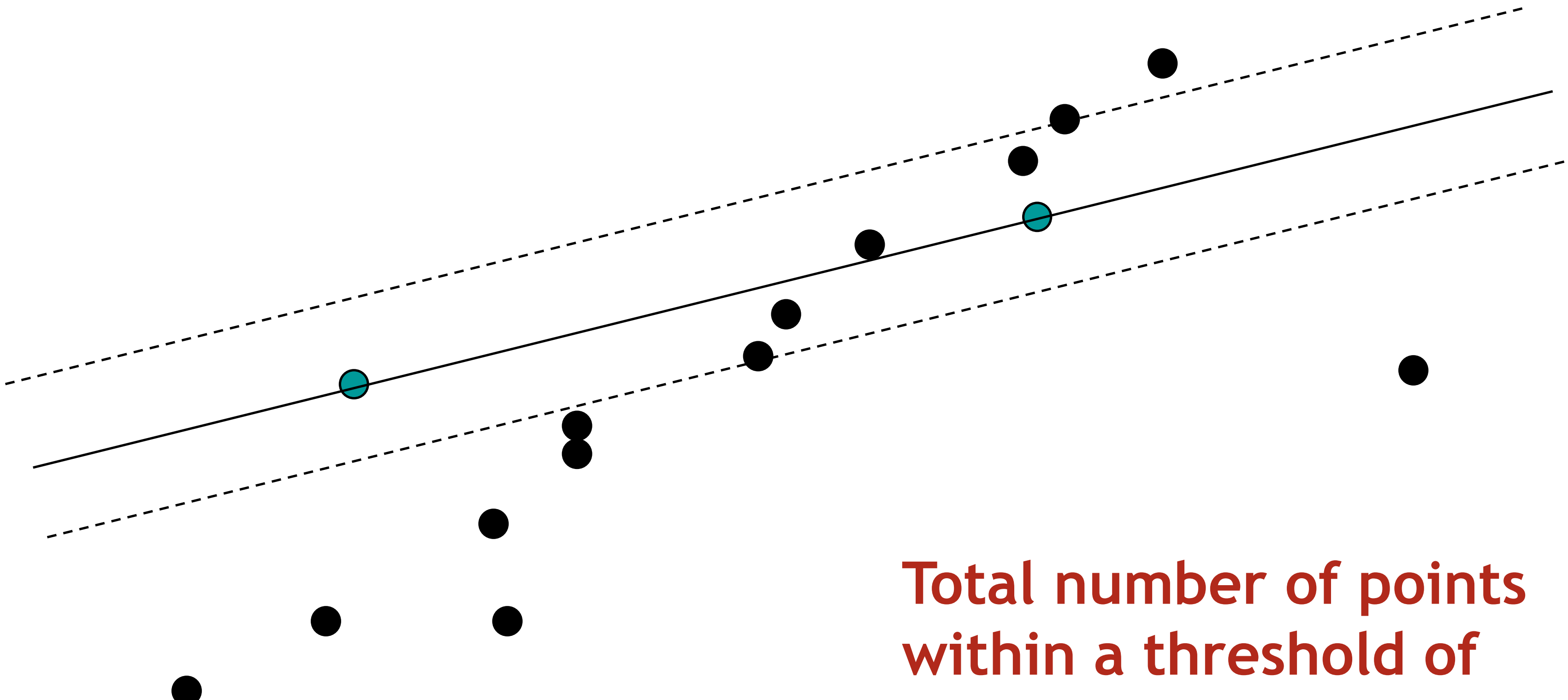
1. Task: Estimate the best line



Fit a line to them

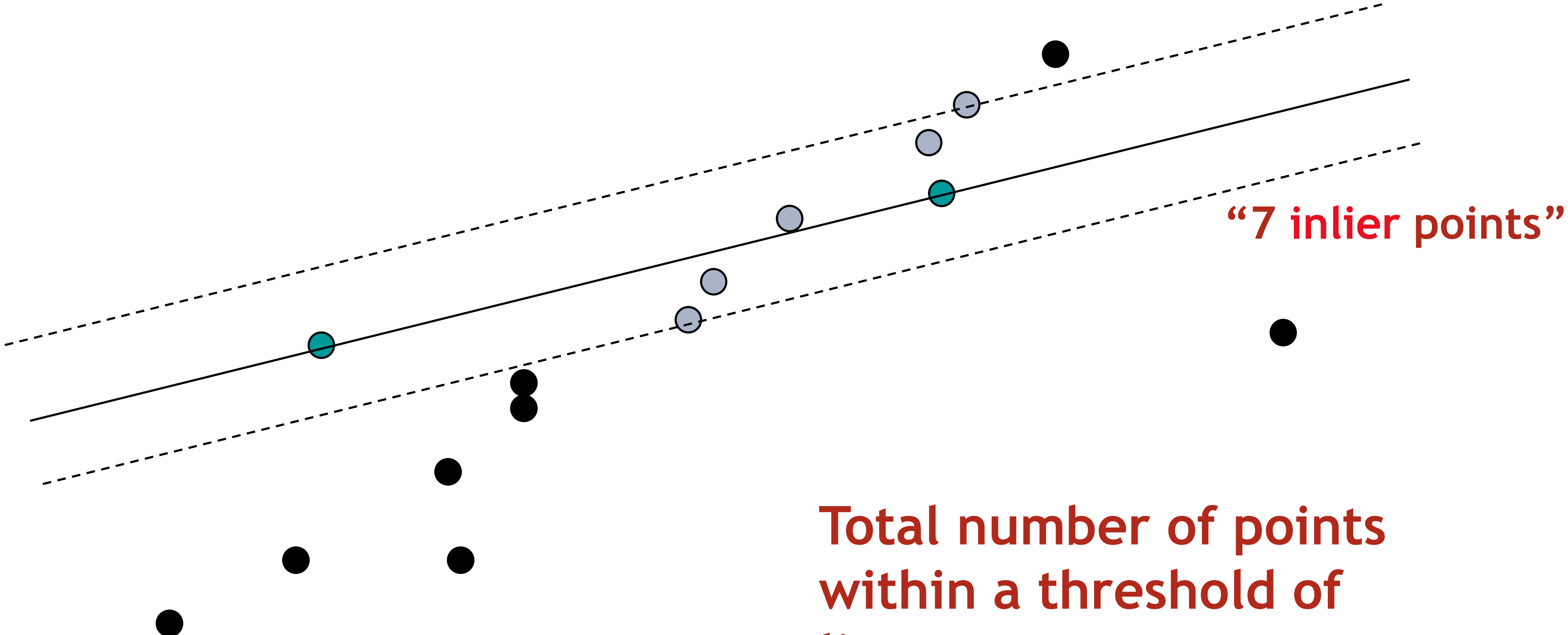
RANSAC LINE FITTING EXAMPLE

1. Task: Estimate the best line



Total number of points within a threshold of line.

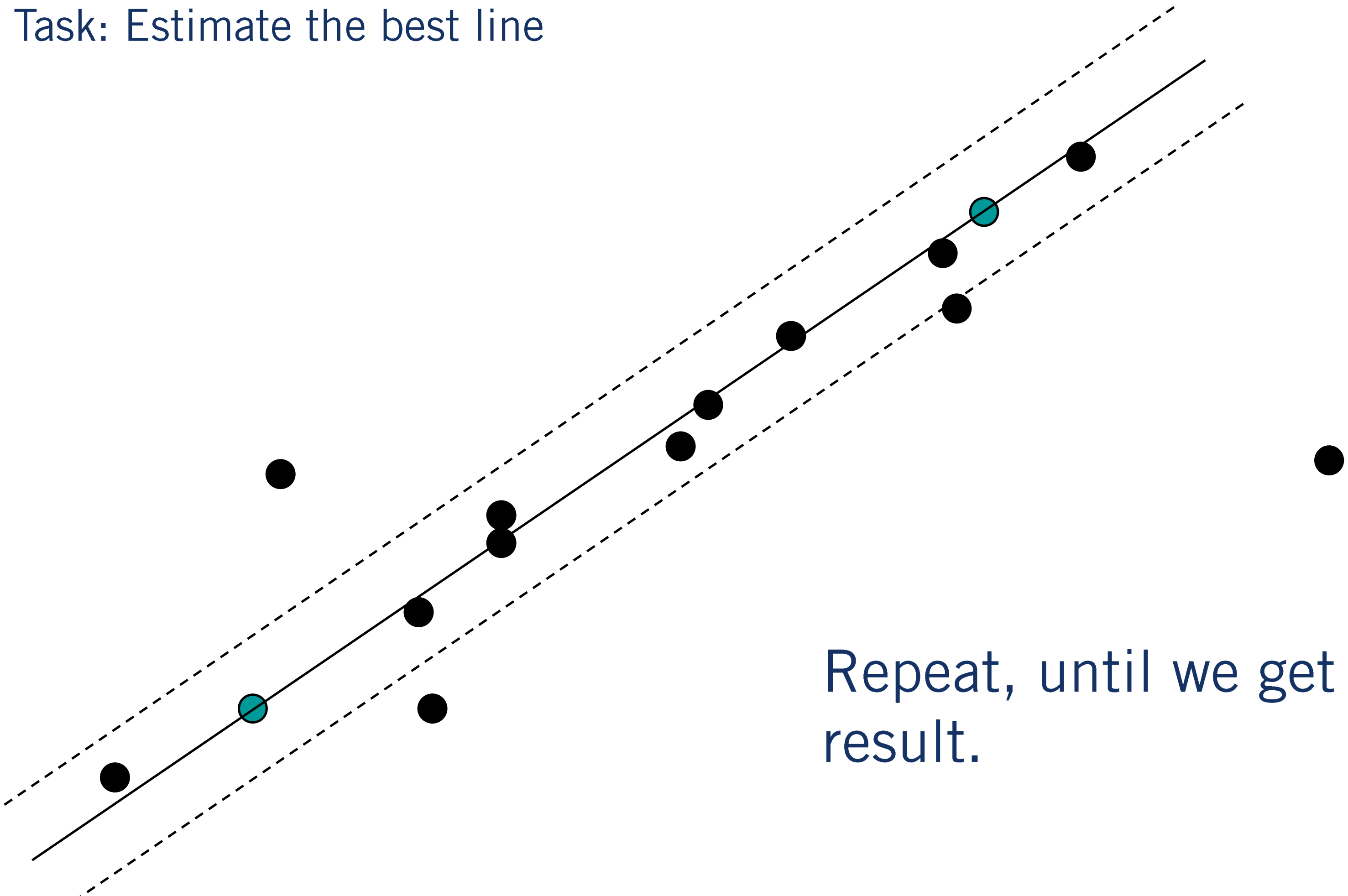
1. Task: Estimate the best line



Total number of points within a threshold of line.

RANSAC LINE FITTING EXAMPLE

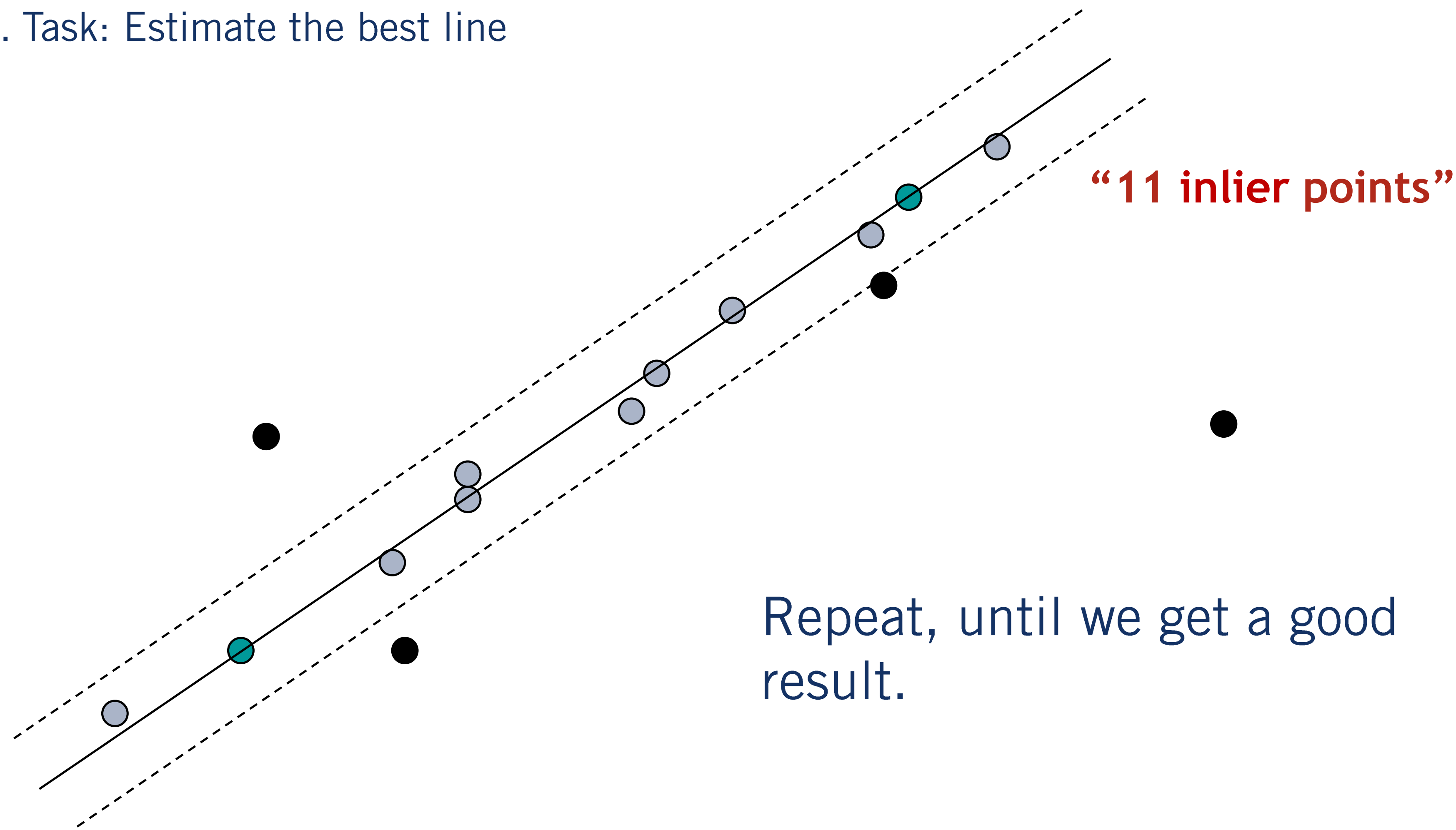
1. Task: Estimate the best line



Repeat, until we get a good result.

RANSAC LINE FITTING EXAMPLE

1. Task: Estimate the best line



Repeat, until we get a good result.

Algorithm 15.4: RANSAC: fitting lines using random sample consensus

Determine:
    n — the smallest number of points required
    k — the number of iterations required
    t — the threshold used to identify a point that fits well
    d — the number of nearby points required
        to assert a model fits well
Until k iterations have occurred
    Draw a sample of n points from the data
        uniformly and at random
    Fit to that set of n points
    For each data point outside the sample
        Test the distance from the point to the line
            against t; if the distance from the point to the line
            is less than t, the point is close
    end
    If there are d or more points close to the line
        then there is a good fit. Refit the line using all
        these points.
end
Use the best fit from this collection, using the
    fitting error as a criterion

How many samples are needed?

- Suppose w is fraction of inliers (points from line).
- n points needed to define hypothesis (2 for lines)
- k samples chosen.

Prob. that a single sample of n points is correct: w^n

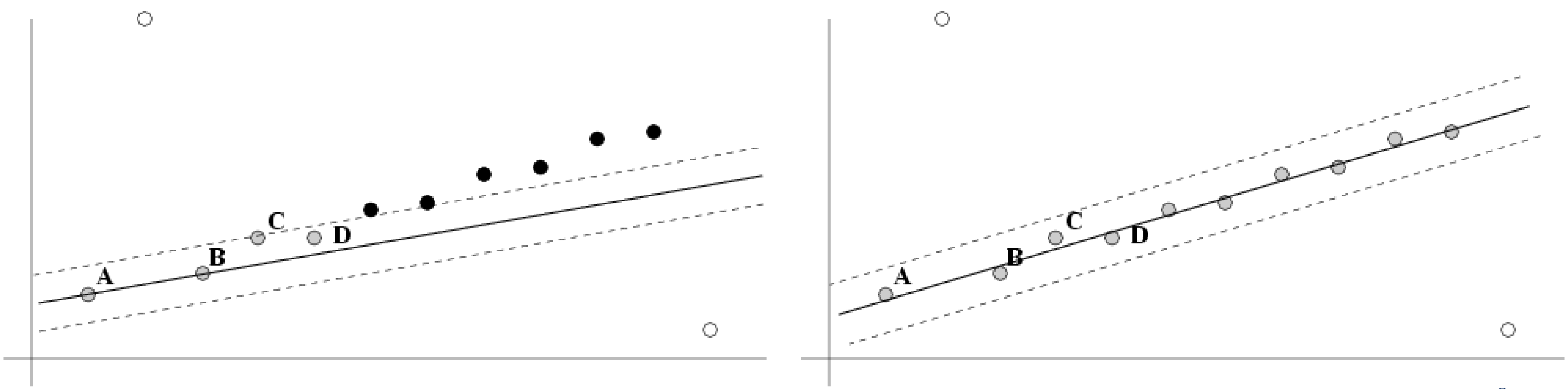
Prob. that all k samples fail is: $(1-w^n)^k$

$\Rightarrow$ Choose k high enough to keep this below desired failure rate.

RANSAC: COMPUTED K (P=0.99)

Sample size n	Proportion of outliers						
	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

1. RANSAC divides data into inliers and outliers and yields estimate computed from minimal set of inliers.
2. Improve this initial estimate with estimation over all inliers (e.g. with standard least-squares minimization).
3. But this may change inliers, so alternate fitting with re-classification as inlier/outlier.



- **Pros**:
  - General method suited for a wide range of model fitting problems
  - Easy to implement and easy to calculate its failure rate
- **Cons**:
  - Only handles a moderate percentage of outliers without cost blowing up
  - Many real problems have high rate of outliers (but sometimes selective choice of random subsets can help)
- A voting strategy, The Hough transform, can handle high percentage of outliers