Robust Fitting

Mathematical Models and Methods for Image Processing

Giacomo Boracchi

https://boracchi.faculty.polimi.it/

May 29th 2025

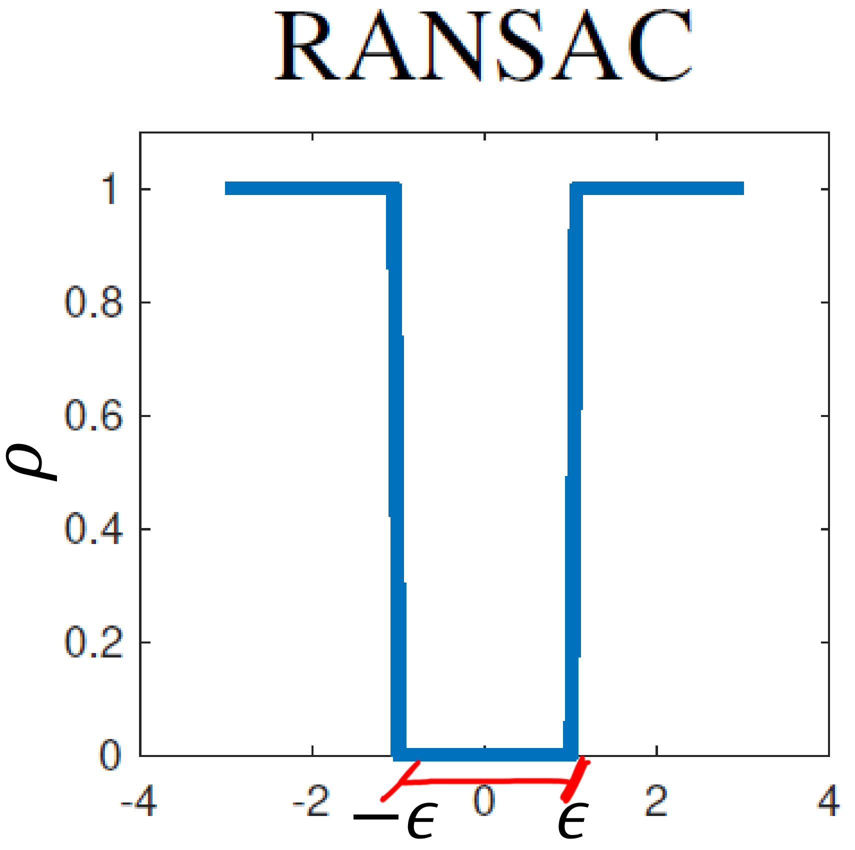
Ransac as M-estimator

(Steward 1999) RanSaC can be seen as a particular M-estimator since the **loss it minimizes** is the number of points having residual above the inlier threshold ϵ

$$f(r_i) = \begin{cases} 1, & r_i > \epsilon \\ 0, & r_i \leq \epsilon \end{cases}$$

Of course selecting inlier thrshold ϵ is very critical

Ransac achieves a theoretical breakdown of 50% of outliers, but in practice, provided a good selection of ϵ, this can be even higher

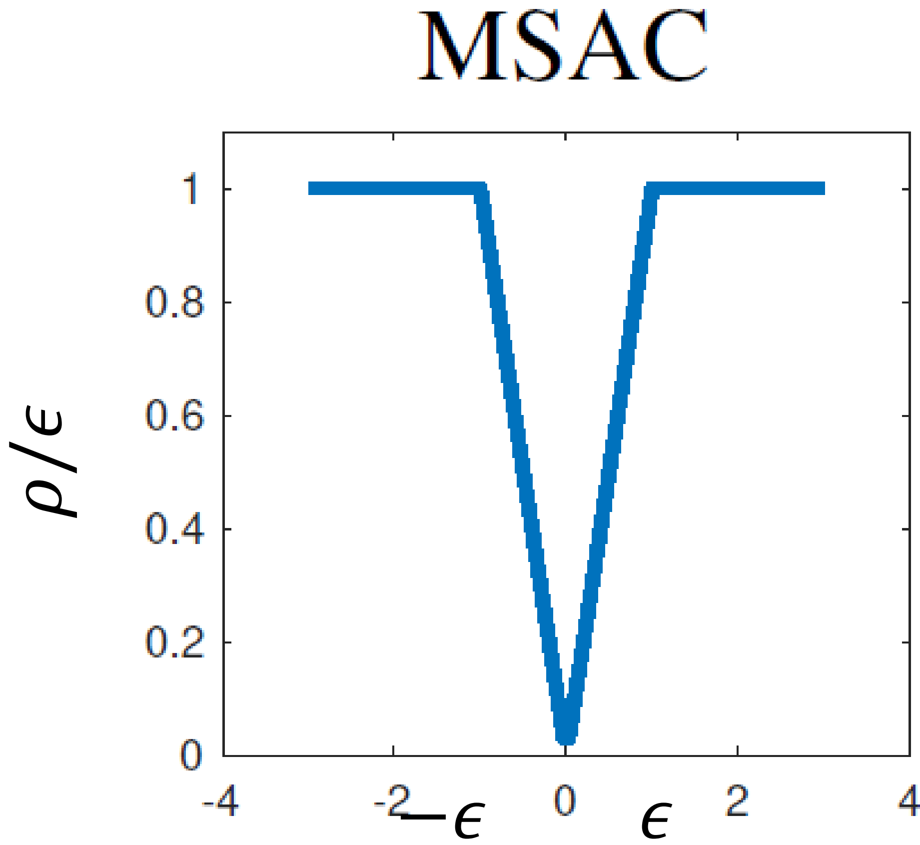


MSAC

(Torr and Zisserman 2000) a different loss function to be minimized within the RanSaC framework

$$f(r_i) = \begin{cases} \epsilon, & |r_i| > \epsilon \\ |r_i|, & |r_i| \leq \epsilon \end{cases}$$

This turns to be more effective and should be preferred to RanSaC



Ransac vs MSaC

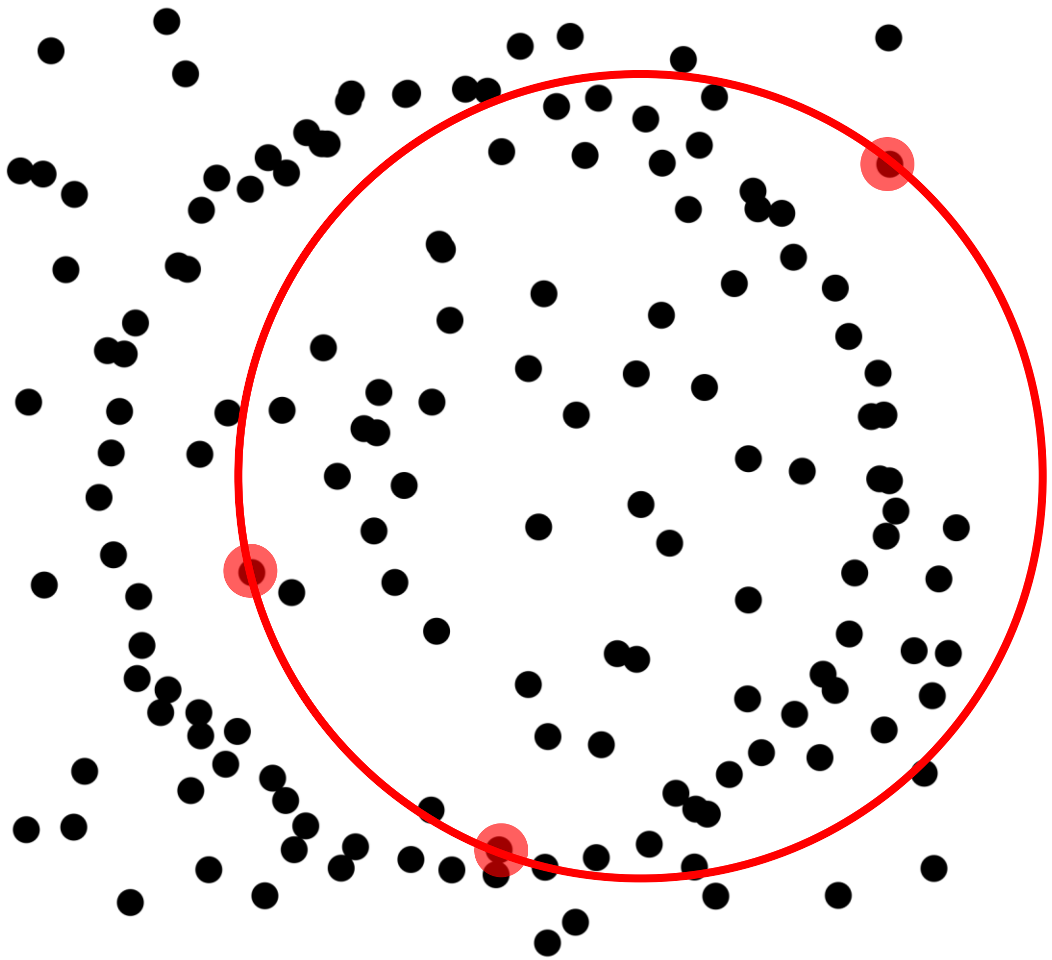
Input: X data, ϵ inlier threshold, k_{max} max iteration
Output: θ^* model estimate
$J^* = -\infty, k = 0;$
repeat
  Select randomly a minimal sample set $S \subset X$;
  Estimate parameters θ on S;
  Evaluate $J(\theta) = \sum_{x \in X} \hat{f}_\epsilon(r(x, \theta));$
  if $J(\theta) > J^*$ **then**
    $\theta^* = \theta;$
    $J^* = J(\theta);$
  end
  $k = k + 1;$
until $k > k_{max}$;
Optimize θ^* on its inliers.

Input: X data, ϵ inlier threshold, k_{max} max iteration
Output: θ^* model estimate
$J^* = +\infty, k = 0;$
repeat
  Select randomly a minimal sample set $S \subset X$;
  Estimate parameters θ on S;
  Estimate inlier set $I = \{x \in X : r(x, \theta)^2 < \epsilon^2\};$
  Evaluate $J(\theta) = \sum_{x \in I} r(x, \theta) + (|X| - |I|)\epsilon;$
  if $J(\theta) < J^*$ **then**
    $\theta^* = \theta;$
    $J^* = J(\theta);$
  end
  $k = k + 1;$
until $k > k_{max}$;
Optimize θ^* on its inliers.

Credits Luca Magri

Least Median of Squares

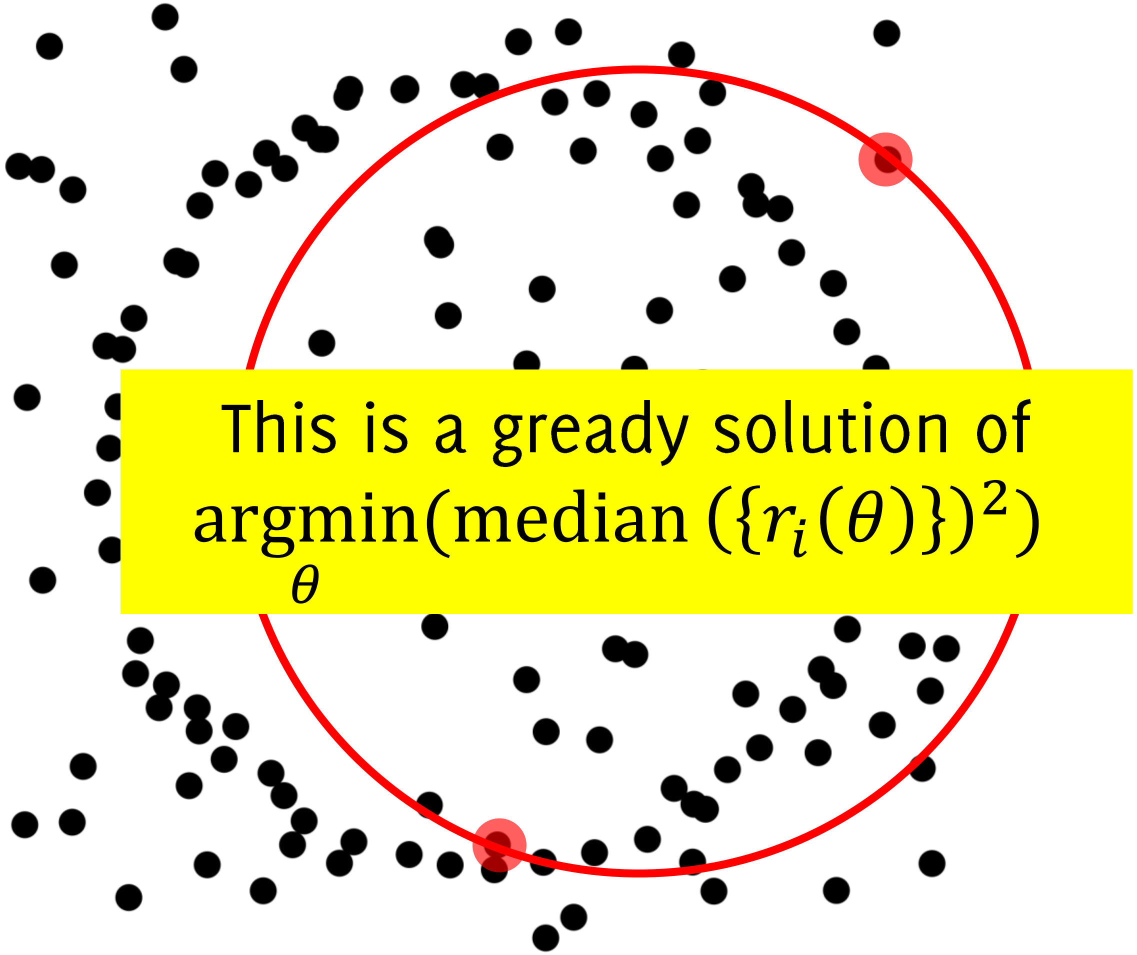
L-meds: Least Median of Squares, Rousseeuw e Leroy (1987)



Input: X data, k_{max} max iteration
Output: θ^* model estimate
$J^* = +\infty, k = 0;$
repeat
  Select randomly a minimal sample set $S \subset X$;
  Estimate parameters θ on S;
  Evaluate $J(\theta) = \text{median}_{x \in X}(r(x, \theta))$;
  if $J(\theta) < J^*$ **then**
    $\theta^* = \theta;$
    $J^* = J(\theta);$
  end
  $k = k + 1;$
until $k > k_{max}$;
Optimize θ^* on its inliers.

Credits Luca Magri,

L-meds: Least Median of Squares, Rousseeuw e Leroy (1987)



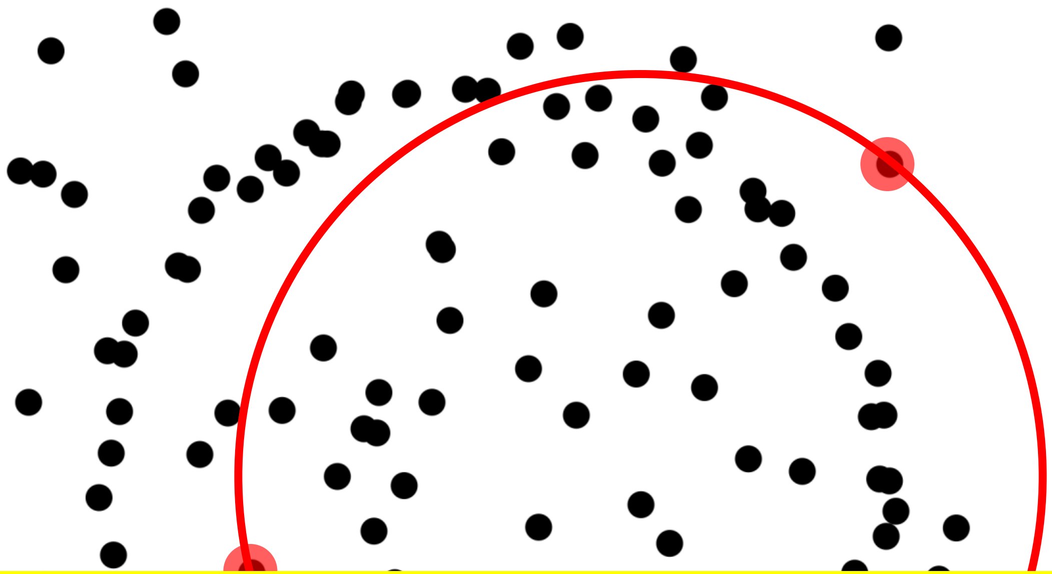
This is a gready solution of

$$\underset{\theta}{\operatorname{argmin}}(\operatorname{median}(\{r_i(\theta)\})^2)$$

Input: X data, k_{max} max iteration
Output: θ^* model estimate
$J^* = +\infty, k = 0;$
repeat
  Select randomly a minimal sample set $S \subset X$;
  Estimate parameters θ on S;
  Evaluate $J(\theta) = \text{median}_{x \in X}(r(x, \theta))$;
  if $J(\theta) < J^*$ **then**
    $\theta^* = \theta$;
    $J^* = J(\theta)$;
  end
  $k = k + 1$;
until $k > k_{max}$;
Optimize θ^* on its inliers.

Credits Luca Magri,

L-meds: Least Median of Squares, Rousseeuw e Leroy (1987)

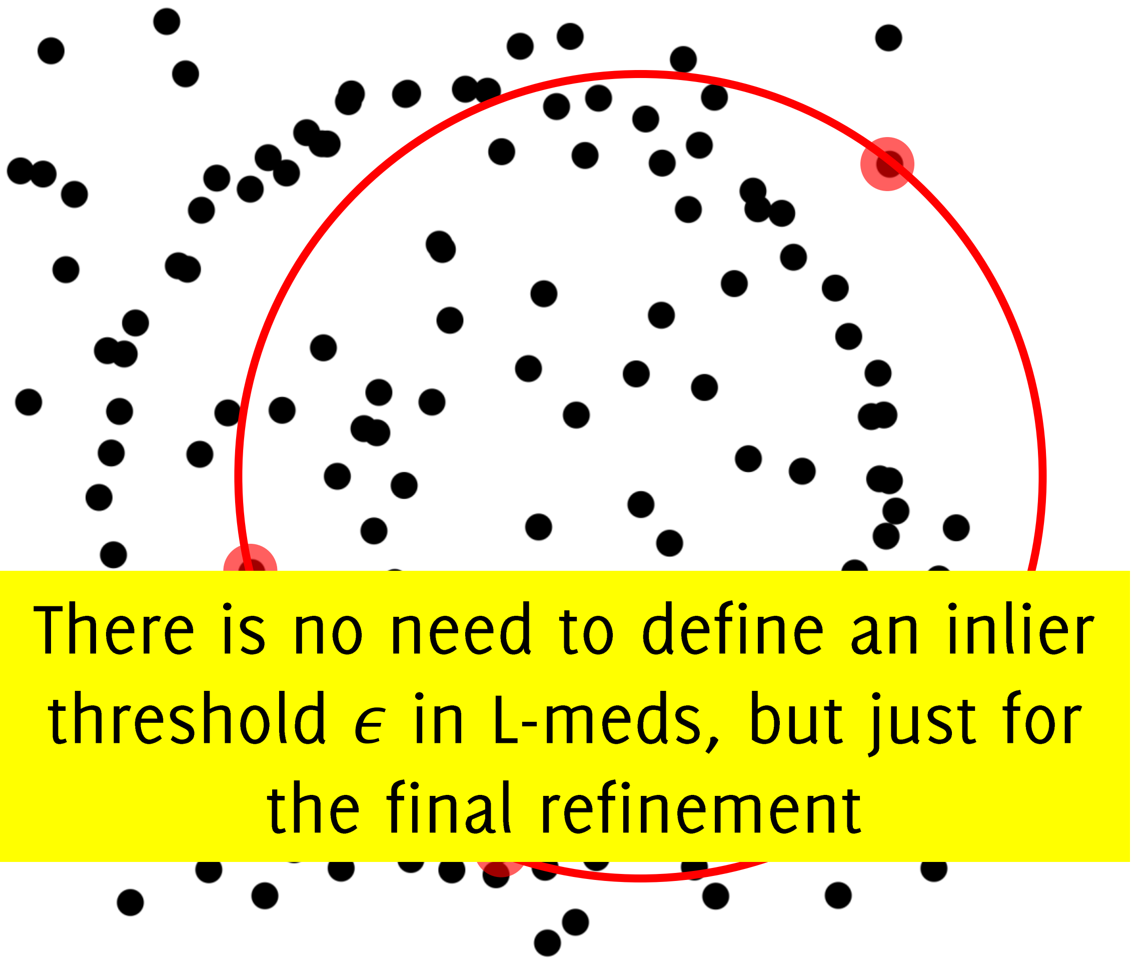


Since there is no explicit definition of inliers here, inliers can be identified as points having residuals (w.r.t. to the final model) that are smaller than 2.5σ

Input: X data, k_{max} max iteration
Output: θ^* model estimate
$J^* = +\infty, k = 0;$
repeat
  Select randomly a minimal sample set $S \subset X$;
  Estimate parameters θ on S;
  Evaluate $J(\theta) = \text{median}_{x \in X}(r(x, \theta))$;
  if $J(\theta) < J^*$ **then**
    $\theta^* = \theta$;
    $J^* = J(\theta)$;
  end
  $k = k + 1$;
until $k > k_{max}$;
Optimize θ^* on its inliers.

Credits Luca Magri,

L-meds: Least Median of Squares, Rousseeuw e Leroy (1987)



There is no need to define an inlier threshold ϵ in L-meds, but just for the final refinement

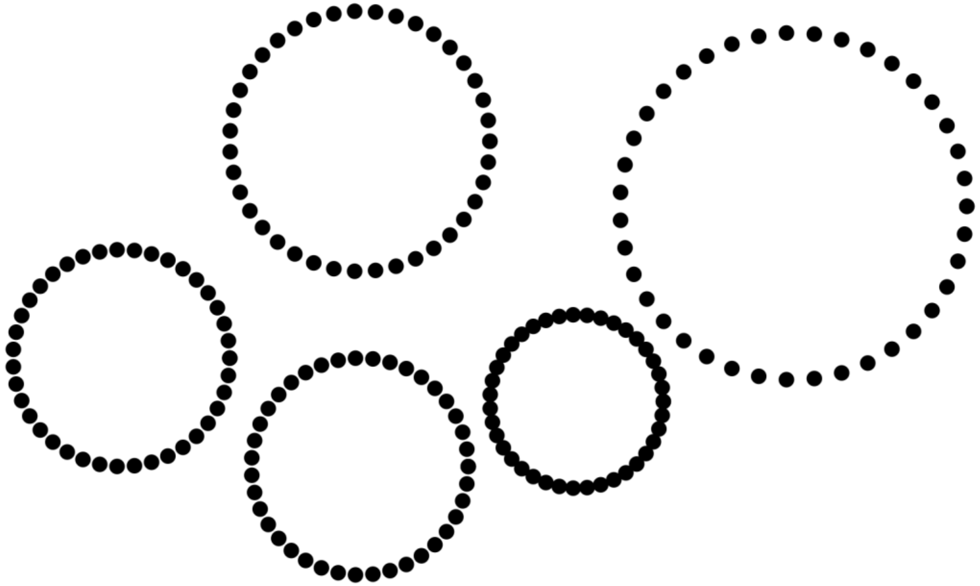
Input: X data, k_{max} max iteration
Output: θ^* model estimate
$J^* = +\infty, k = 0;$
repeat
  Select randomly a minimal sample set $S \subset X$;
  Estimate parameters θ on S;
  Evaluate $J(\theta) = \text{median}_{x \in X}(r(x, \theta))$;
  if $J(\theta) < J^*$ **then**
    $\theta^* = \theta$;
    $J^* = J(\theta)$;
  end
  $k = k + 1$;
until $k > k_{max}$;
Optimize θ^* on its inliers.

Credits Luca Magri,

The problem of fitting multiple
geometric primitives
is ubiquitous in Computer Vision

The problem of multi-model fitting (or structure recovery)

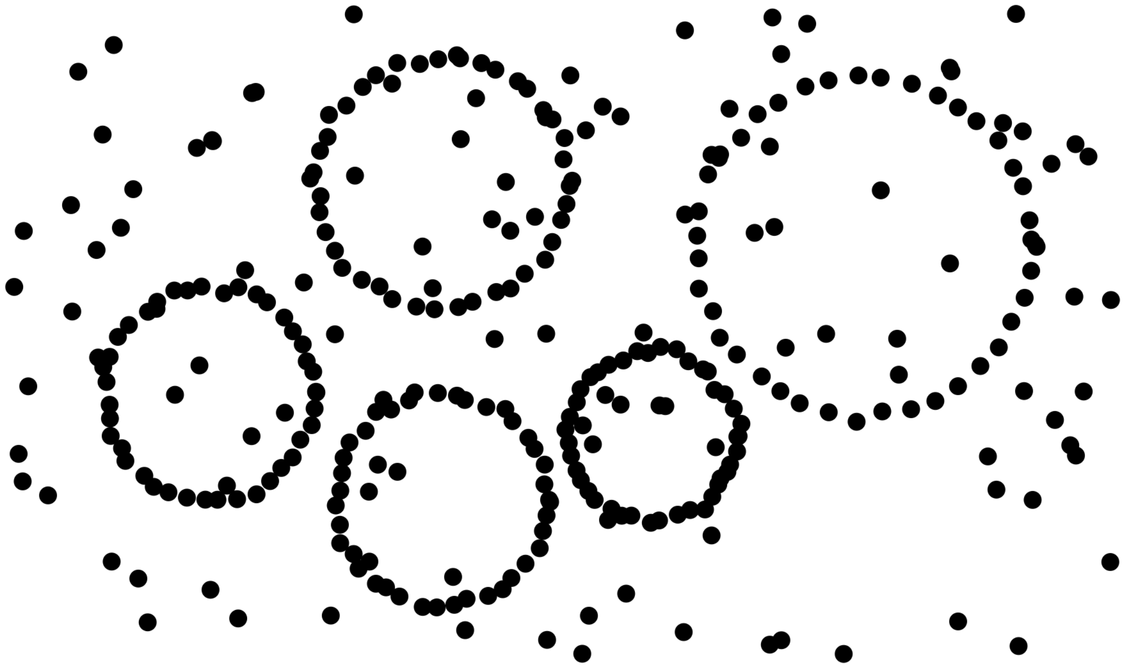
Given a set of data $X = \{x_1, \dots, x_N\} \subset \mathbb{R}^d$, possibly corrupted by noise and outliers, and a family of geometric models Θ



Credits Luca Magri

The problem of multi-model fitting (or structure recovery)

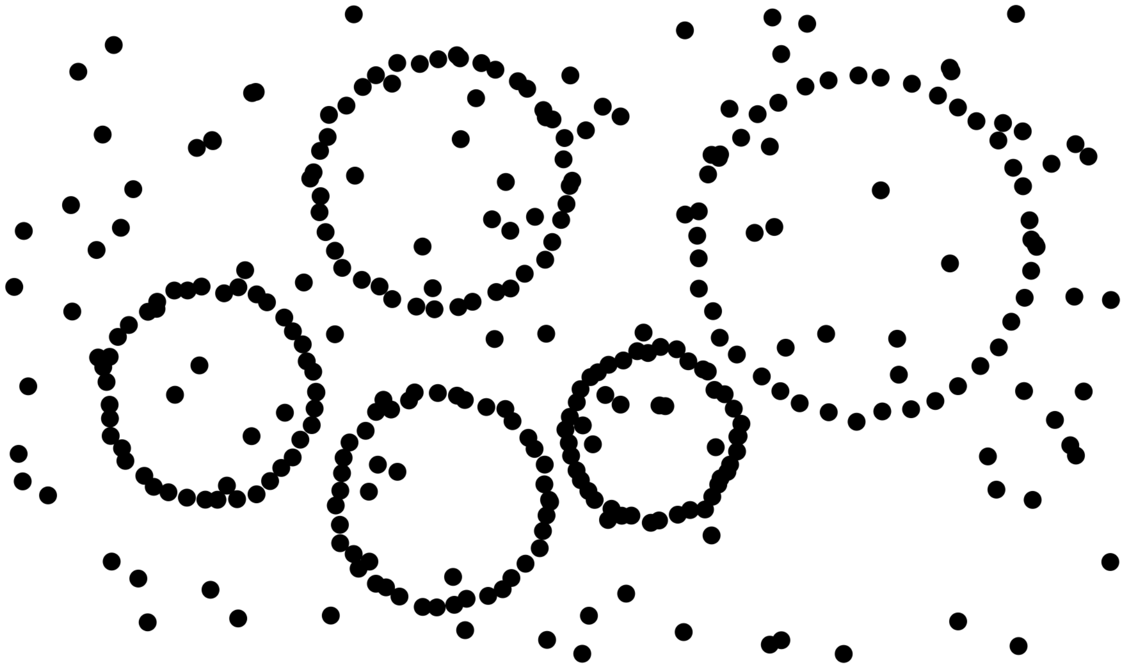
Given a set of data $X = \{x_1, \dots, x_N\} \subset \mathbb{R}^d$, possibly corrupted by noise and outliers, and a family of geometric models Θ



Credits Luca Magri

The problem of multi-model fitting (or structure recovery)

Given a set of data $X = \{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, possibly corrupted by noise and outliers, and a family of geometric models Θ



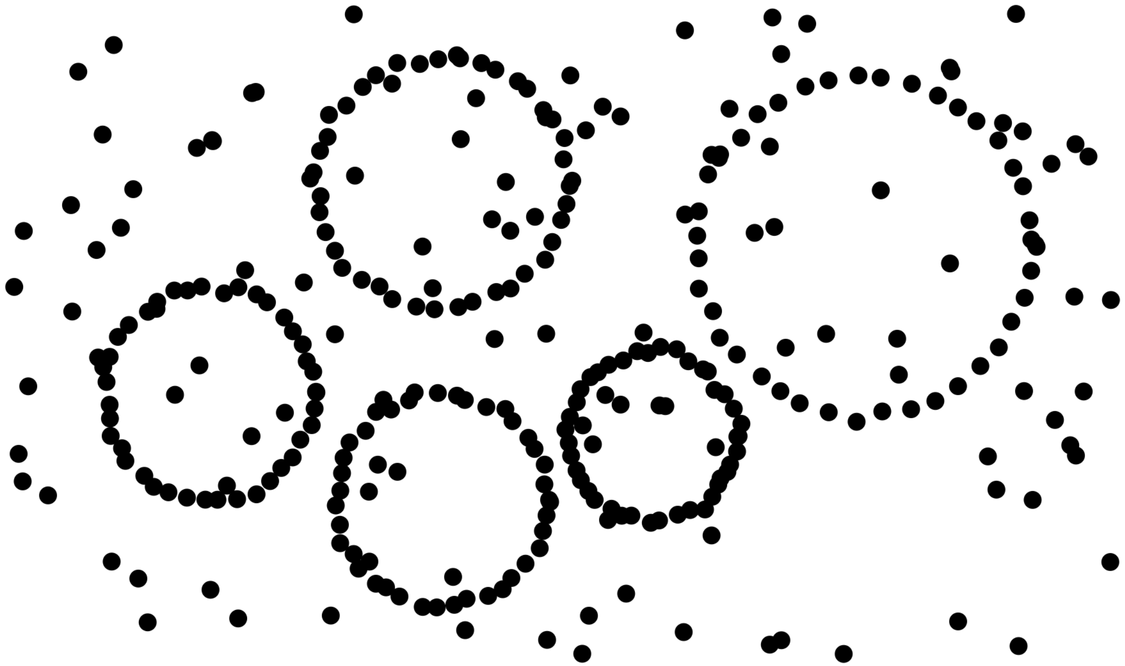
Goal: automatically estimate the models that best explain the data/discover the structures hidden in the data

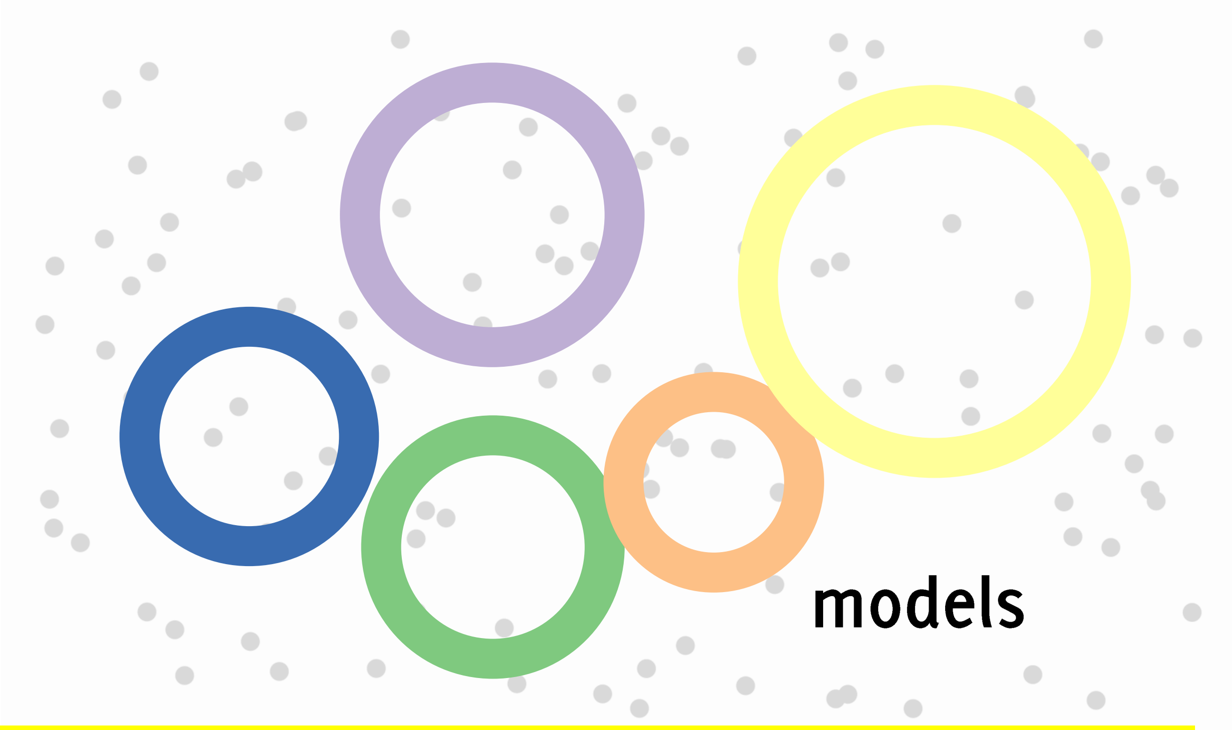
Credits Luca Magri

The problem of multi-model fitting (or structure recovery)

Given a set of data $X = \{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, possibly corrupted by noise and outliers, and a family of geometric models Θ

Goal: automatically estimate the models that best explain the data/discover the structures hidden in the data





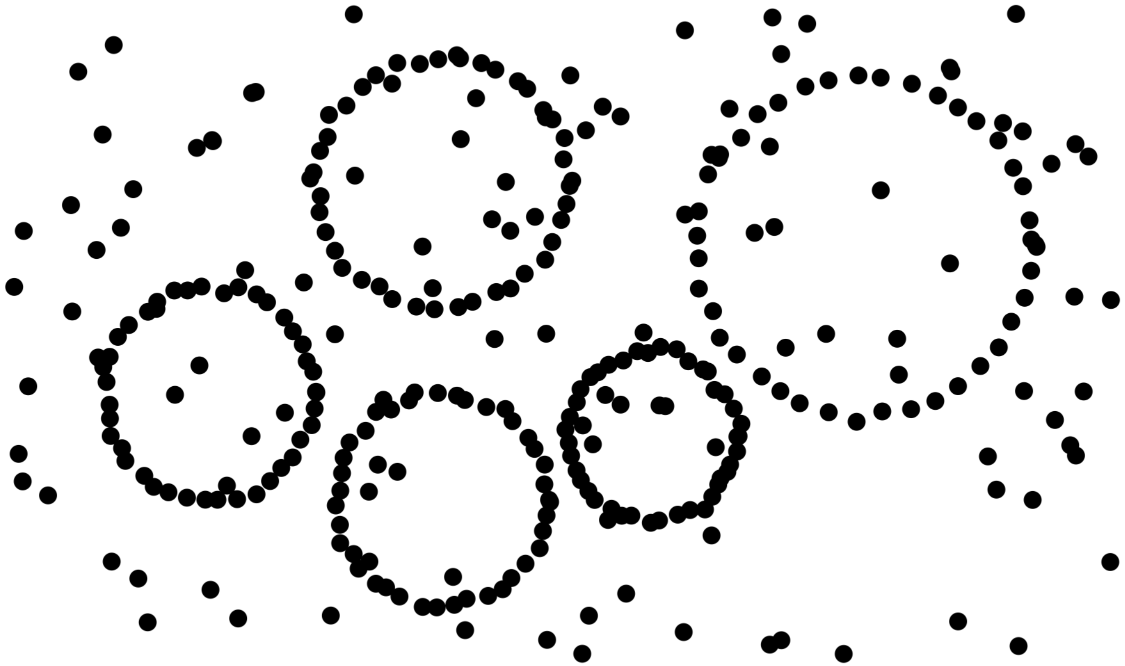
"in the eye of the beholder", mathematical descriptions of the data that an observer fits

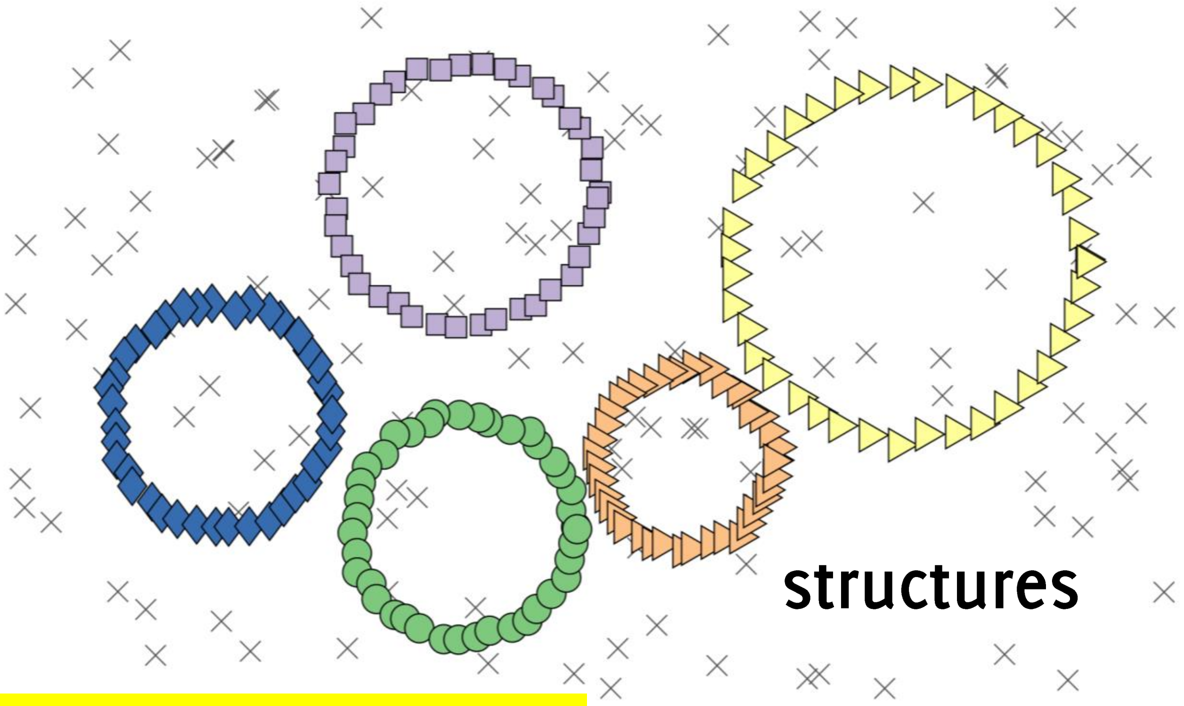
Credits Luca Magri

The problem of multi-model fitting (or structure recovery)

Given a set of data $X = \{x_1, \ldots, x_N\} \subset \mathbb{R}^d$, possibly corrupted by noise and outliers, and a family of geometric models Θ

Goal: automatically estimate the models that best explain the data/discover the structures hidden in the data

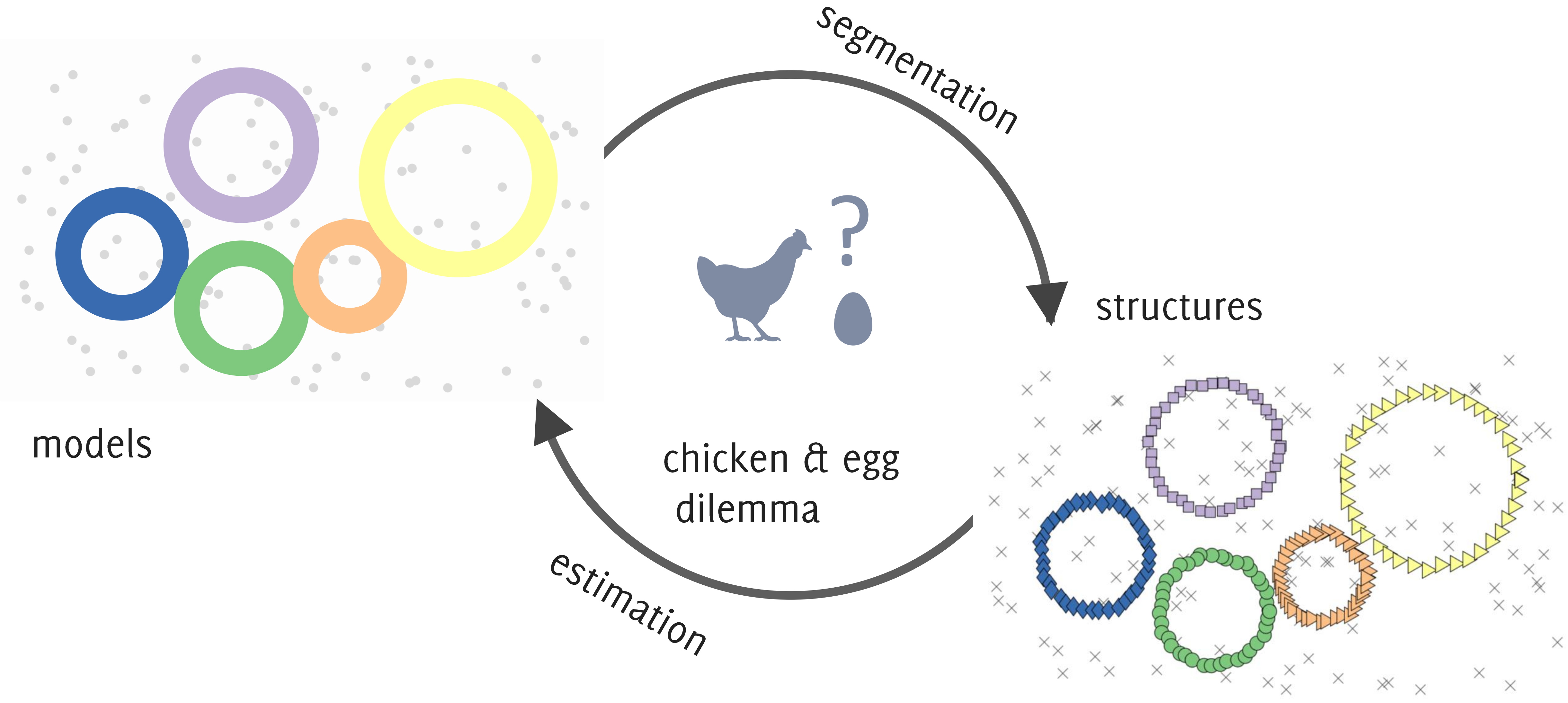




relations among the data, intrinsic to data

Credits Luca Magri

The Challenges of multi-model fitting

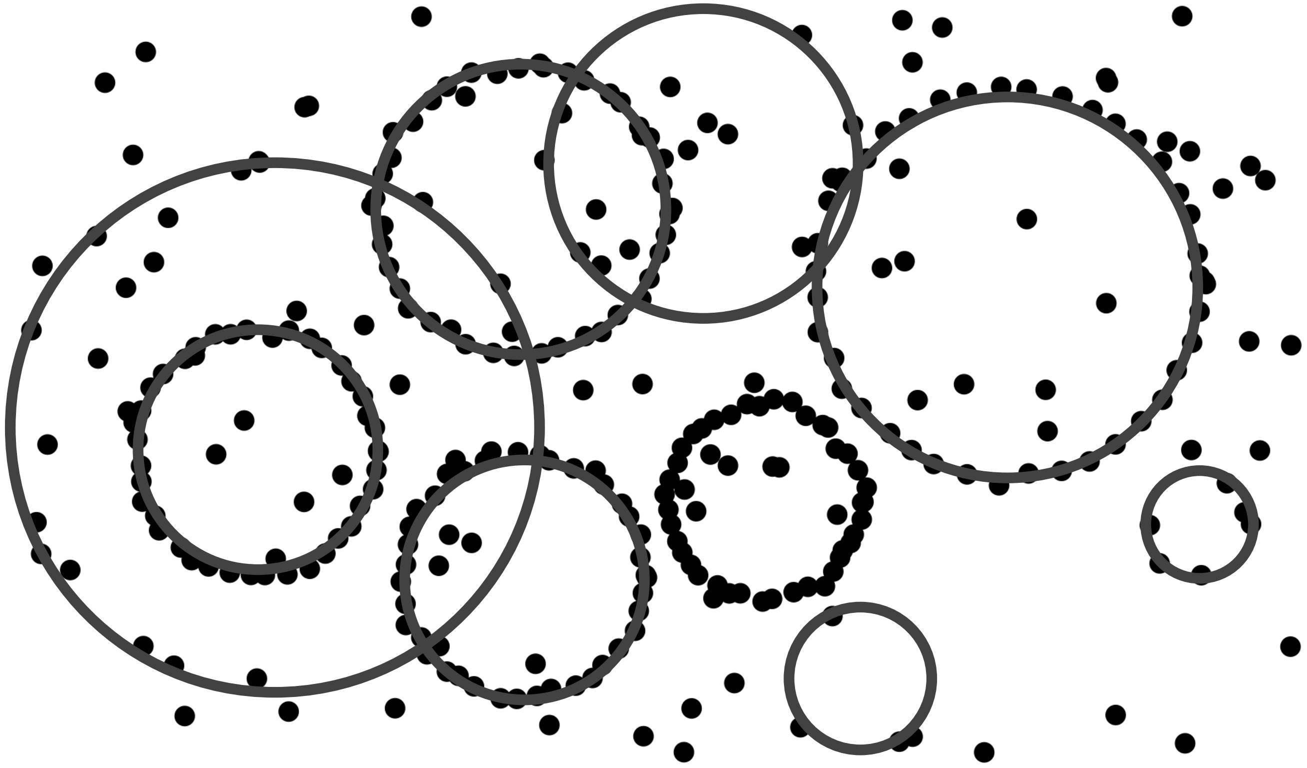


Credits Luca Magri

The Challenges of multi-model fitting

Number of models?

Inliers/outliers?



ill posed

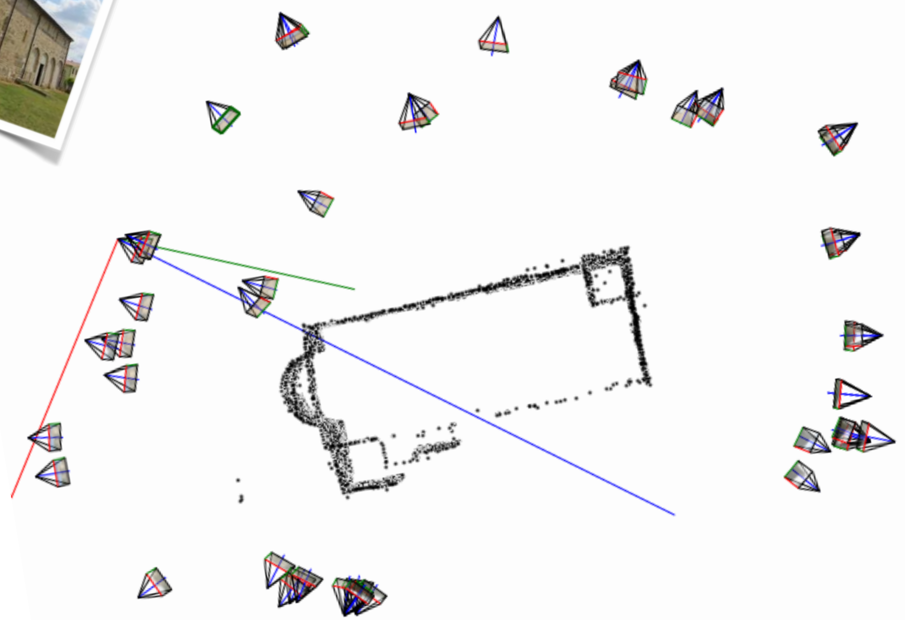
Credits Luca Magri

Multi-Model Fitting

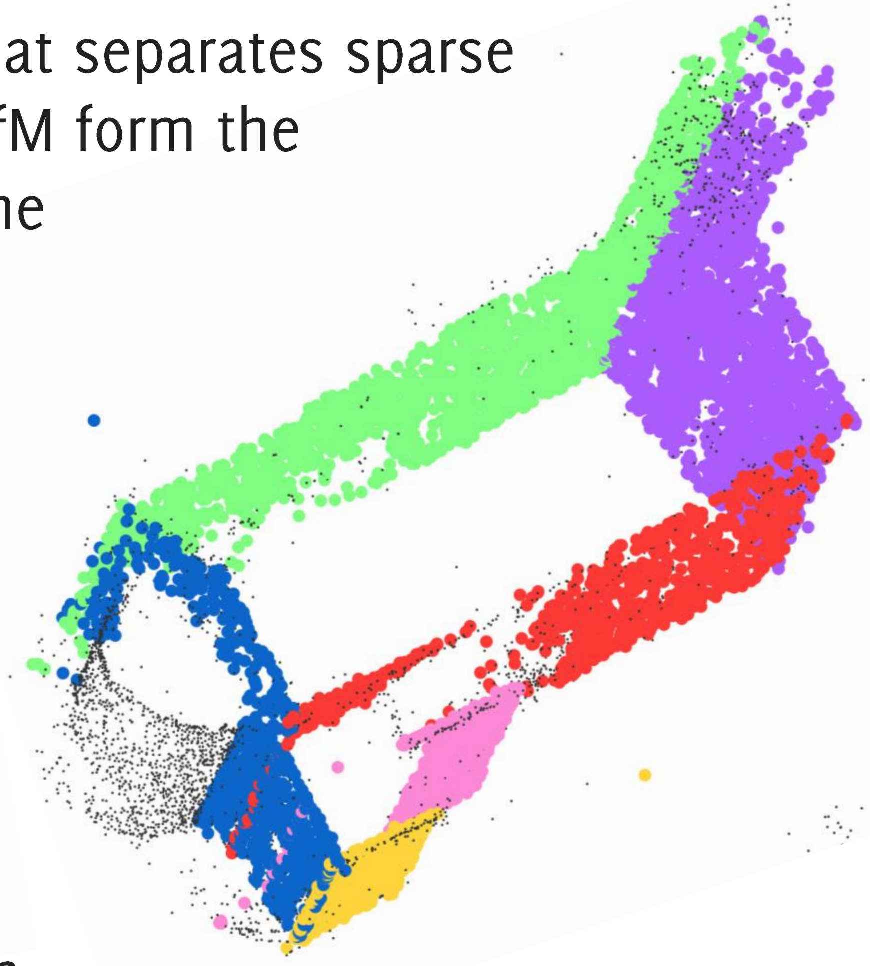
Multi model fitting applications: primitive fitting

Bridge the semantic gap that separates sparse point cloud coming from SfM form the understanding of a 3D scene

input images



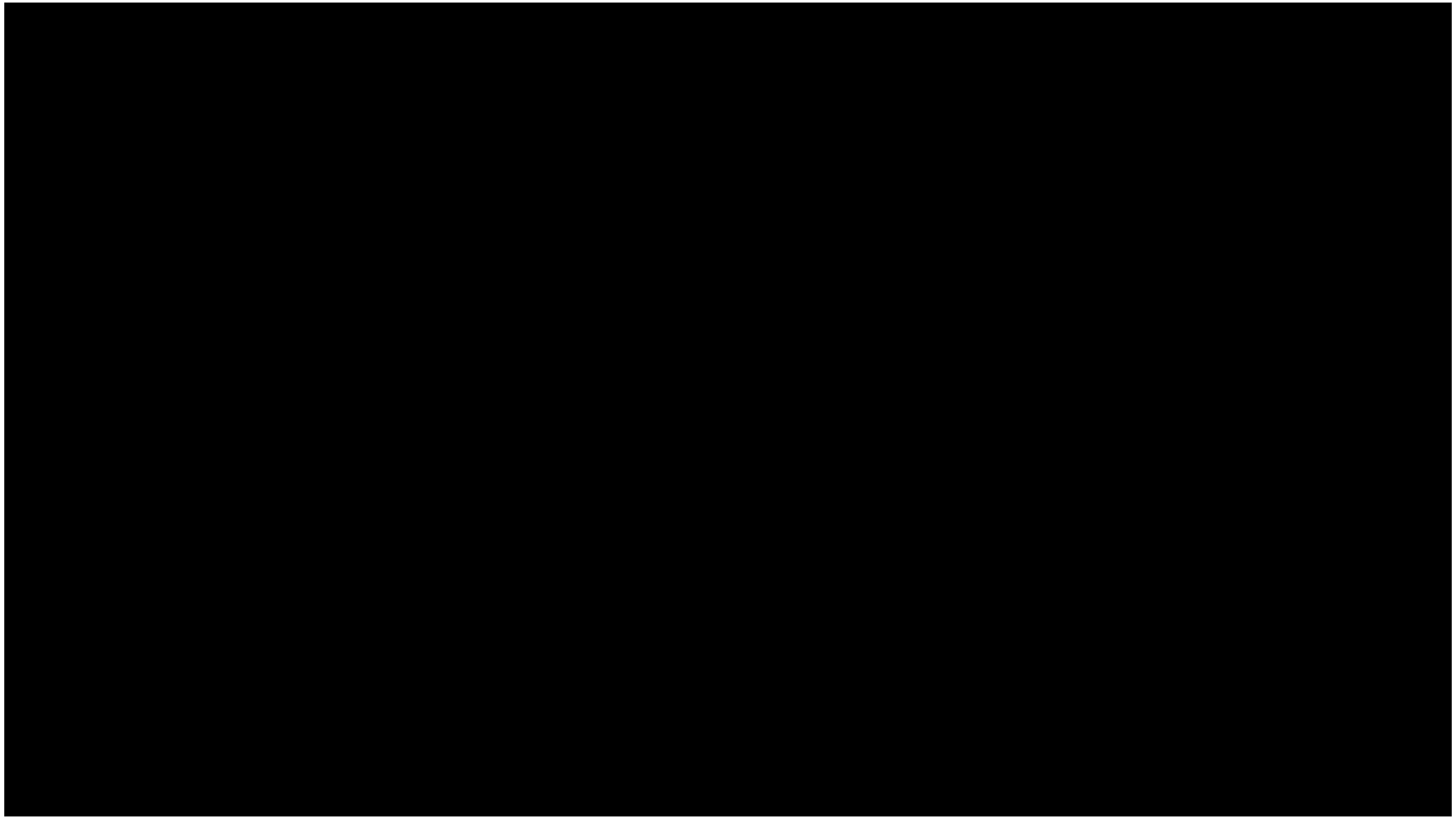
3D sparse reconstruction



$X \subset \mathbb{R}^3, \Theta =$ planes

Credits Luca Magri

Multimodel fitting for 3D scattered data



L. Magri, and A. Fusiello. "Reconstruction of interior walls from point cloud data with min-hashed J-linkage." 2018 3DV

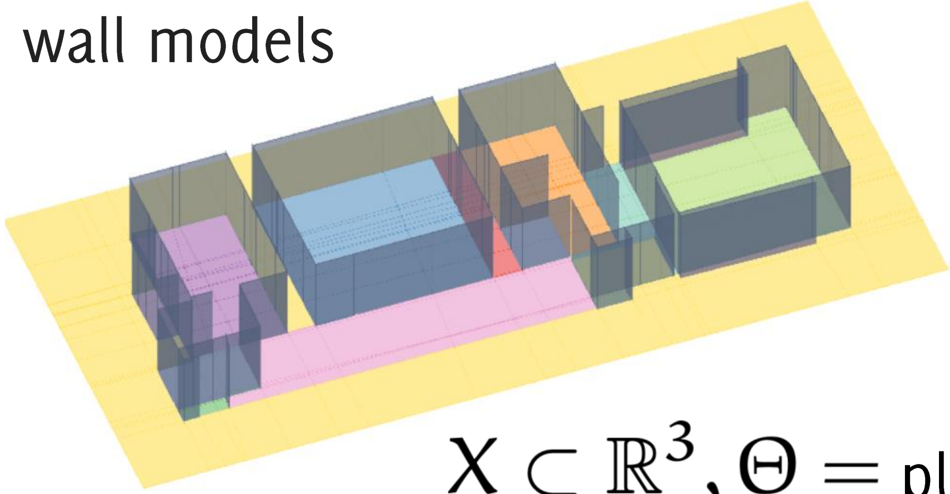
L. Magri, and A. Fusiello. "IMPROVING AUTOMATIC RECONSTRUCTION OF INTERIOR WALLS FROM POINT CLOUD DATA." International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences (2019).

L. Magri, and Andrea Fusiello. "T-linkage: A continuous relaxation of j-linkage for multi-model fitting." CVPR 2014

Multi model fitting applications: scan2bim

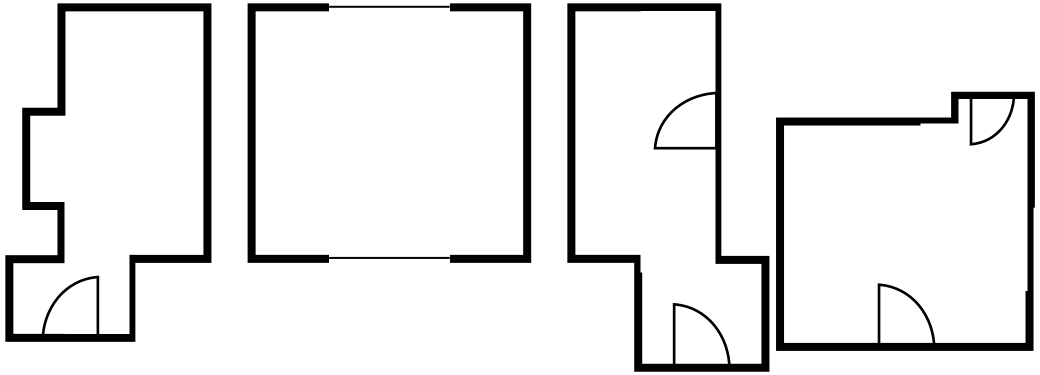
scanned point cloud

wall models



$X \subset \mathbb{R}^3, \Theta =$ planes

floor-plan



$X \subset \mathbb{R}^2, \Theta =$ lines

Given a scanned point cloud of an interior environment, detect its primary facility surfaces – such as floors, walls, and ceilings.

Credits Luca Magri

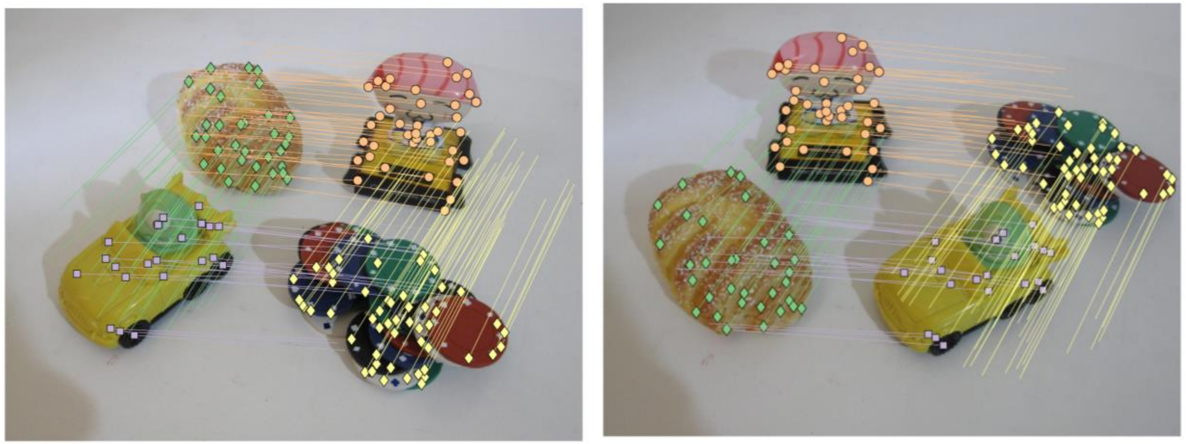
Multi model fitting applications: two view geometry

Geometric fit on corresponding matches across two images

plane detection

$X \subset \mathbb{R}^4, \Theta =$ homographies

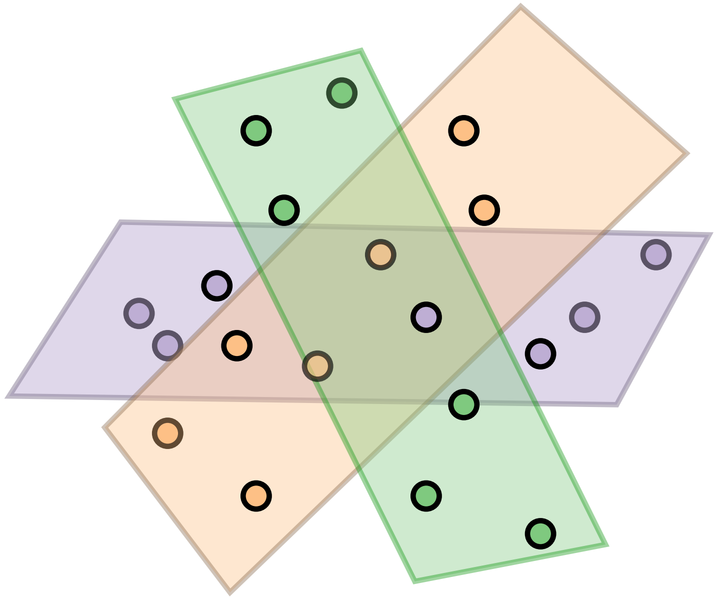
epipolar geometry



$X \subset \mathbb{R}^4, \Theta = \text{fundamental matrices}$

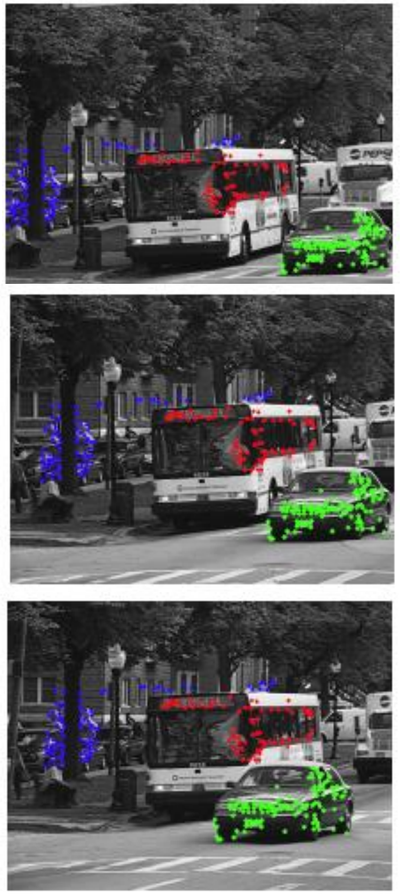
Credits Luca Magri

Multi model fitting applications: subspace clustering



$$X \subset \mathbb{R}^d, \Theta = \text{subspaces}$$

3D Video segmentation



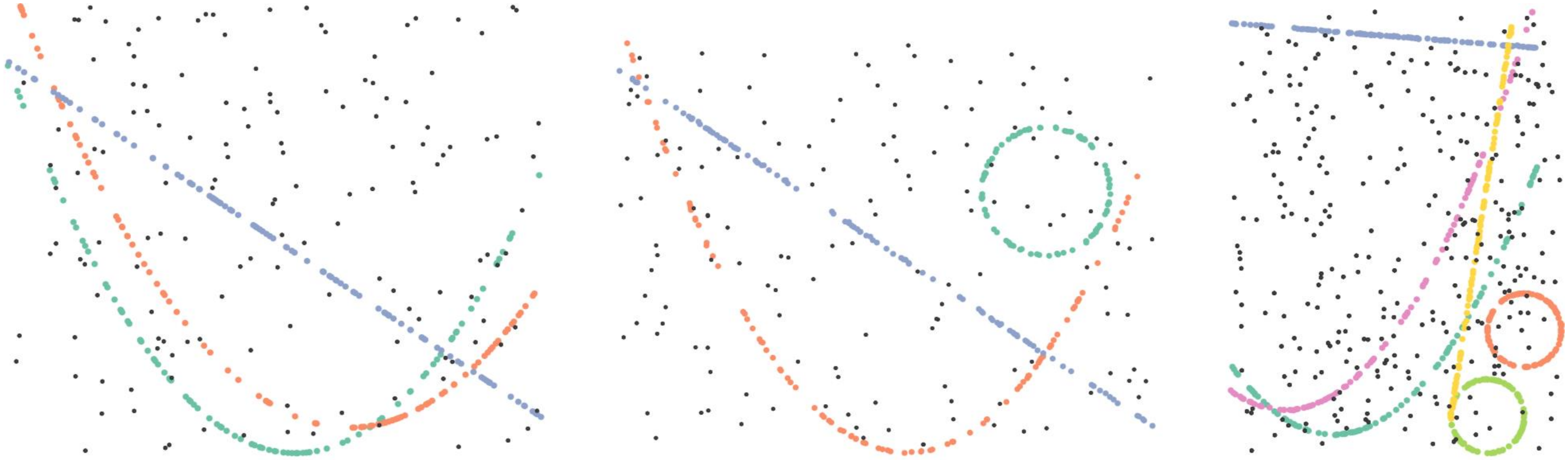
Face clustering

Credits Luca Magri

Template Detection

Our Collaboration with an Italian Company T&O

Multimodel (and multi-class) fitting


(a) (b) (c)

L. Magri, A. Fusiello. "Fitting Multiple Heterogeneous Models by Multi-Class Cascaded T-Linkage" CVPR 2019.

Multimodel (and multi-class) fitting

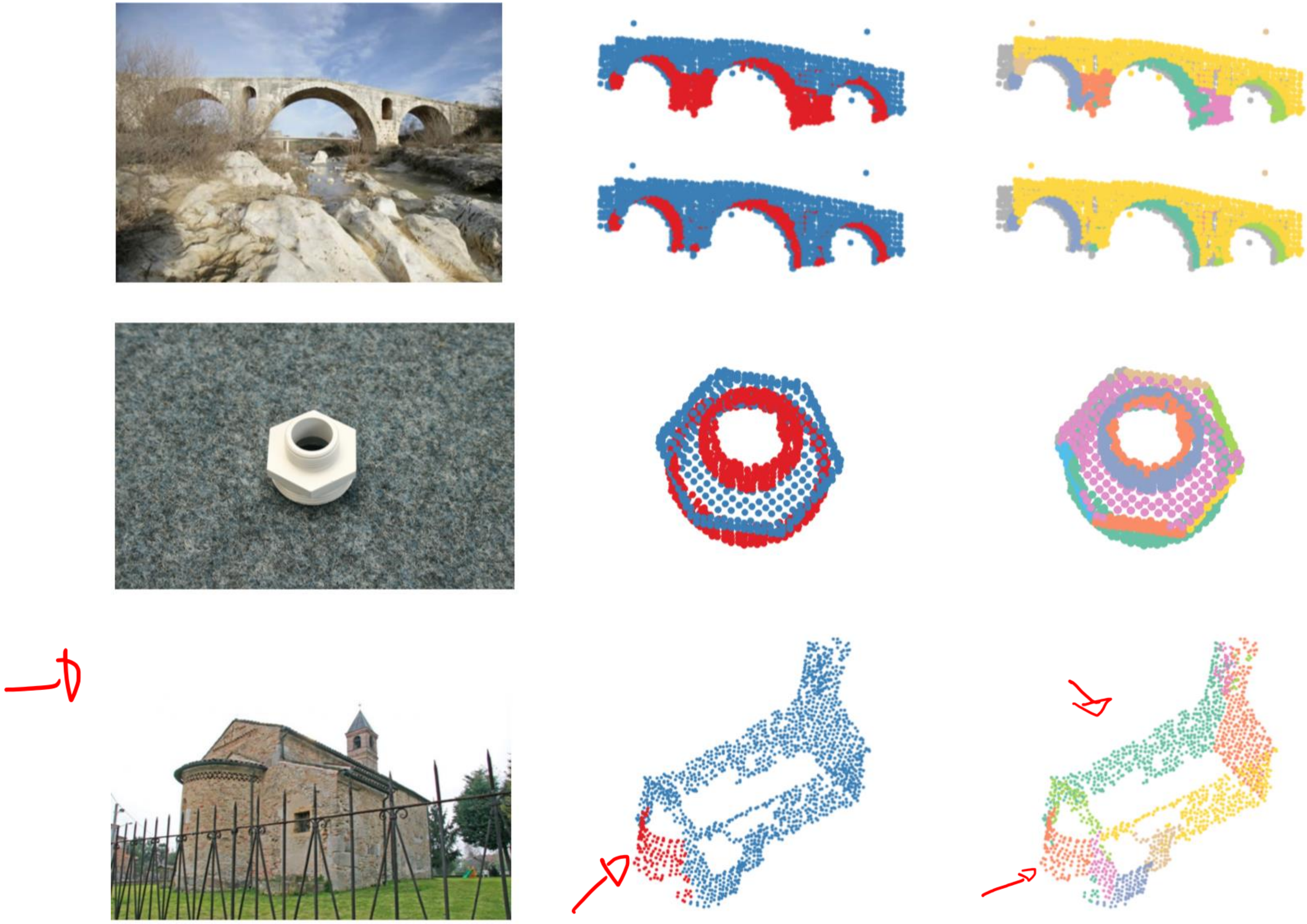


Image of the scene colour coded class colour coded model

L. Magri, A. Fusiello. "Fitting Multiple Heterogeneous Models by Multi-Class Cascaded T-Linkage" CVPR 2019.

MultiLink

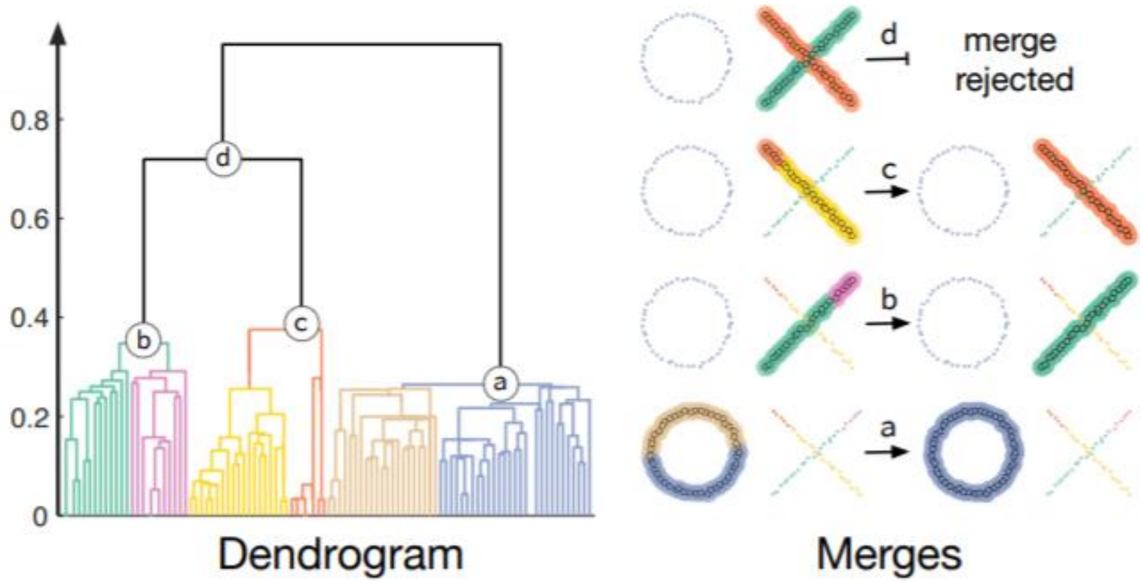
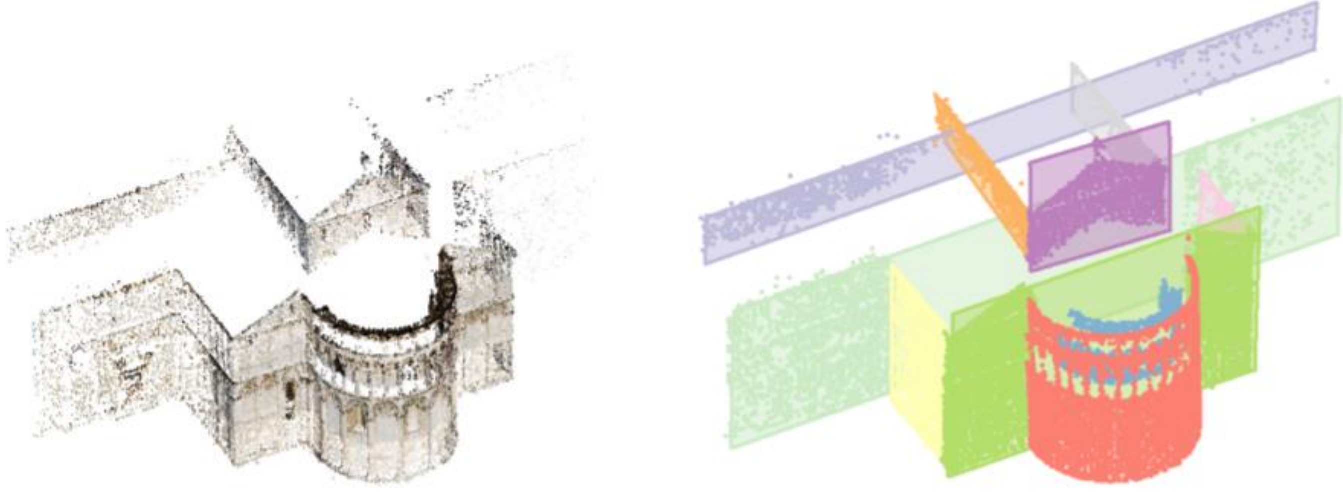


Figure 2: MultiLink combines single-linkage clustering and GRIC. Clusters are merged as long as the GRIC score improves when fitting suitable models on-the-fly. Colors indicate how cluster aggregation proceeds in the dendrogram.

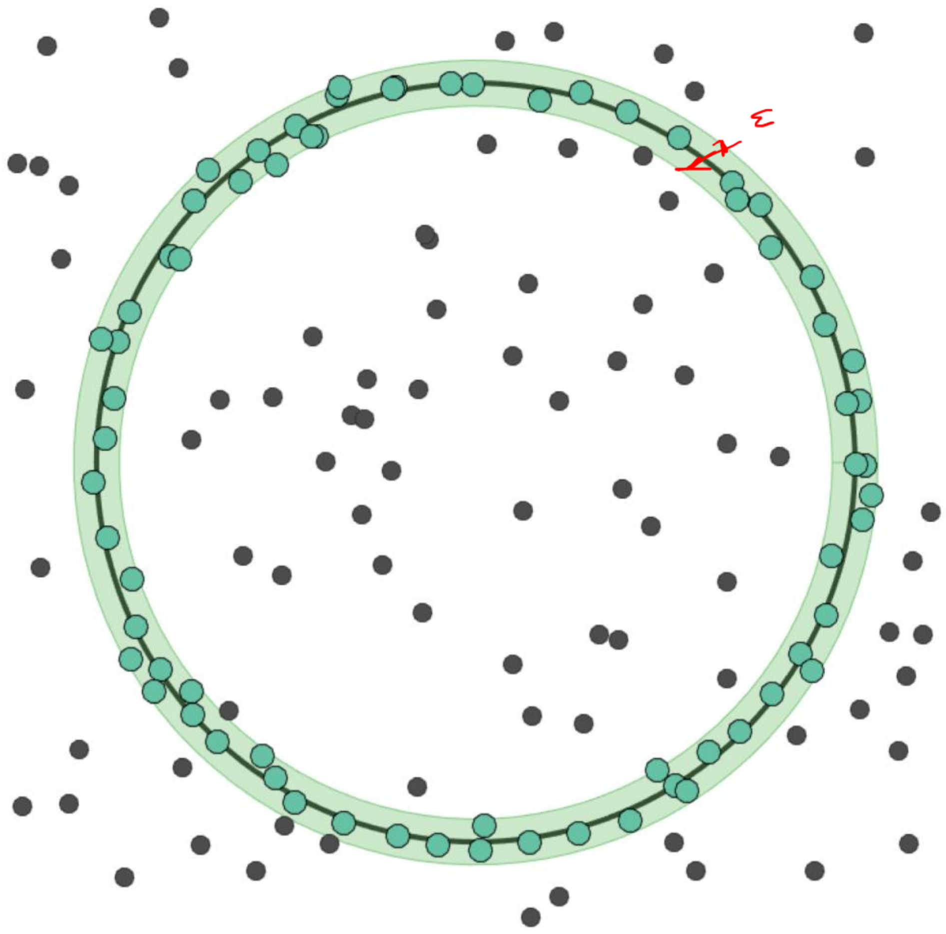


(a) Input point cloud (b) Recovered structures

[L. Magri, F. Leveni, G. Boracchi, MultiLink: Multi-class Structure Recovery via Agglomerative Clustering and Model Selection, CVPR 2021]()

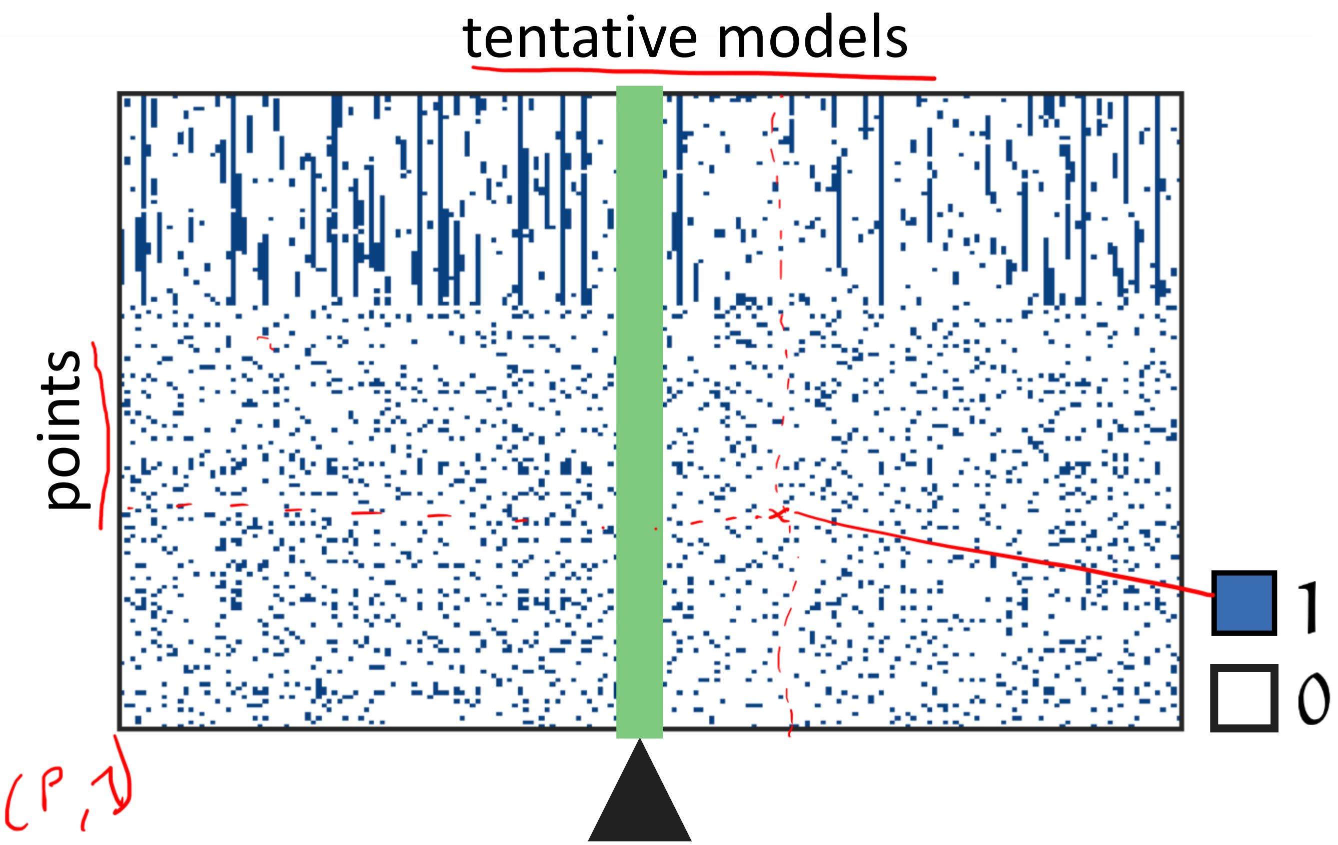
Multi-model Fitting Solutions

Let's go back to RanSaC



Data driven search of model space

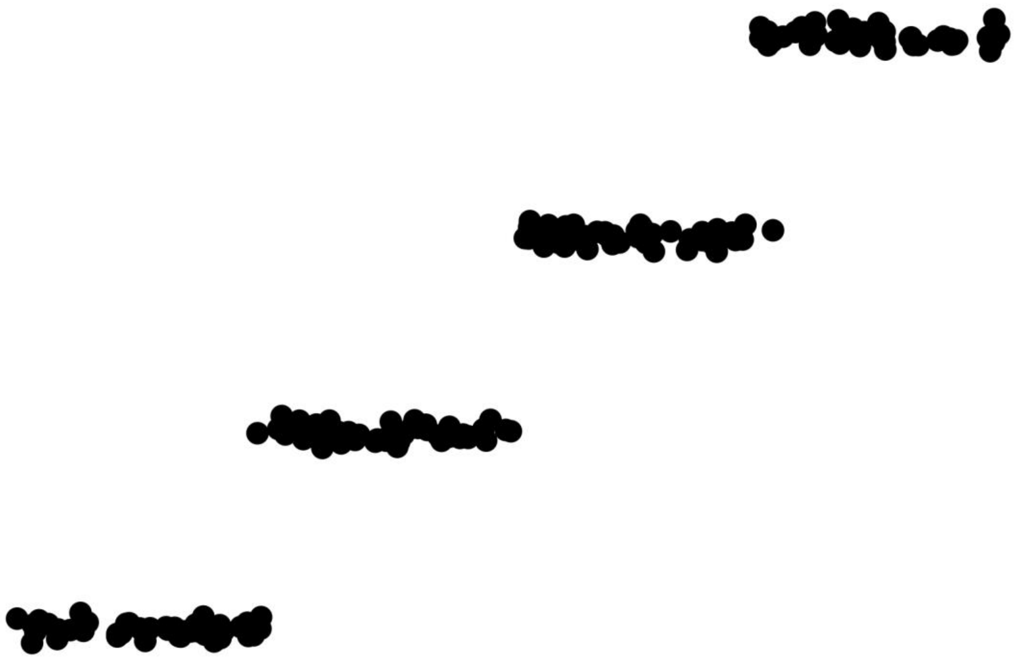
$$H = \{\theta_1, \theta_2, \ldots, \theta_m\} \approx \Theta$$



pick the column with the maximum sum

Sequential RanSaC [Zuliani 05]

Start RanSaC on the dataset X searching for the best fit for a single instance of the model



Line fitting example

Sequential RanSaC [Zuliani 05]

Start RanSaC on the dataset X searching for the best fit for a single instance of the model

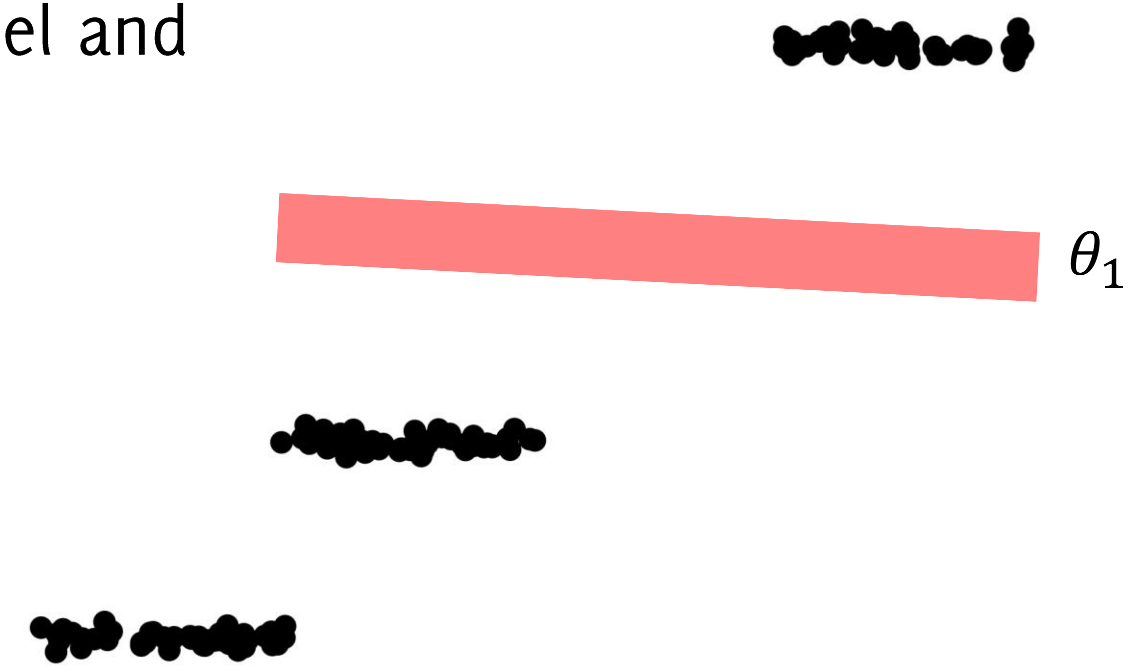
Once detected a model θ_1, keep the model and remove all the inliers

Line fitting example

Sequential RanSaC [Zuliani 05]

Start RanSaC on the dataset X searching for the best fit for a single instance of the model

Once detected a model θ_1, keep the model and remove all the inliers from X



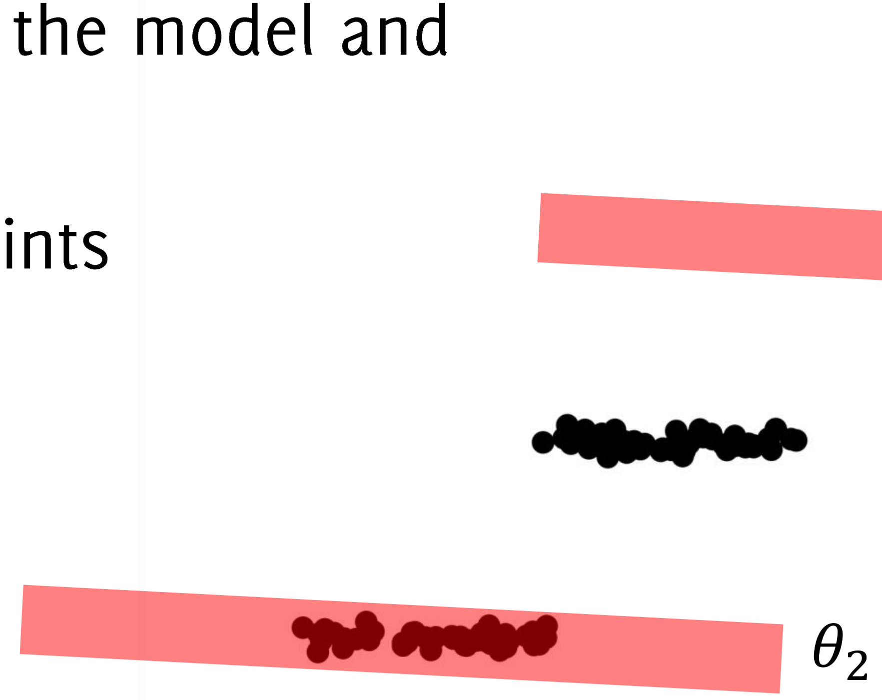
Line fitting example

Sequential RanSaC [Zuliani 05]

Start RanSaC on the dataset X searching for the best fit for a single instance of the model

Once detected a model θ_1, keep the model and remove all the inliers from X

Iterate through the remaining points



Line fitting example

Sequential RanSaC [Zuliani 05]

Start RanSaC on the dataset X searching for the best fit for a single instance of the model

Once detected a model θ_1, keep the model and remove all the inliers from X

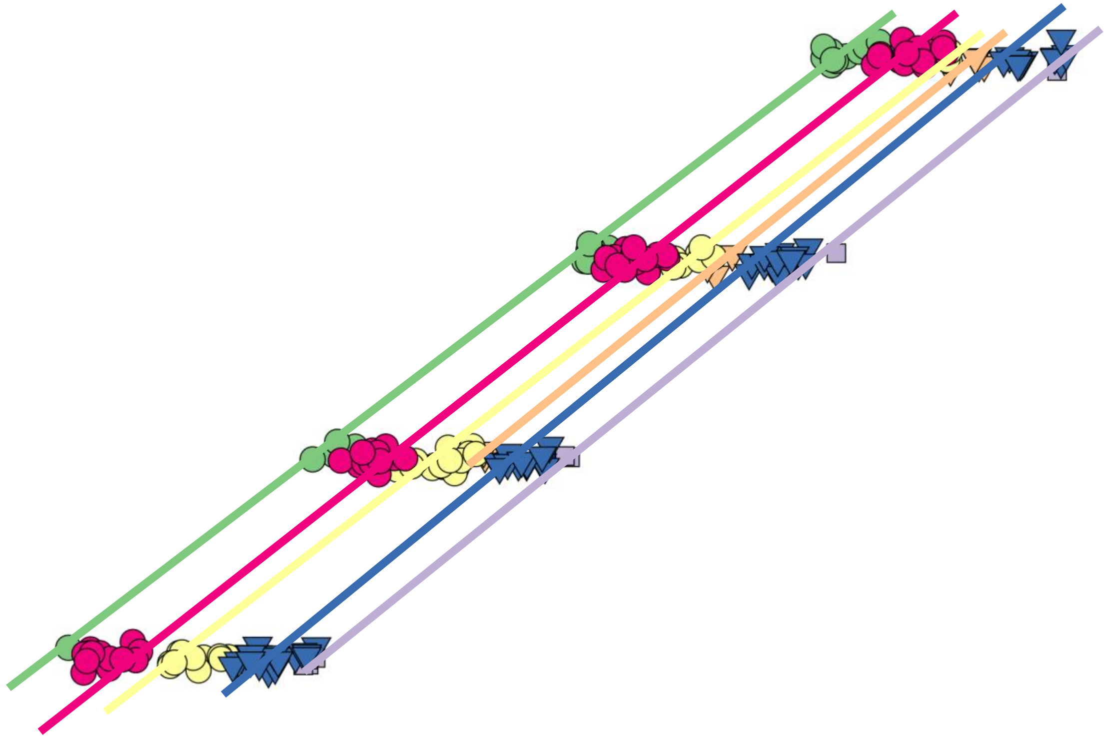
Iterate through the remaining points until there are no models with a sufficiently large consensus

Line fitting example

G. Boracchi

Sequential RanSaC [Zuliani 05]

Unfortunately, this does not fit well with the multi-model scenario and the problem becomes even more severe in presence of outliers



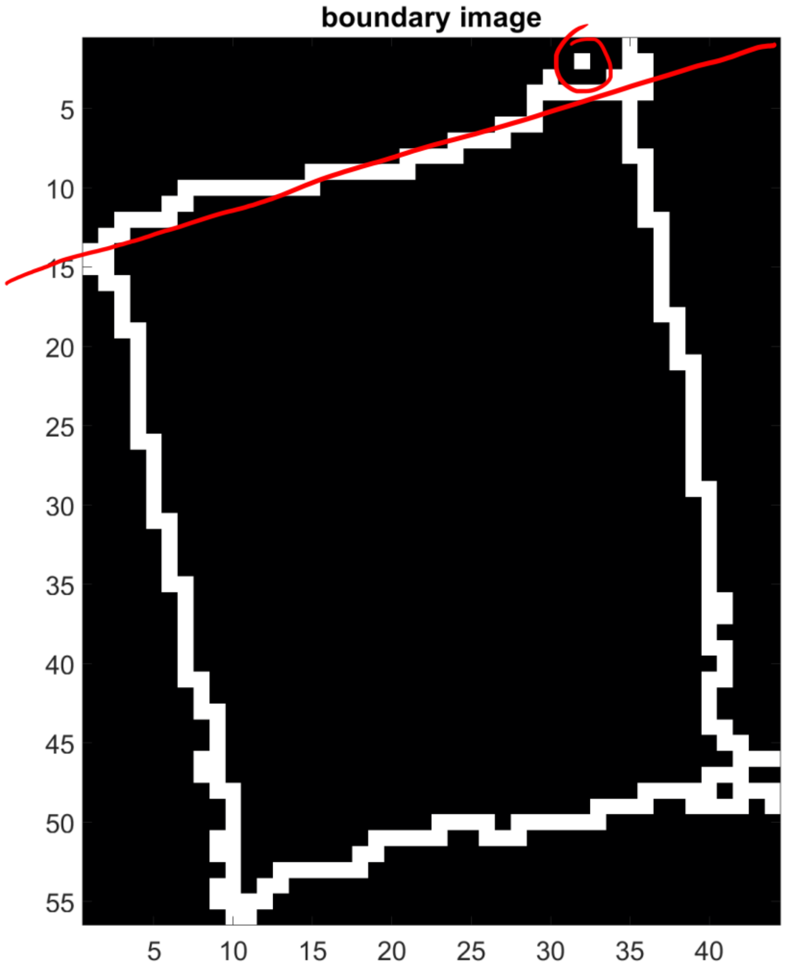
Line fitting example

Line Detection: Hough Transform

Extracting Line Equations From Edges

Line Detection: The problem

Finding all the lines passing through points in (a binary) image

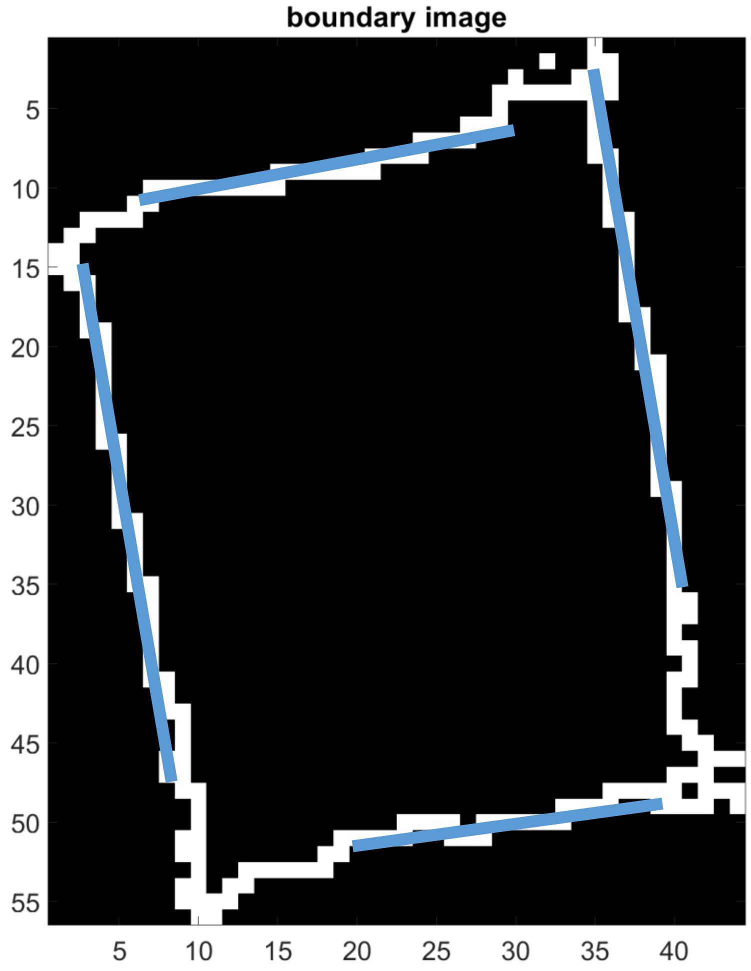


Line Detection: The problem

Finding all the lines passing through points in (a binary) image

Finding lines means

- Having an analytical expression for each line
- Estimating its direction, length
- Thus, clustering points belonging to the same segment

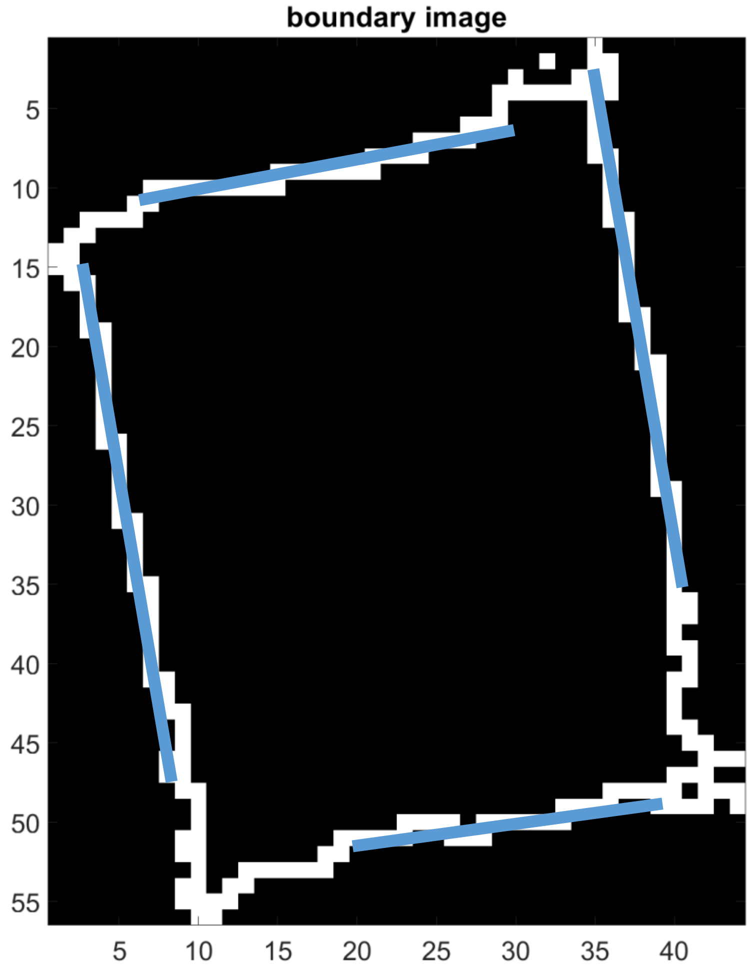


Line Detection: The problem

Brute-force attempt:

Given n points in a binary image, find subsets that lie on straight lines

- Compute all the lines passing through **any pair of points**
- Check **subsets of points** that belong / are close to these lines

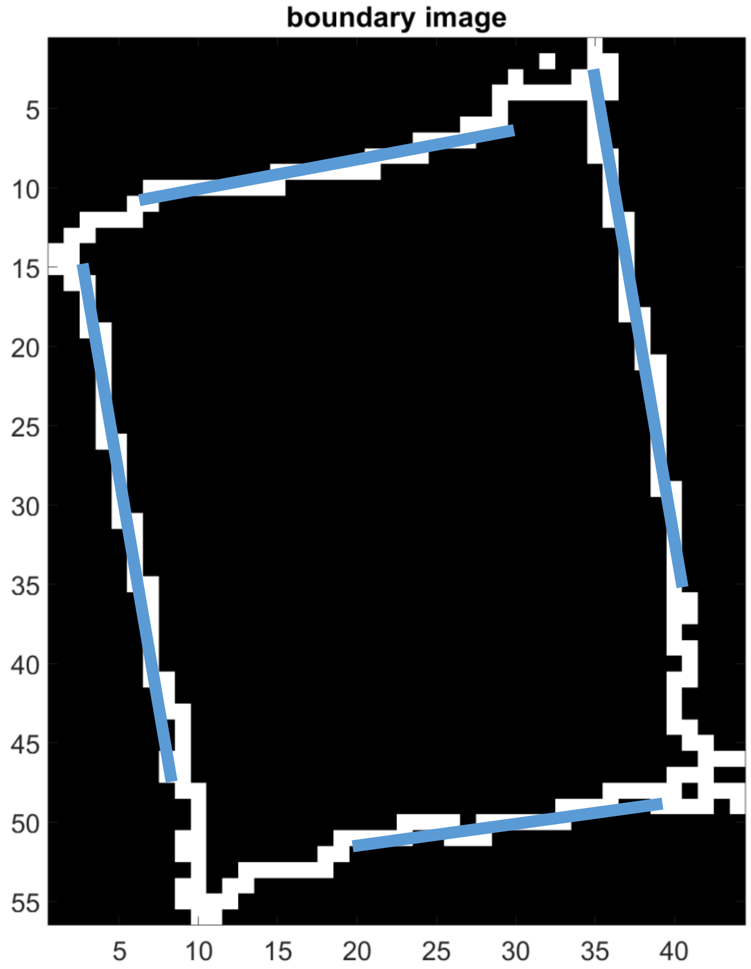


Line Detection: The problem

Brute-force attempt:

This requires computing

- $\frac{n(n-1)}{2}$ straight lines
- $n\left(\frac{n(n-1)}{2}\right)$ comparisons
- Computationally prohibitive task in all but the most trivial applications $\sim n^3$

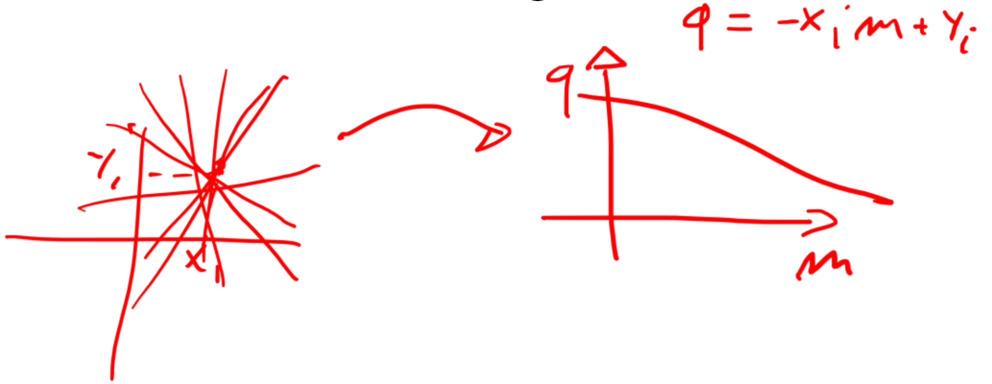


Hough Transform

Identify lines in the *"parameter space"* i.e. in the space of the parameters identifying lines (m, q). Let a straight line be:

$$y = mx + q$$

Now, for a given point (x_i, y_i), the equation $q = -x_i m + y_i$ in the variables m, q denotes the coefficients of all the lines that belong to the star passing through (x_i, y_i)

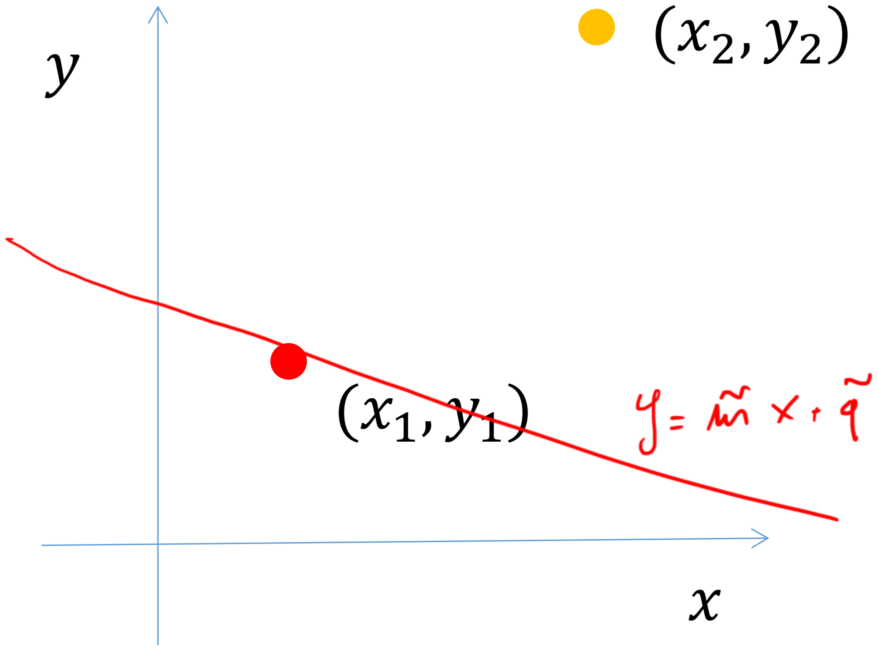


Key intuition:

$$q = -x_i m + y_i$$

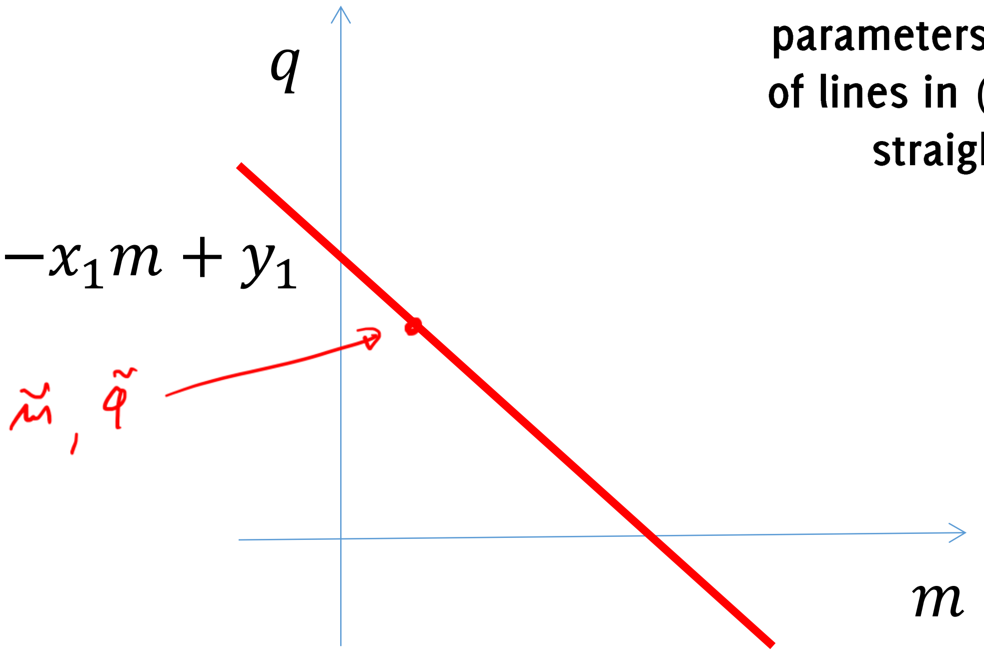
can be also seen as the equation of a straight line in m, q in **the parameter space**

Line Intersections in the parameter space



Point space

$$q = -x_1 m + y_1$$

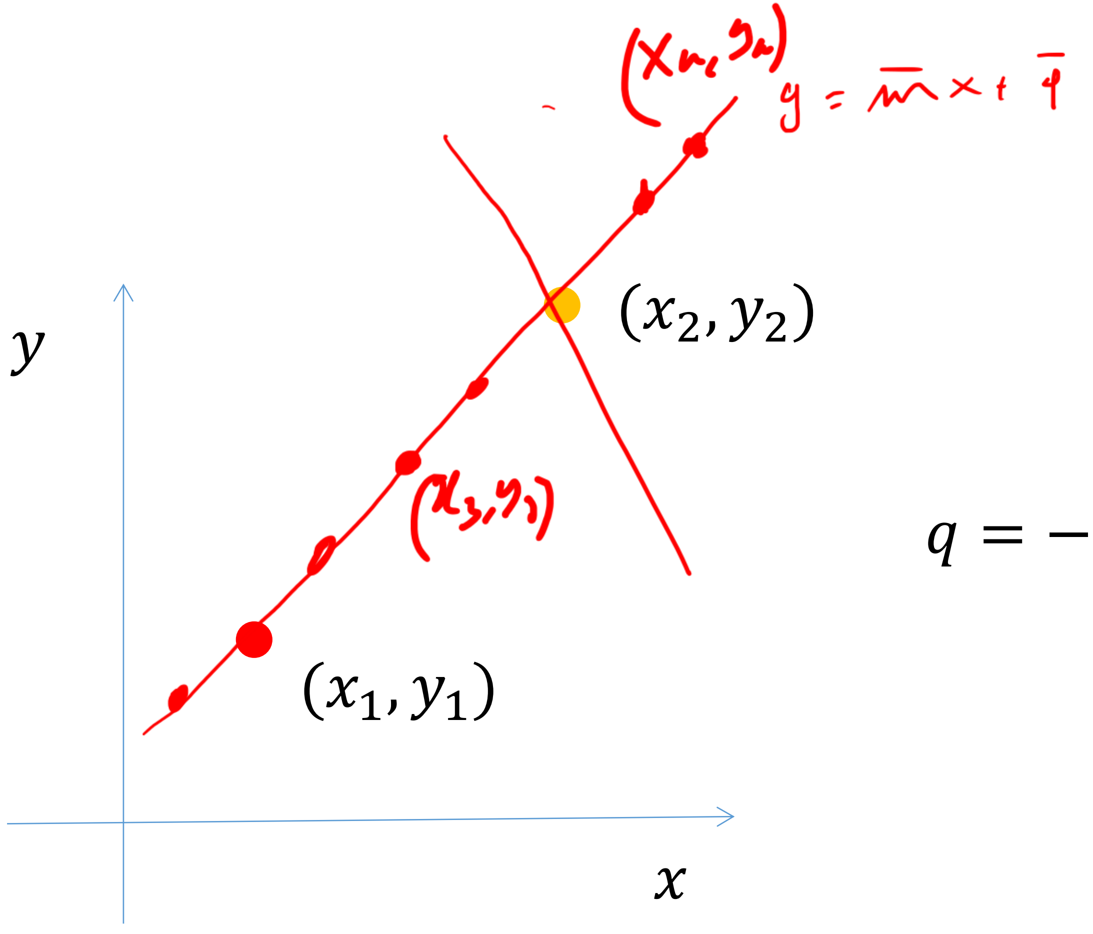


Parameter space

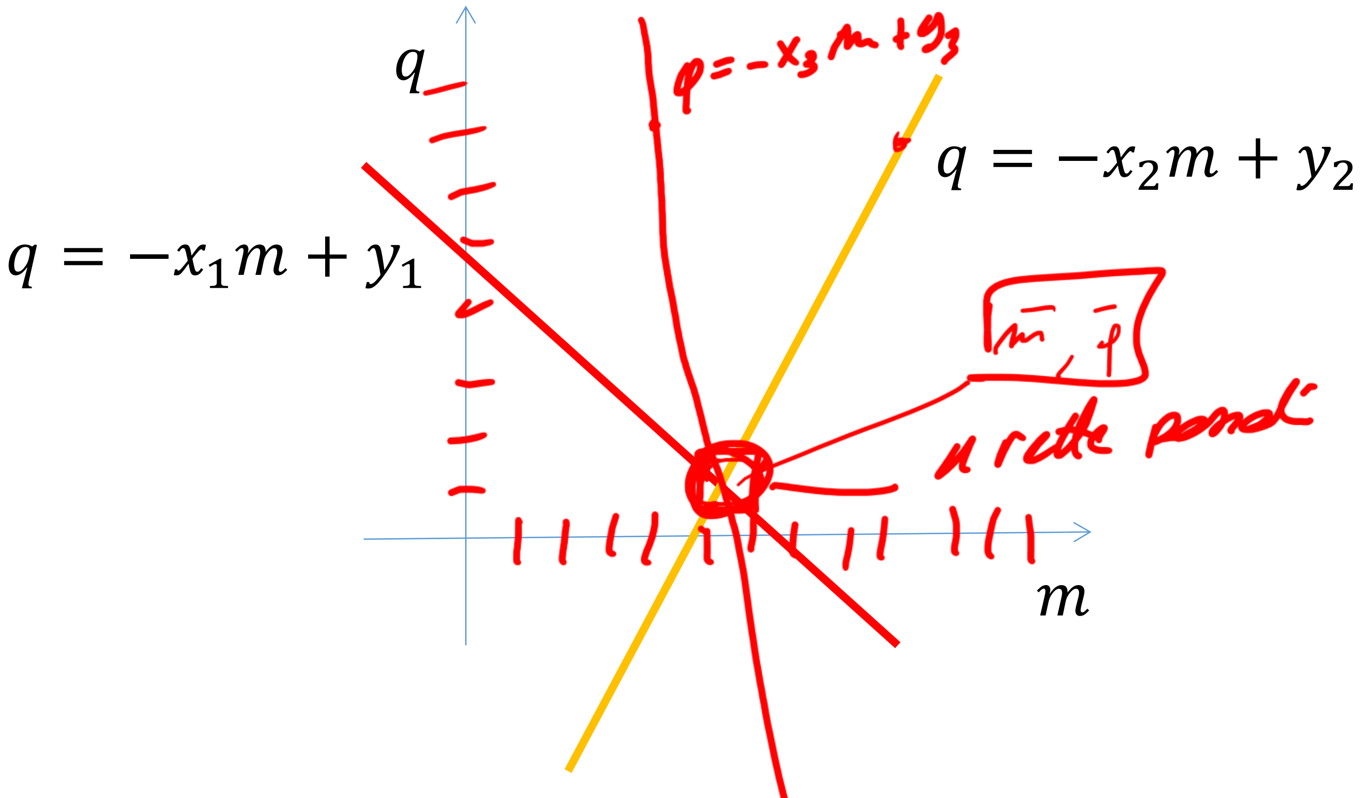
The set of all the parameters of the star of lines in (x_1, y_1) is a straight line

Line Intersections in the parameter space

The two straight lines in the parameter space intersect in a point, corrisponding to a line passing to both (x_1, y_1) and (x_2, y_2)

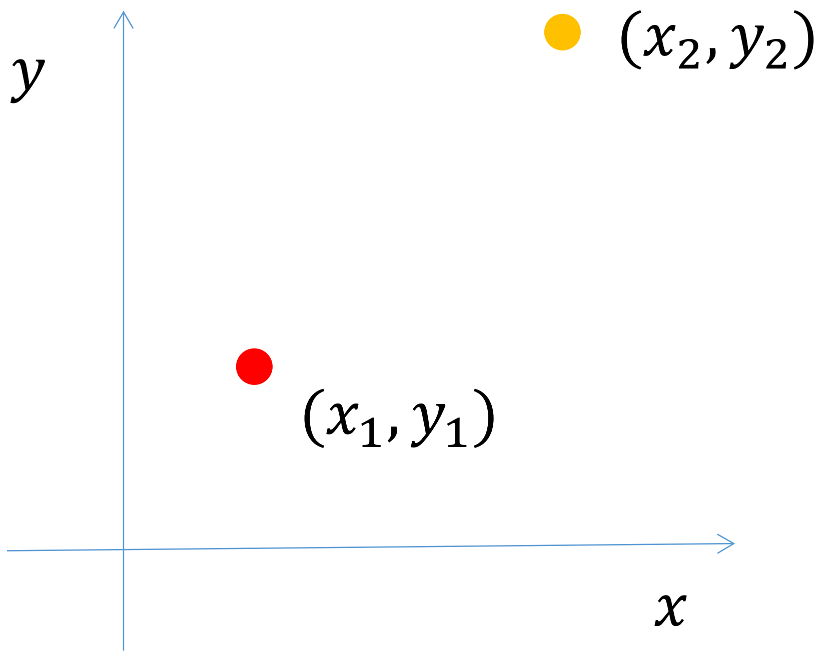


Point space

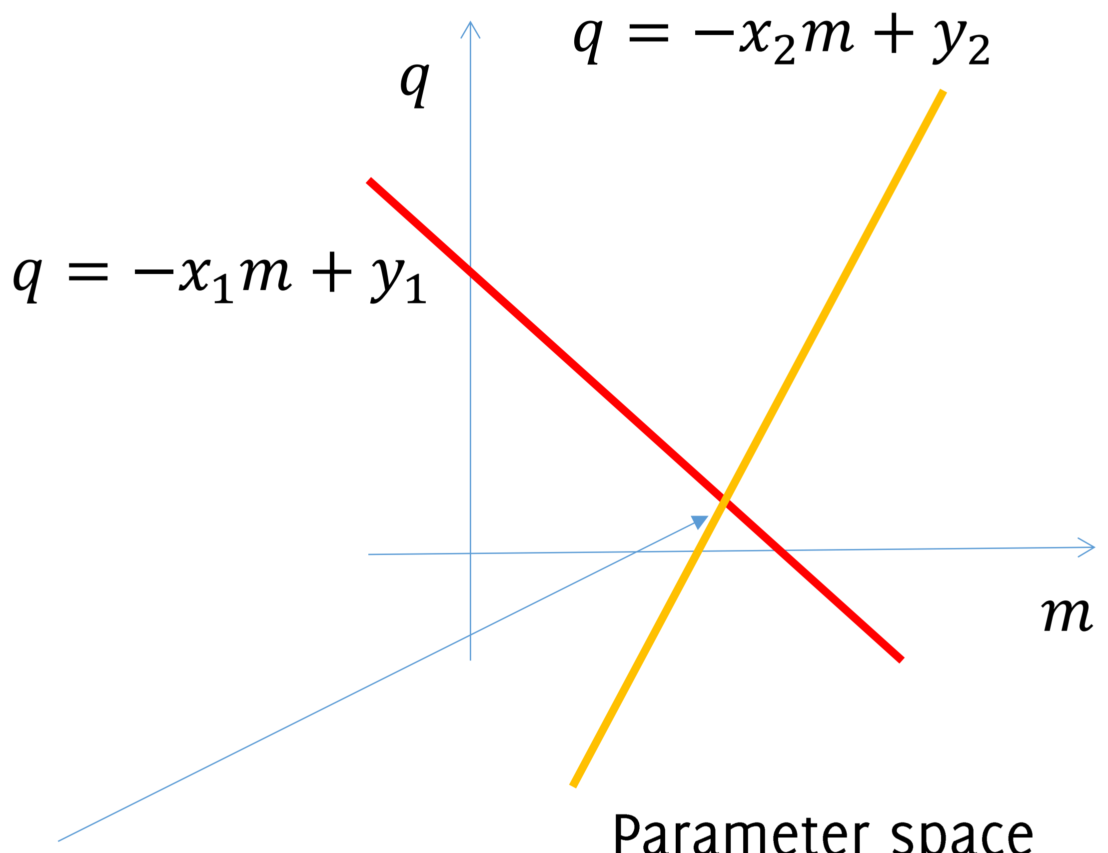


Parameter space

Line Intersections in the parameter space



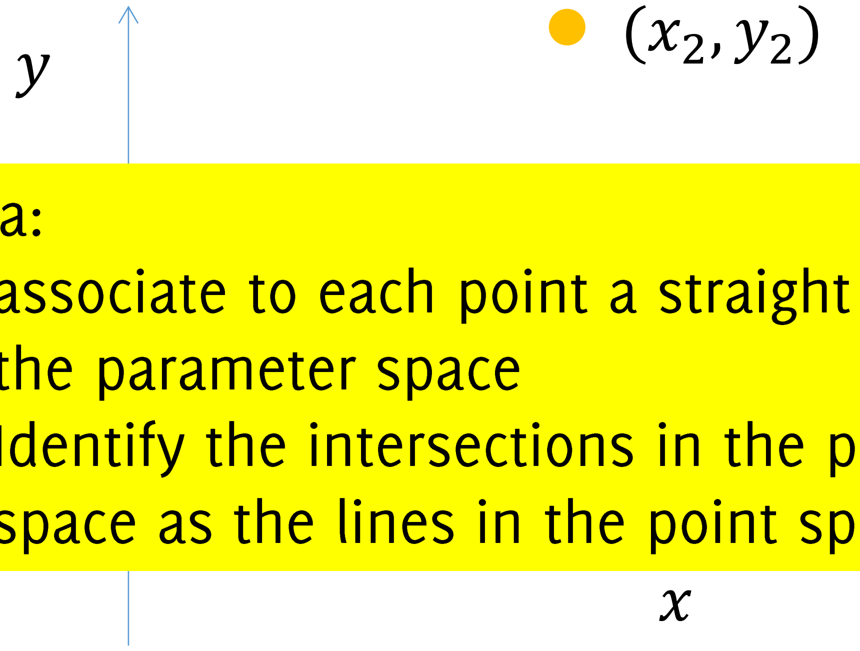
Point space



Parameter space

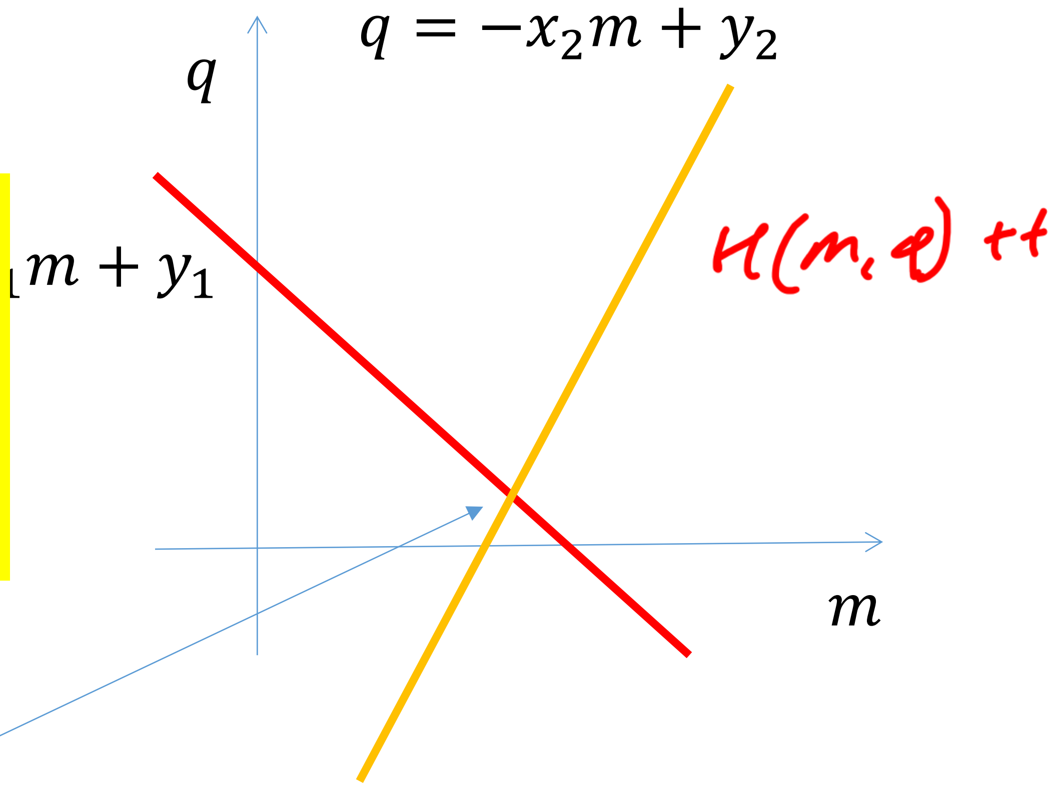
$\bar{m}, \bar{q}$ such that $y = \bar{m}x + \bar{q}$ passes through both (x_1, y_1) and (x_2, y_2)

Intersections in the parameter space



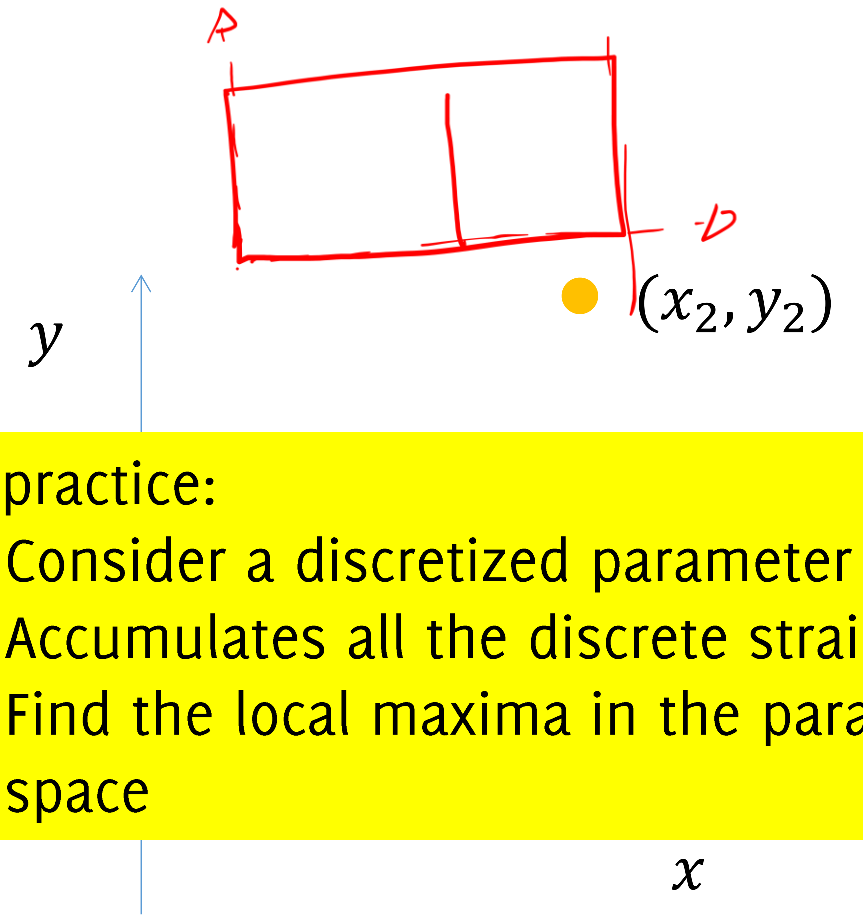
Idea:

- associate to each point a straight line in the parameter space
- Identify the intersections in the parameter space as the lines in the point space

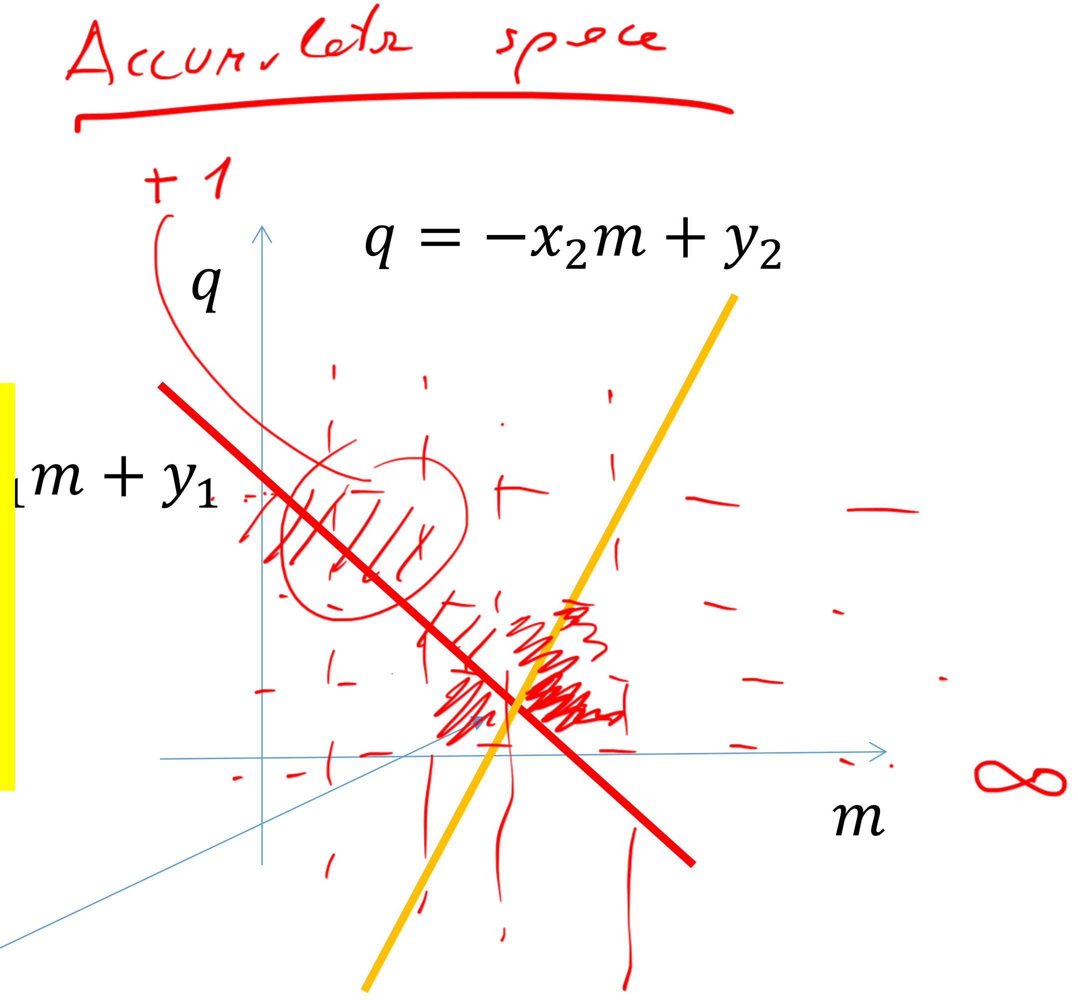


$\overline{m}, \overline{q}$ such that $y = \overline{m}x + \overline{q}$ passes through both (x_1, y_1) and (x_2, y_2)

Intersections in the parameter space



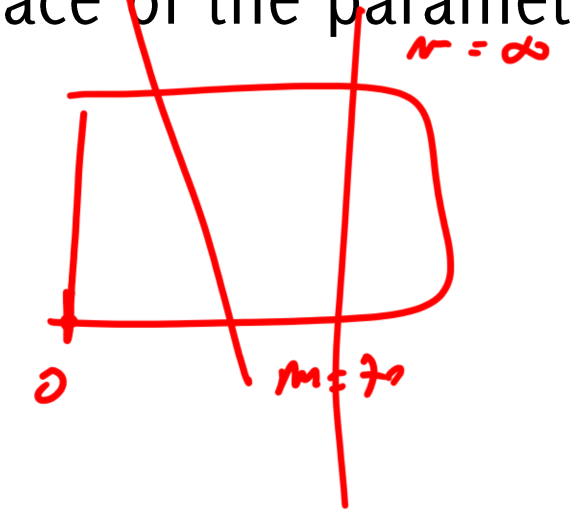
In practice:
- Consider a discretized parameter space
- Accumulates all the discrete straight lines
- Find the local maxima in the parameter space



$\bar{m}, \bar{q}$ such that $y = \bar{m}x + \bar{q}$ passes through both (x_1, y_1) and (x_2, y_2)

Hough Transform

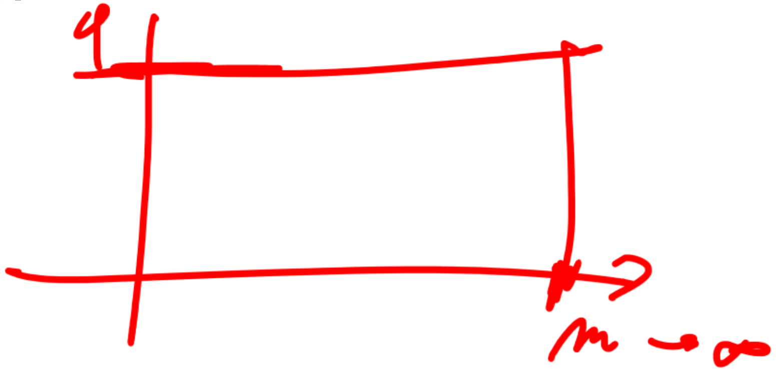
Identify lines in the "parameter space" i.e. in the space of the parameters identifying lines.



$$q = -x_i m + y_i, \qquad \forall (x_i, y_i)$$

Core Idea:

- Discretize the parameter space where m, q live
- Accumulate the consensus in the parameter space by summing +1 at those bins where a straight line passess through
- Locate local maxima in the accumulator space



Major issue: m goes to infinity at vertical lines!

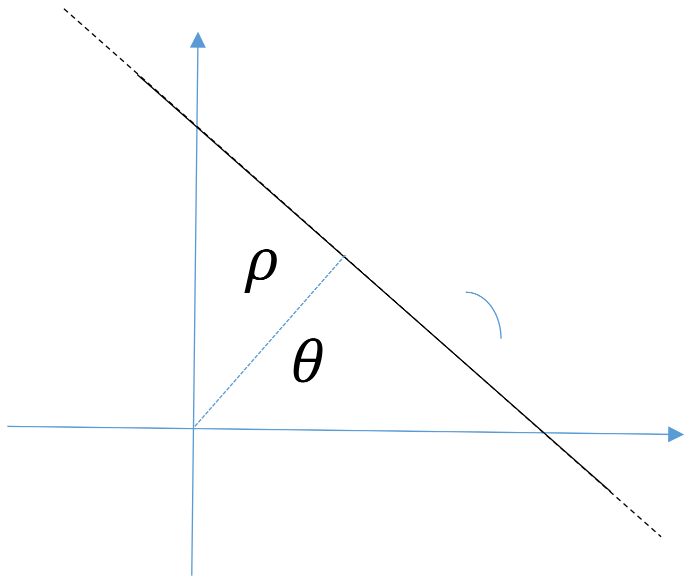
New Parametrization for Hough Transform

There is a more convenient way of expressing a strainght line, for this purpose:

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

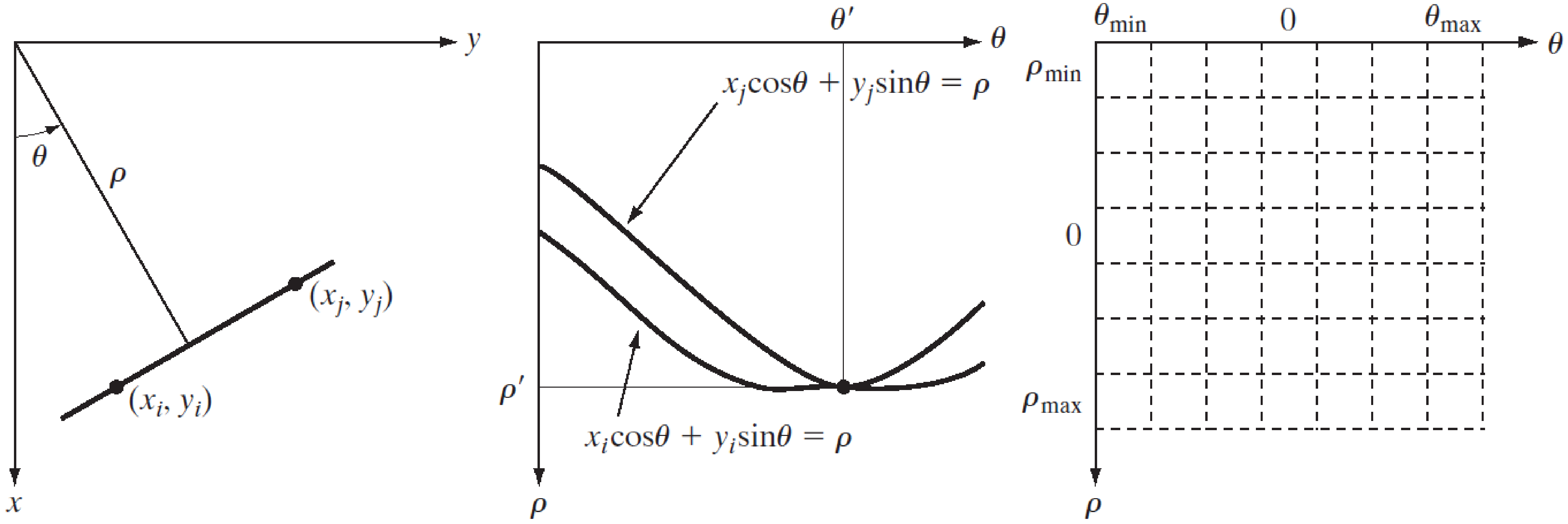
Where $\left\{(\rho, \theta),\ \rho \in [-L, L],\ \theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]\right\}$ being L the image diagonal.

ρ is the signed distance from the origin



Same as before: a line in the image space is a point in parameter Hough space.

New Parametrization of straight lines



Gonzalez and Woods «Digital image Processing», Prentice Hall;, 3° edition

Hough Transform

The Hough transform identifies **through an optimized voting procedure** the most represented lines

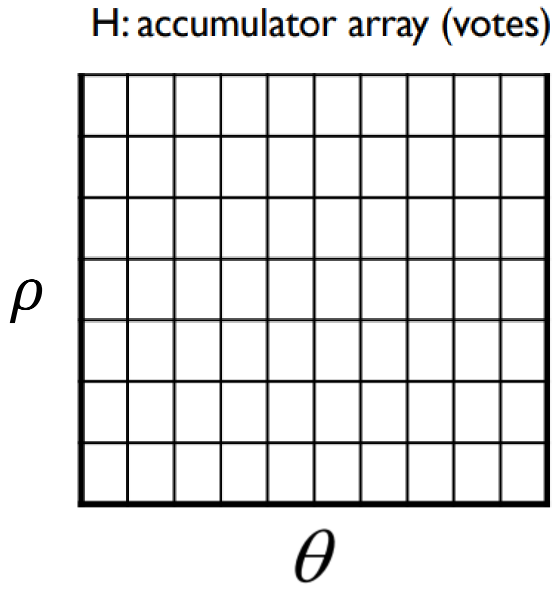
The voting procedure is performed in the «accumulator space» which is a grid in (ρ, θ)-domain, for all the possible values.

From the Accumulator space we then extract local maxima, namely pairs (ρ, θ) corresponding to lines passing through most of points

What is the maximum size of the domain?

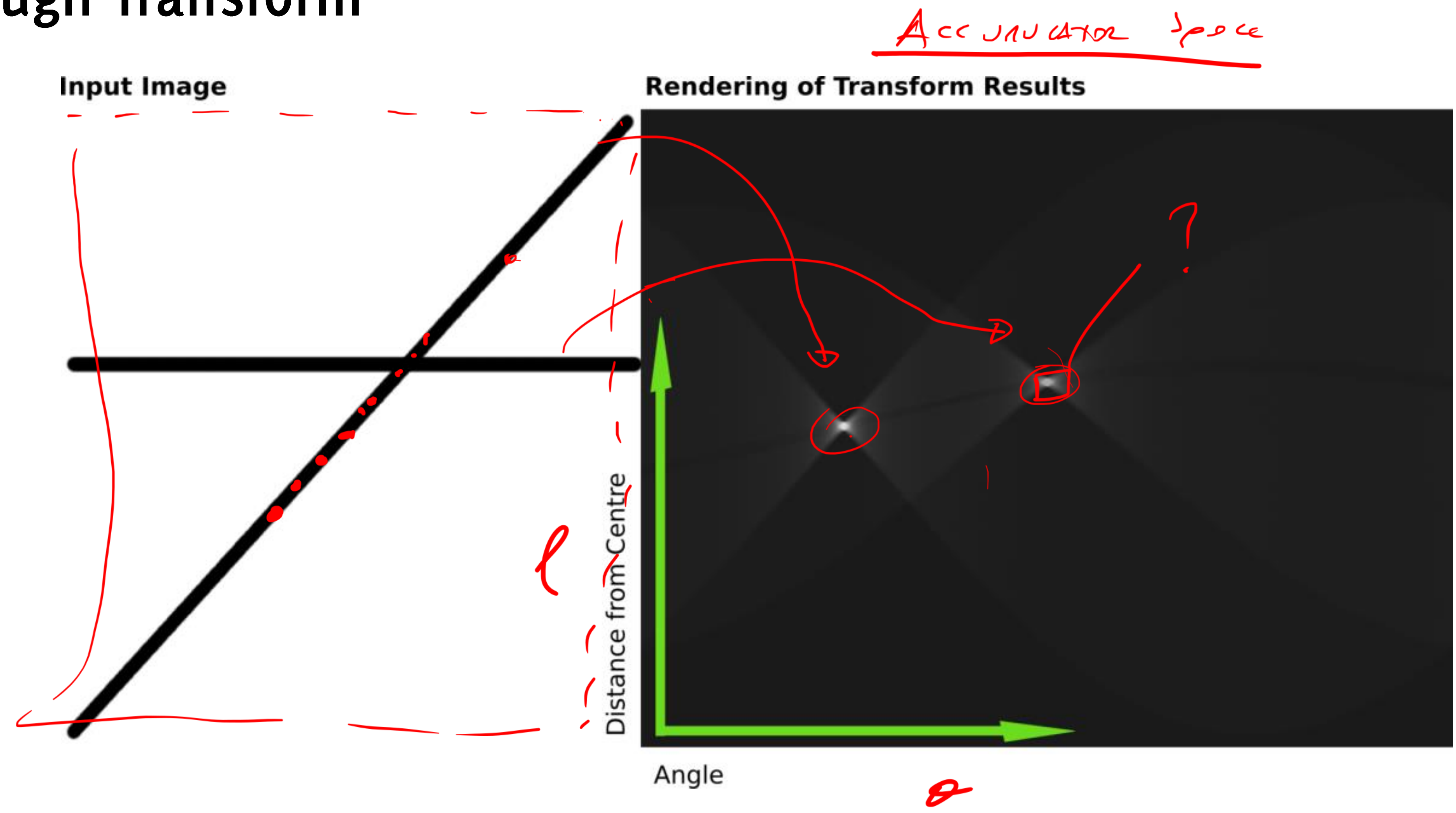
Hough Transform: the algorithm

```
Initialize H[rho,θ]=0

for each edge point (x,y) in the image:

  for θ in range(θmin,θmax):

    rho = x cos(θ) - y sin(θ)

    H[rho,θ]+=1

Find the value(s) of (d,θ) where H[d,θ] is maximum

The detected line in the image is given by
d = x cos(θ) - y sin(θ)
```

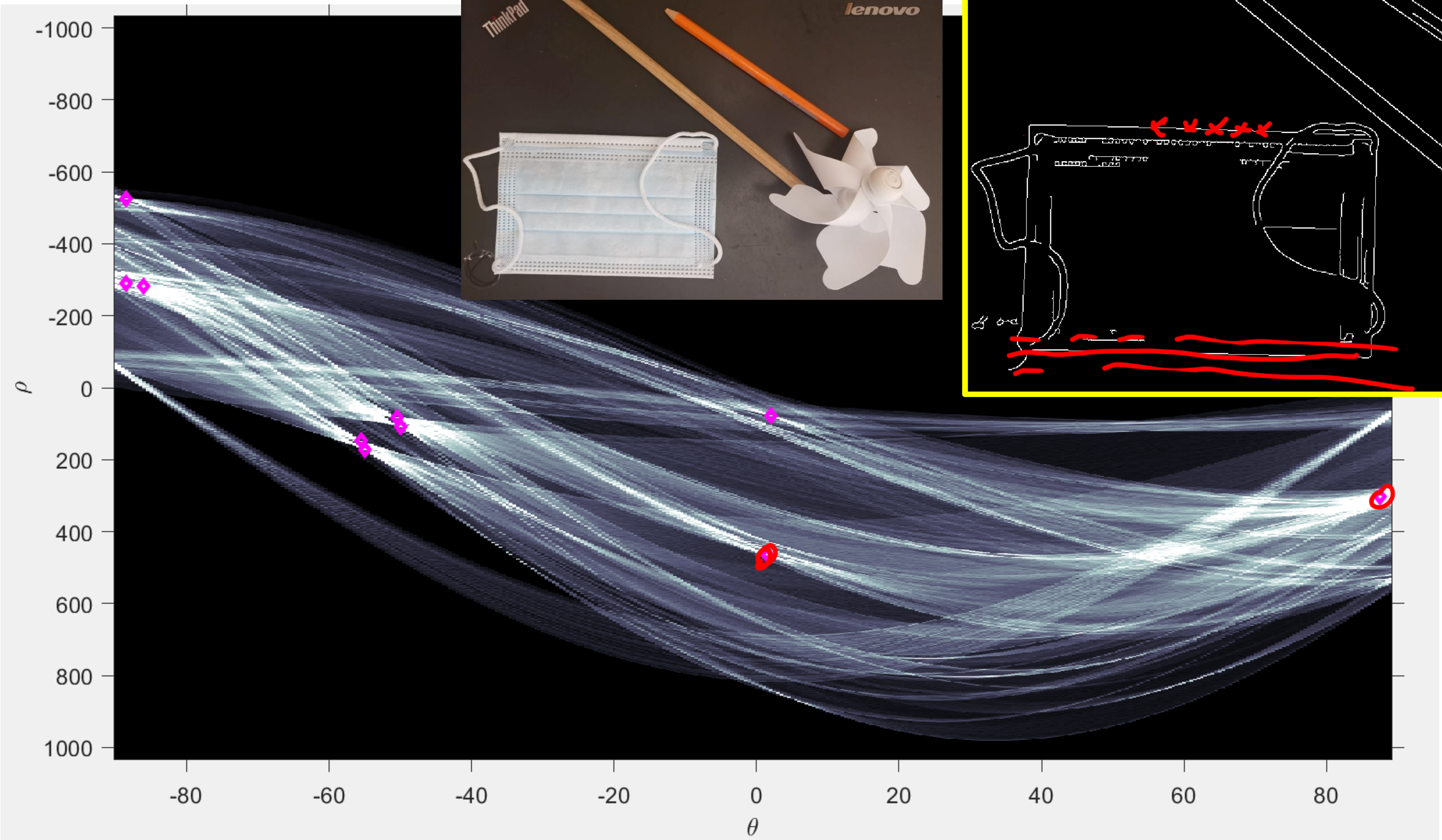


Slide credit: S. Seitz

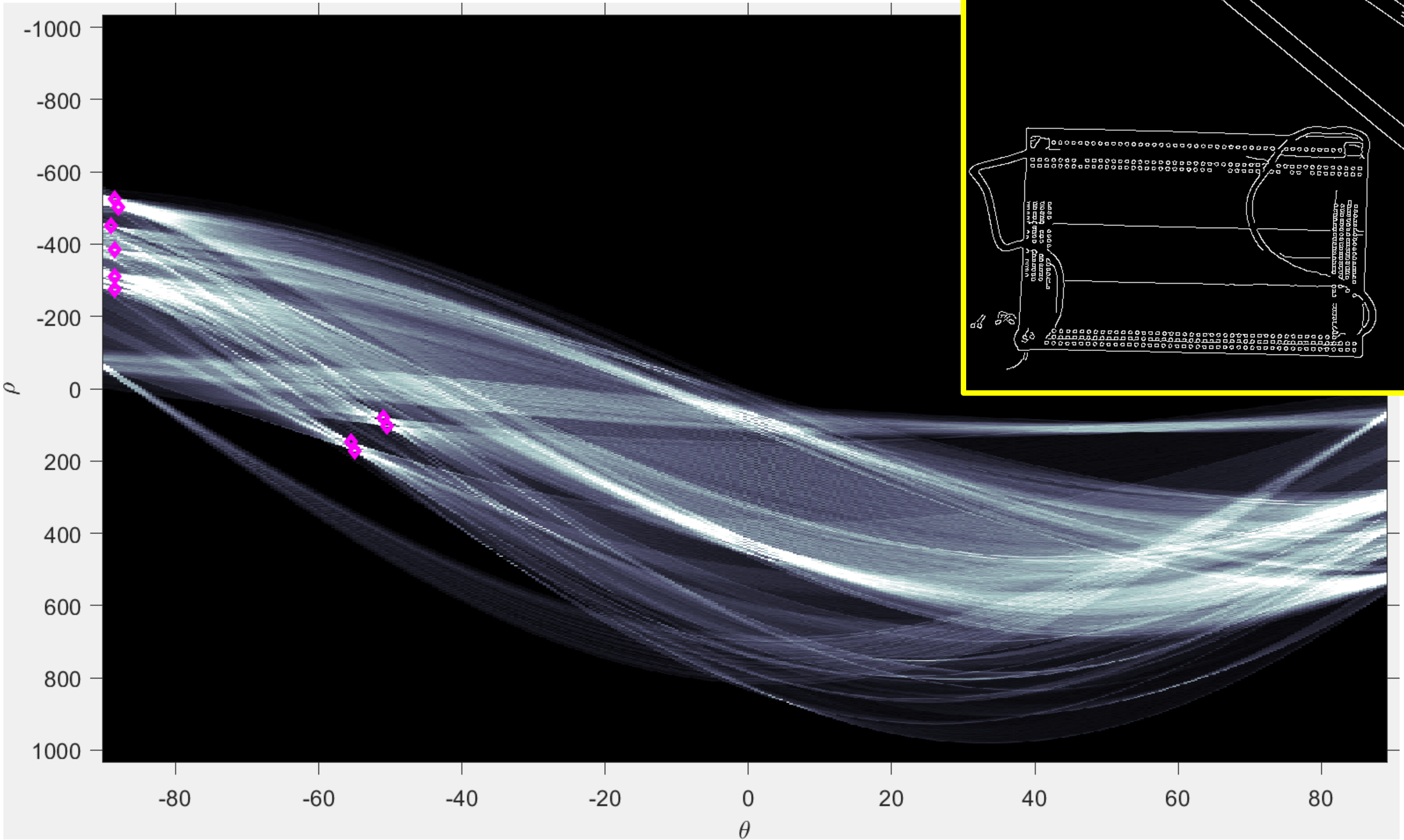
Hough Transform

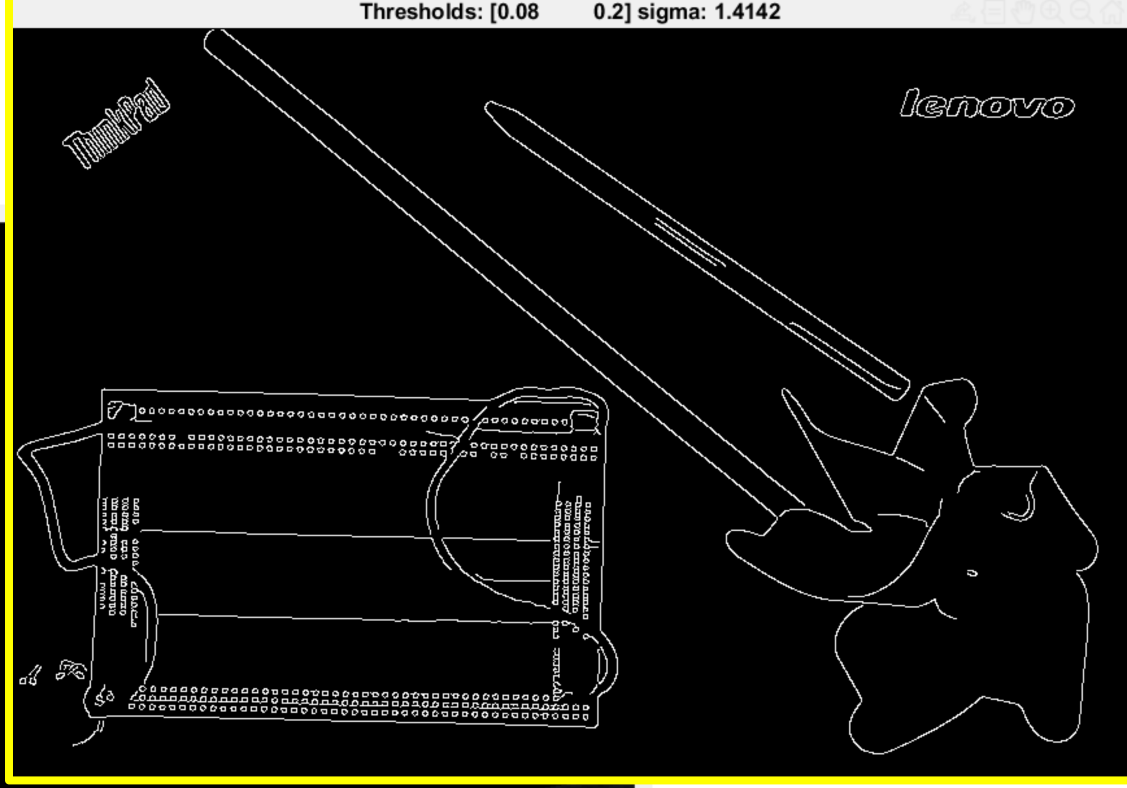


Where is the facemask in H?

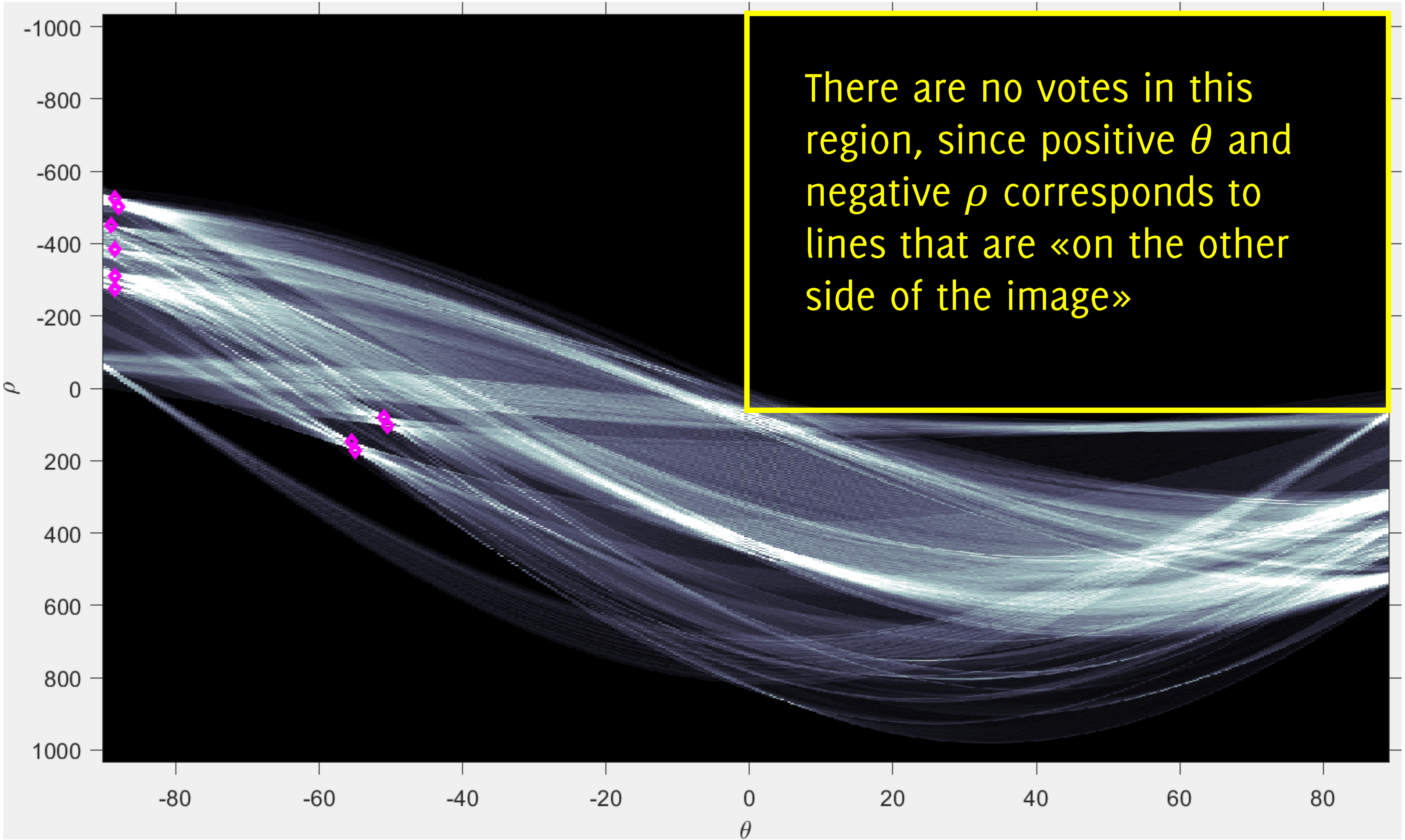


What if we take more edges?





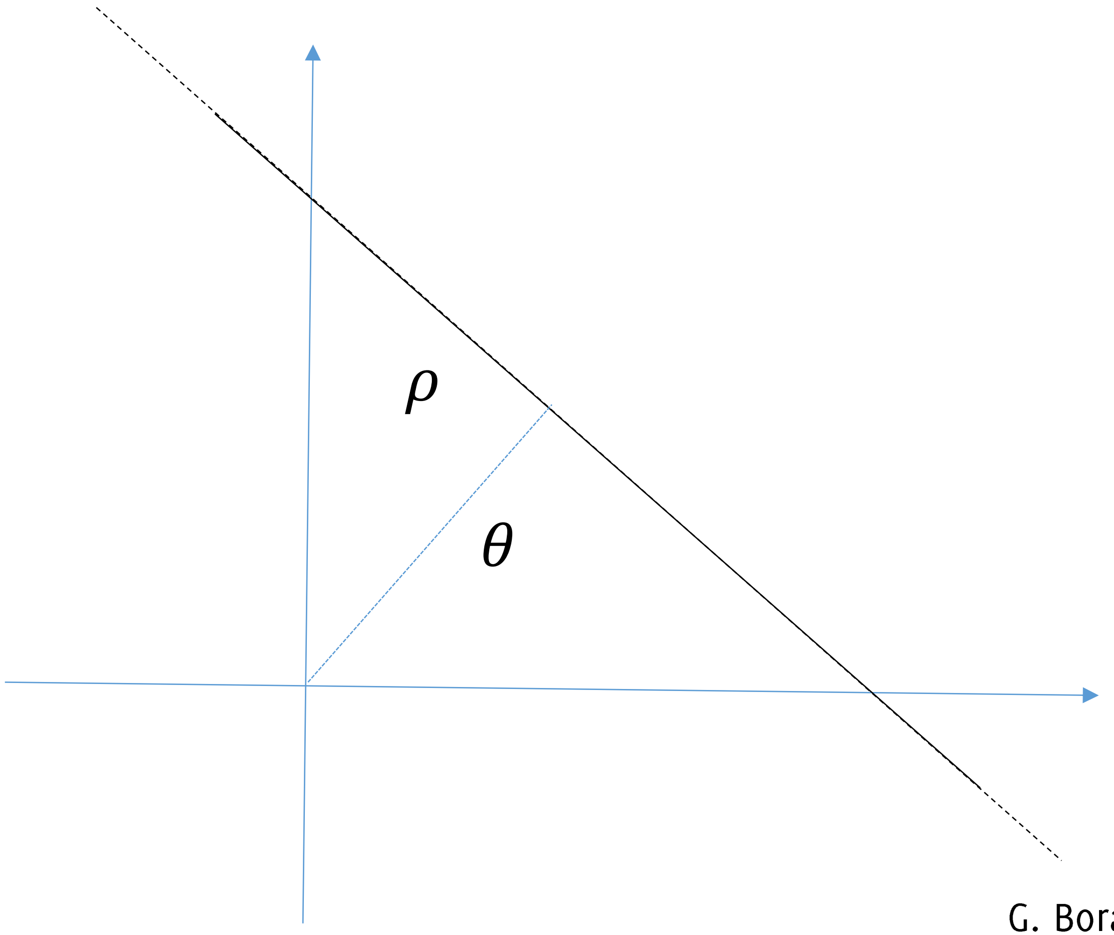
What if we take more edges?



G. Boracchi

Size of the Accumulator Space

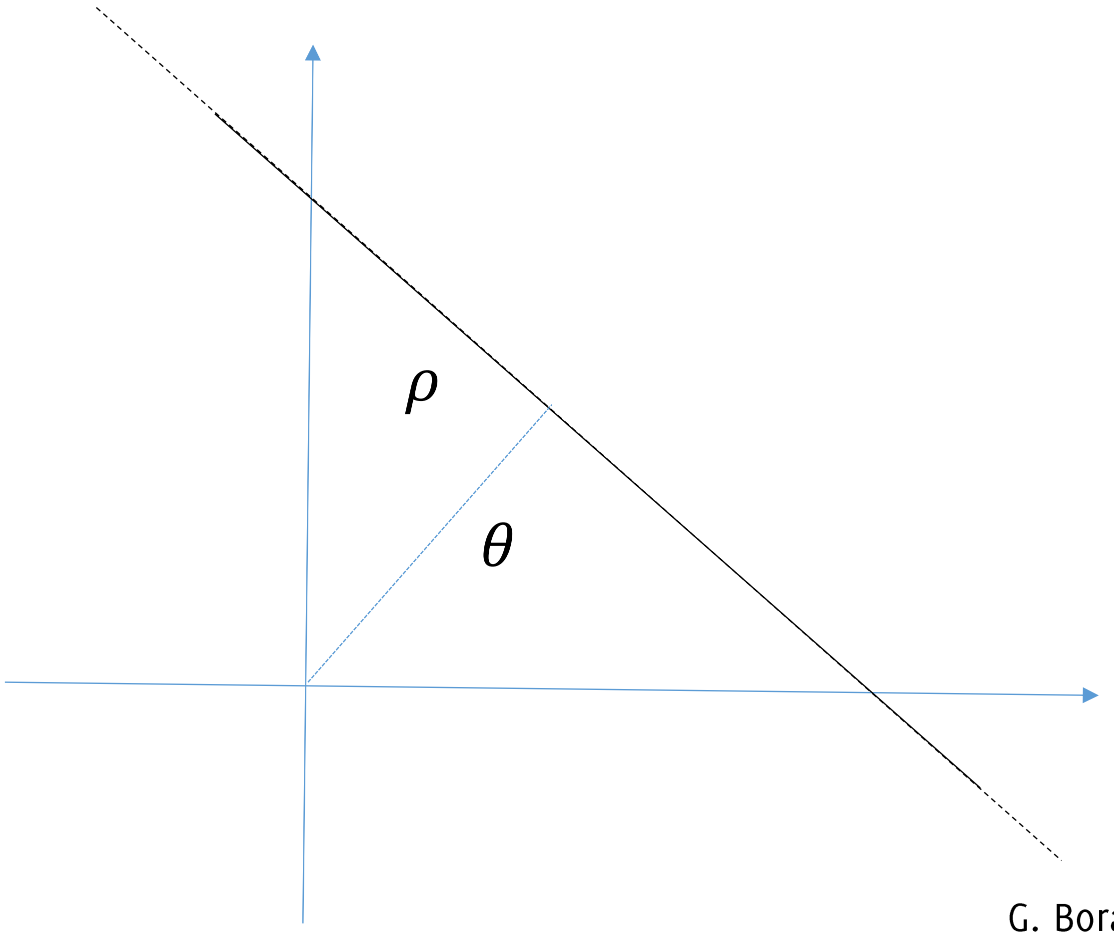
What are the maximum sizes of the accumulator space to represent any line intersecting the $H \times W$ image?



Size of the Accumulator Space

What are the maximum sizes of the accumulator space to represent any line intersecting the $H \times W$ image?

It is the diagonal, so $L = \sqrt{H^2 + W^2}$



Bin size in the accumulator: an important parameter

How large are the bins in the accumulator?

- Too small: many weak peaks due to noise
- Just right: one strong peak per line, despite noise
- Too large:
  - poor accuracy in locating the line
  - many votes from clutter might end up in the same bin

A solution:

- keep bin size small but also vote for neighbors in the accumulator (this is the same as “smoothing” the accumulator image)

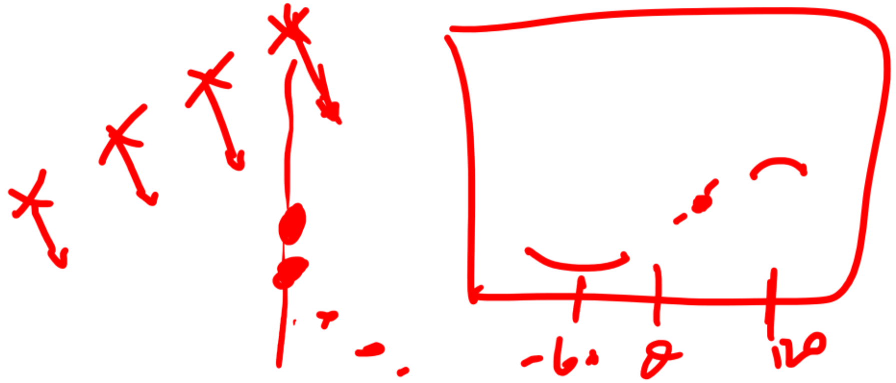
Extension

From the edge detection algorithm, we know the direction of the gradient for each edge pixel

Remember how that **edge direction is orthogonal to gradient direction**

We can enforce that **an edge pixel votes only for those lines** that have (almost) direction parallel of the edge (i.e. orthogonal to gradient)!

- Reduces the computation time
- Reduces the number of useless votes (better visibility of maxima corresponding to real lines)



Hough Transform

The approach is not only limited to lines, but rather to any parametric model that we are able to fit

- Circles can be fit in a 3d accumulator space

It is quite robust to noise

Hough Transform For Circles

slide Credits Alessandro Giusti, USI

Hugh Transform for Circles

1. Every **edge point** casts votes for all **circles** that are compatible with it
2. We choose **circles** that accumulated a lot of votes

How do we parametrize circles?

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

Center $(x = a, y = b)$ and radius r : 3 degrees of freedom

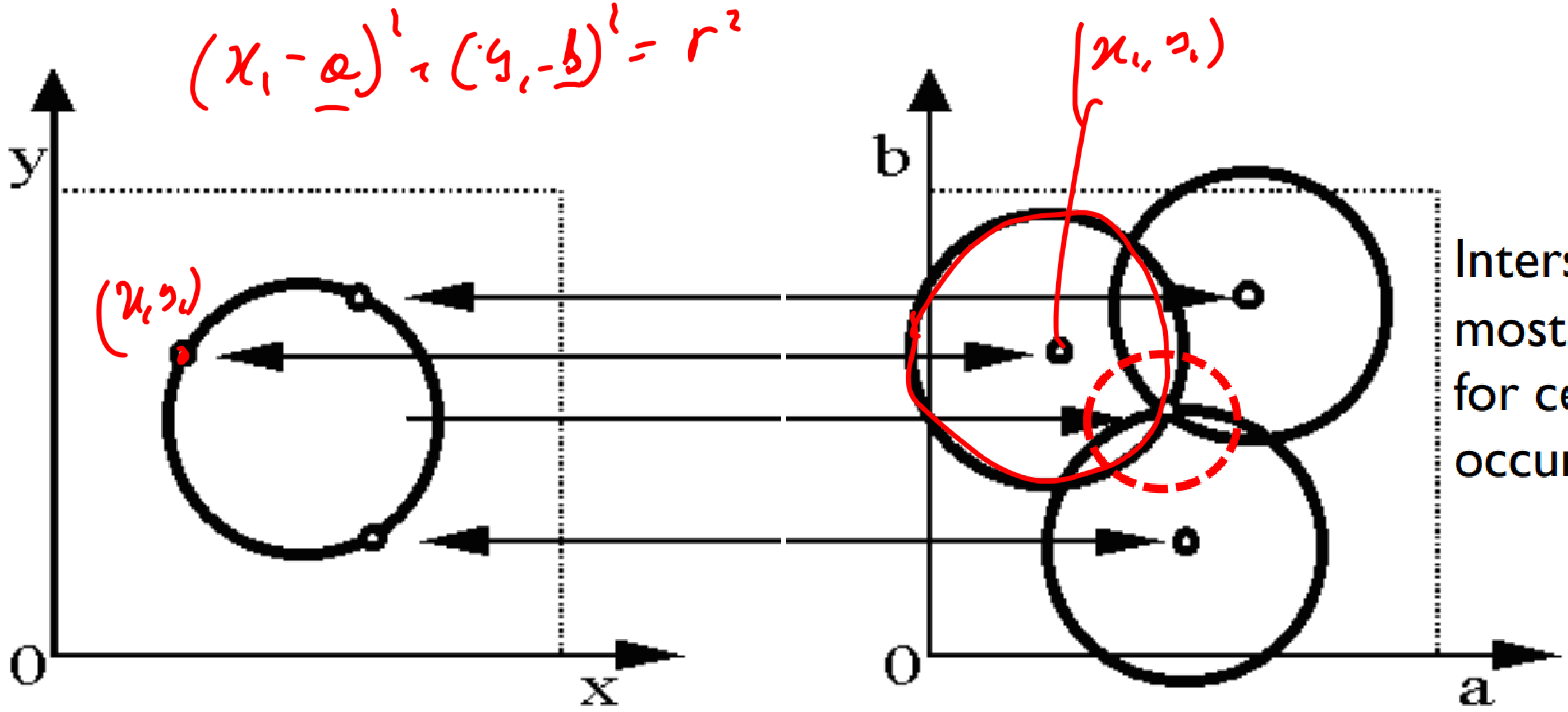
If we assume r known, the Hough space is 2D:

- a: x coordinate of circle center
- b: y coordinate of circle center

The role of (a, b) and (x, y) are interchangeable, thus:

One point in image space maps to a circle in Hough space

Hough space for circles with known radius



Intersection: most votes for center occur here.

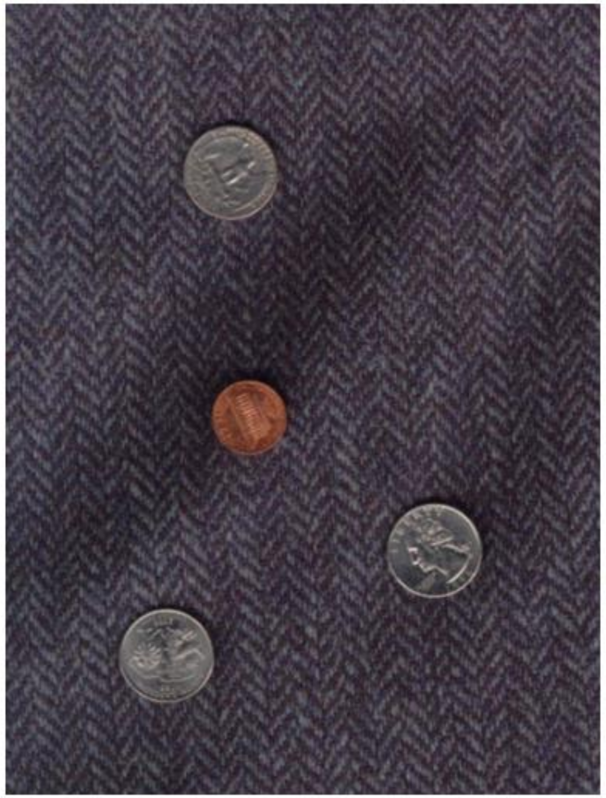
Hugh Transform for Circles

```
Initialize H accumulator to zeros
For every edge pixel (x,y):
  For each possible radius value r:
    For each possible gradient direction θ:
      a = x – r cos(θ) // column
      b = y + r sin(θ) // row
      H[a,b,r] += 1
```

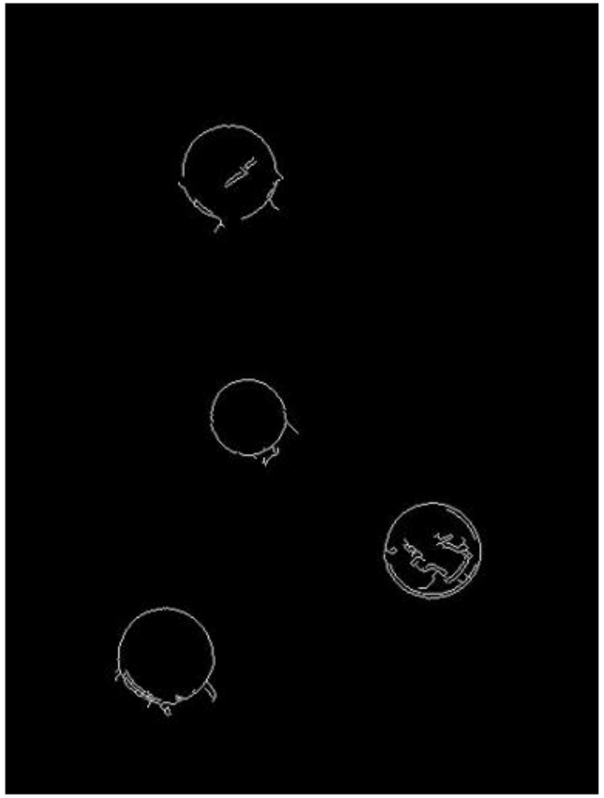
An example

Accumulator for radius equal to radius of a penny

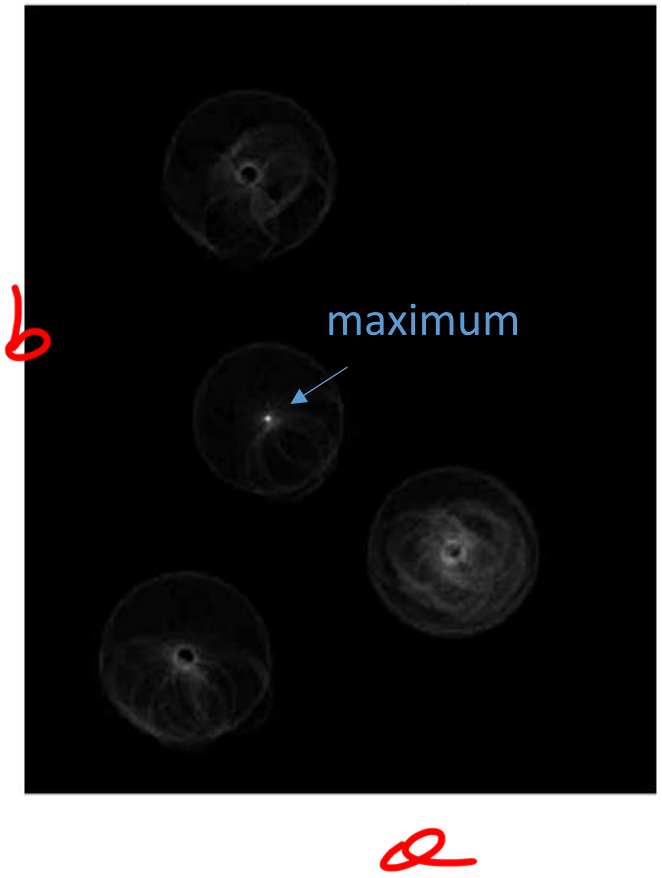
Image



Edges



Accumulator for radius=penny

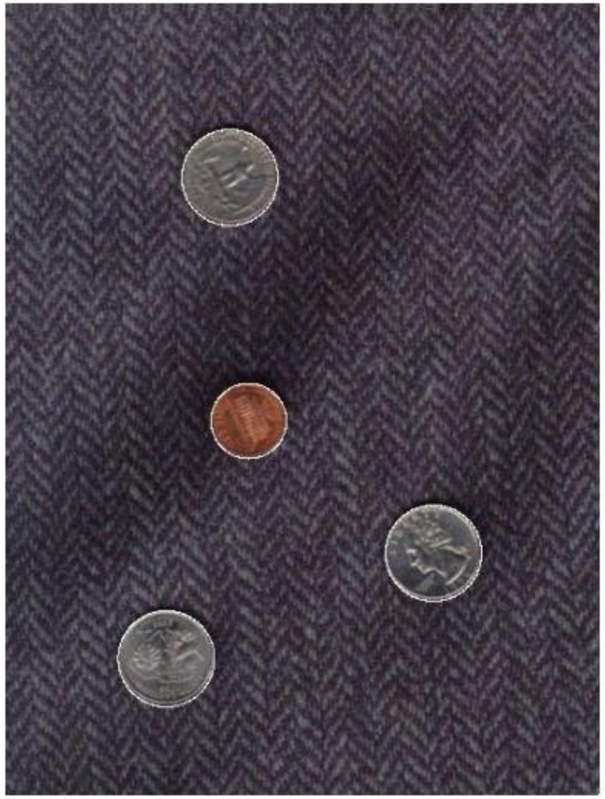


Slide credit: K. Grauman

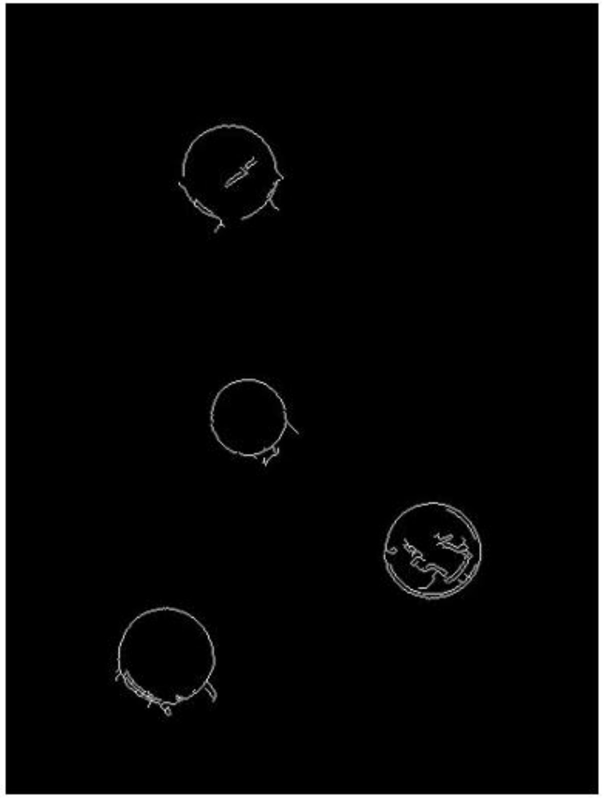
An example

Accumulator for radius equal to radius of a quarter

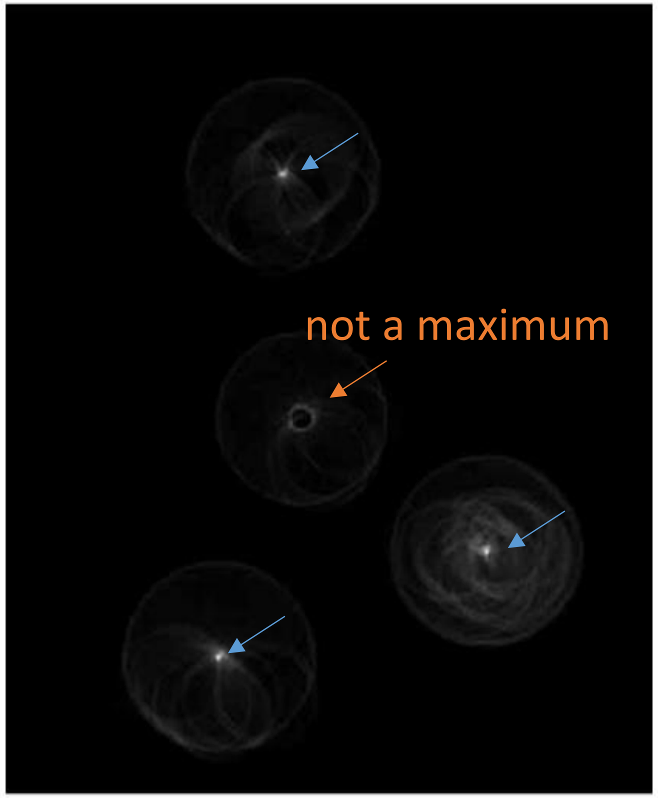
Image



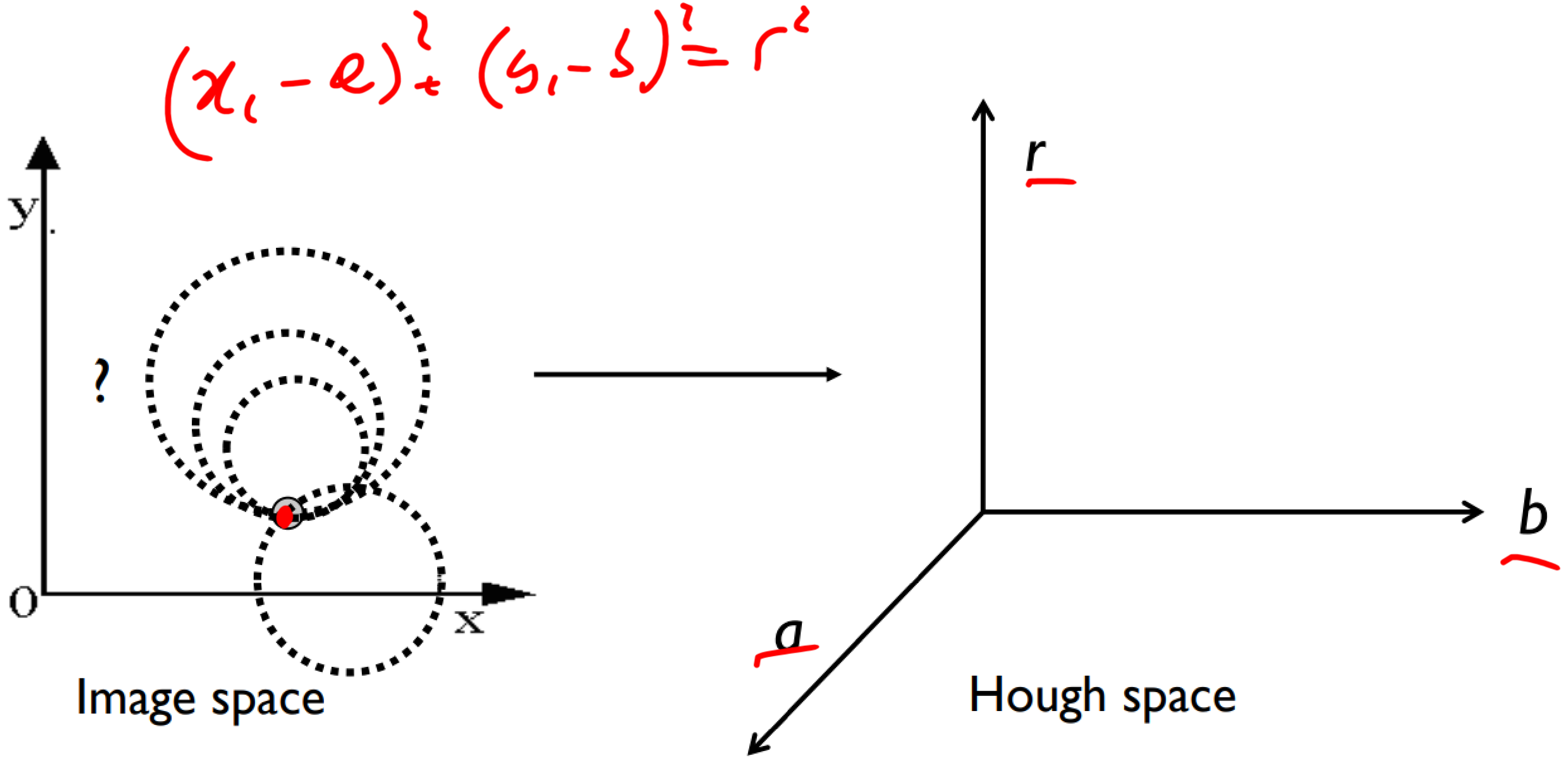
Edges



Accumulator for radius=quarter



Hough space for circles with unknown radius



One point in image space maps to…
a cone in Hough space

Hough space for circles with unknown radius

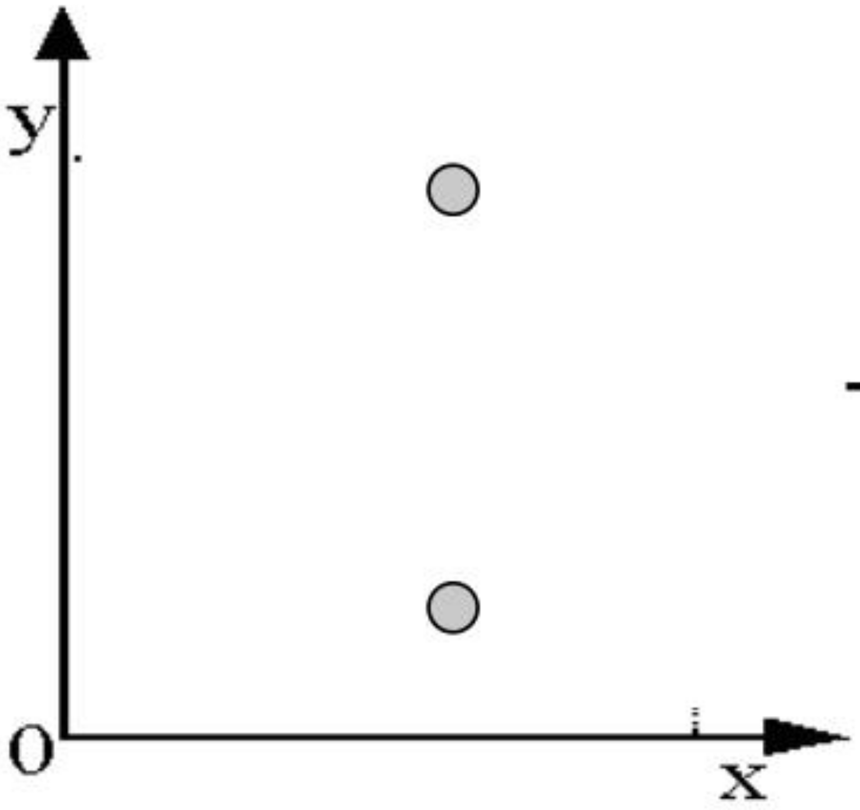
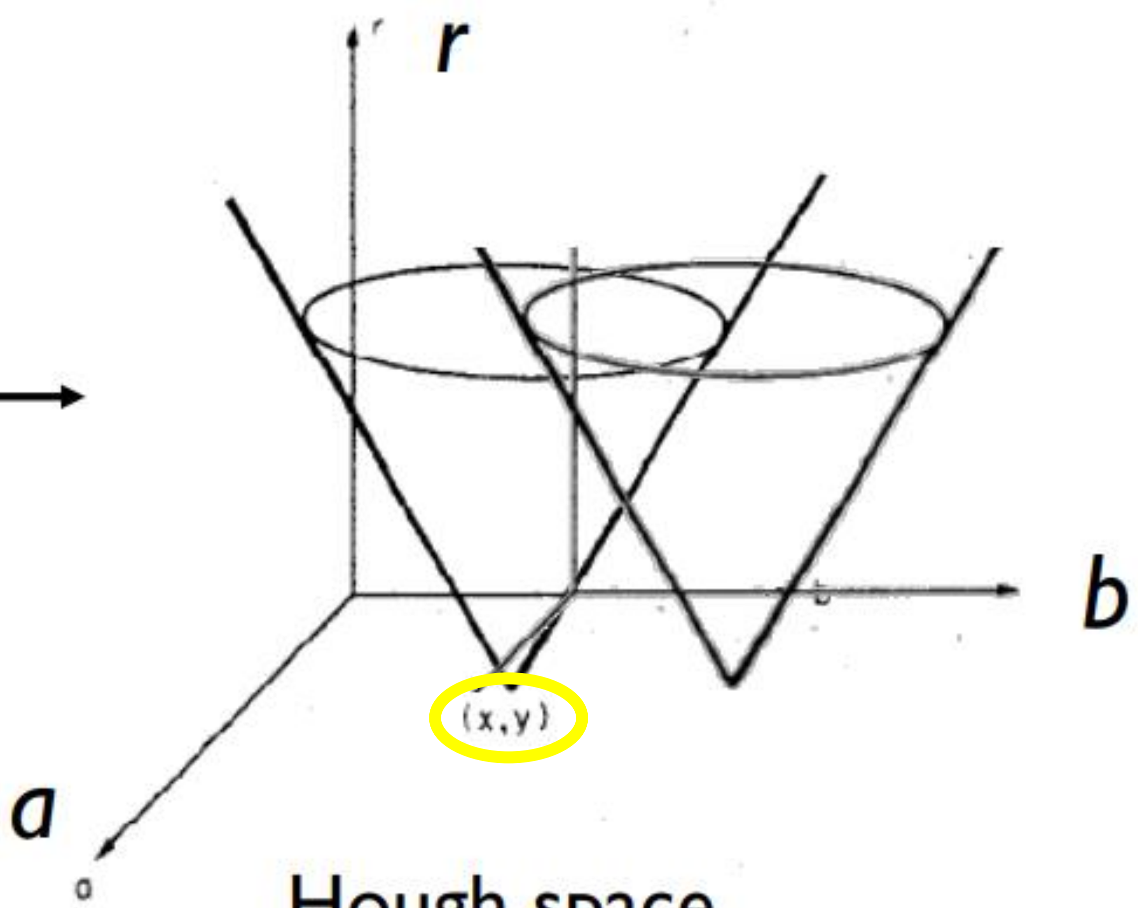


Image space



Hough space

When the radius is zero $(a, b) = (x, y)$

Slide credit: K. Grauman

If we know the gradient direction...

When increasing the radius, the center can only live in a line, thus there is a linear relation between a, b

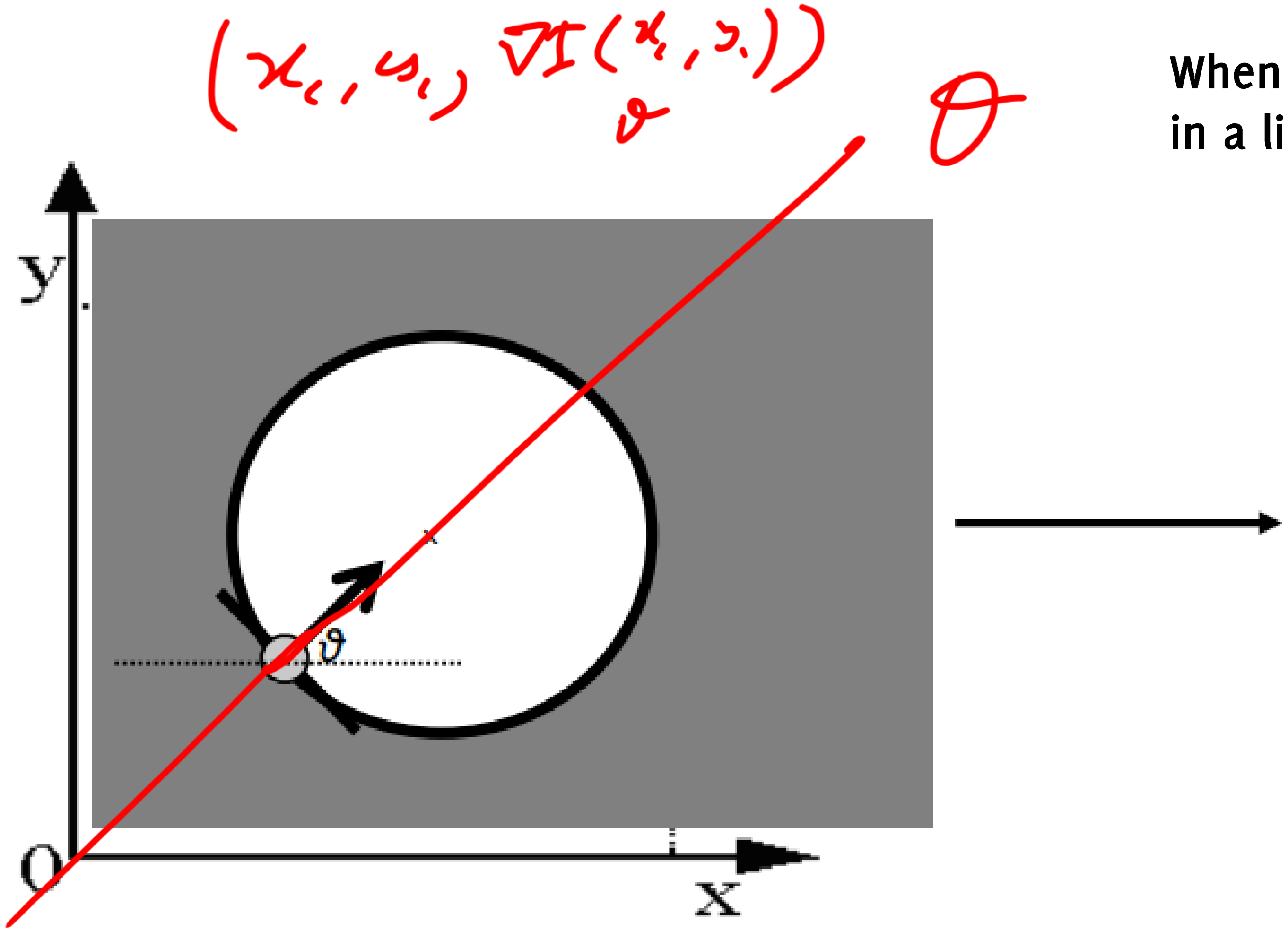
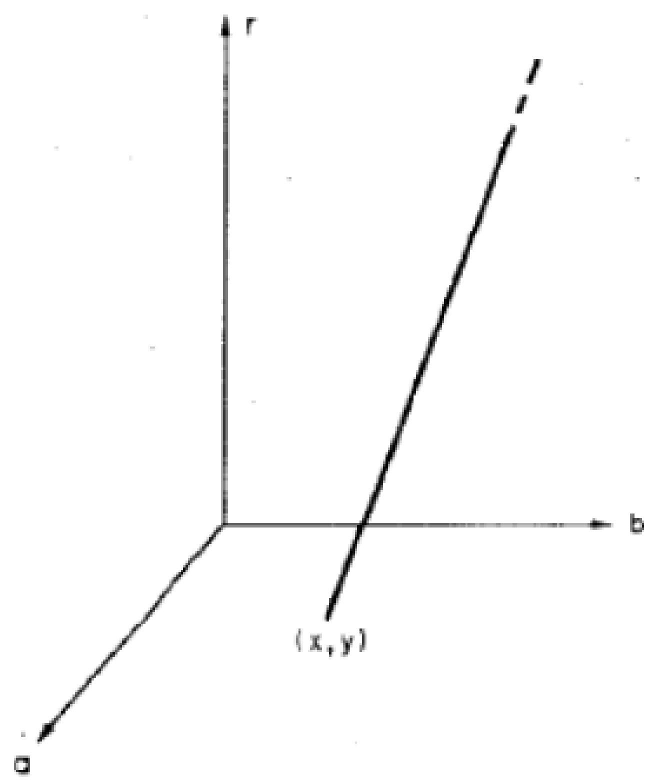


Image space



Hough space

Conclusions

Advantages

- All points are processed independently, so **the algorithm can cope with occlusions and gaps**
- Voting algorithms are **robust to clutter**, because points not corresponding to any model are unlikely to contribute consistently to any single bin
- Can detect **multiple instances of a model** in a single pass

Disadvantages

- Only suitable for models with **few parameters**
- Must filter out spurious peaks in hough accumulator
- Quantization of hough space is tricky

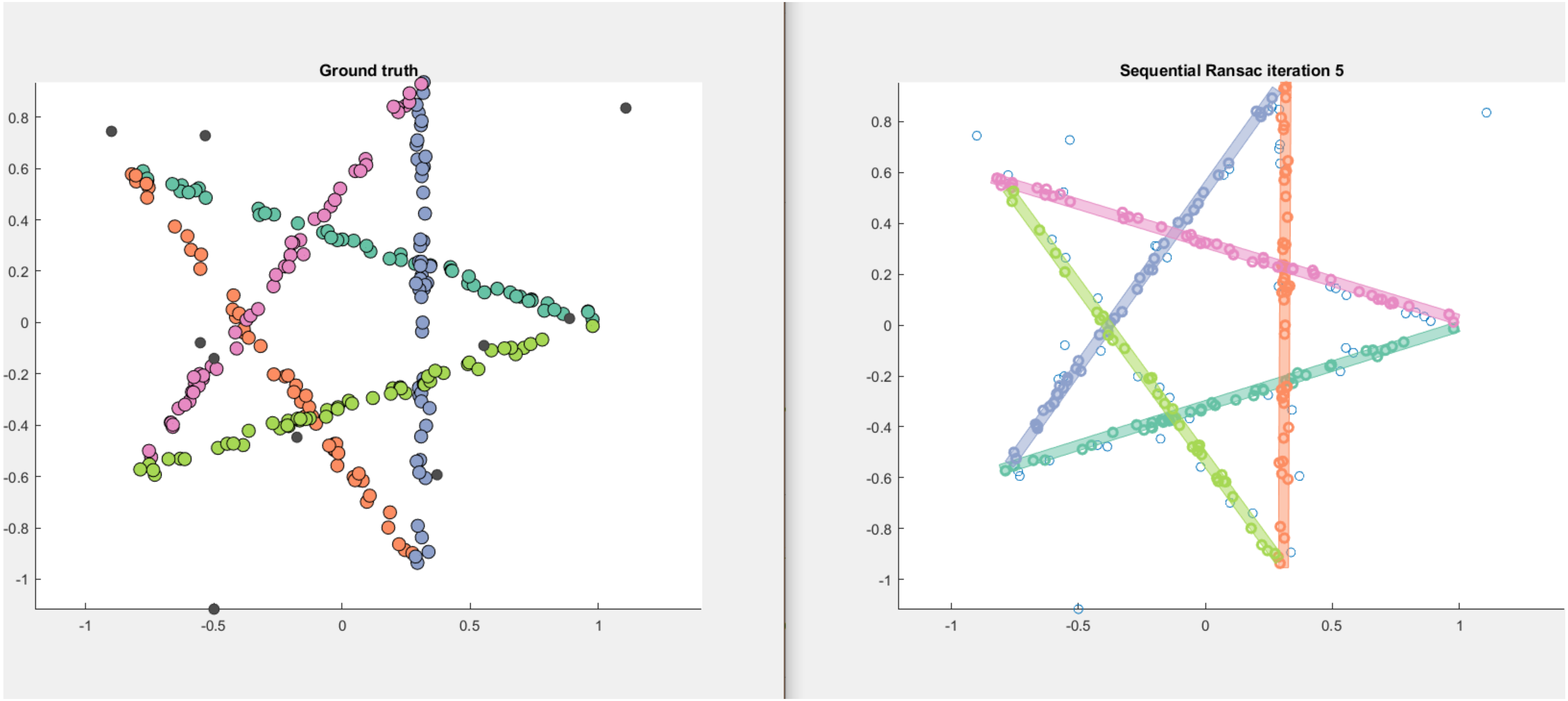
Assigments

demo_robustmmf_TODO

1. Implement Sequantial Ransac for line fitting over
   1. the star5 data,
   2. The stair4 data

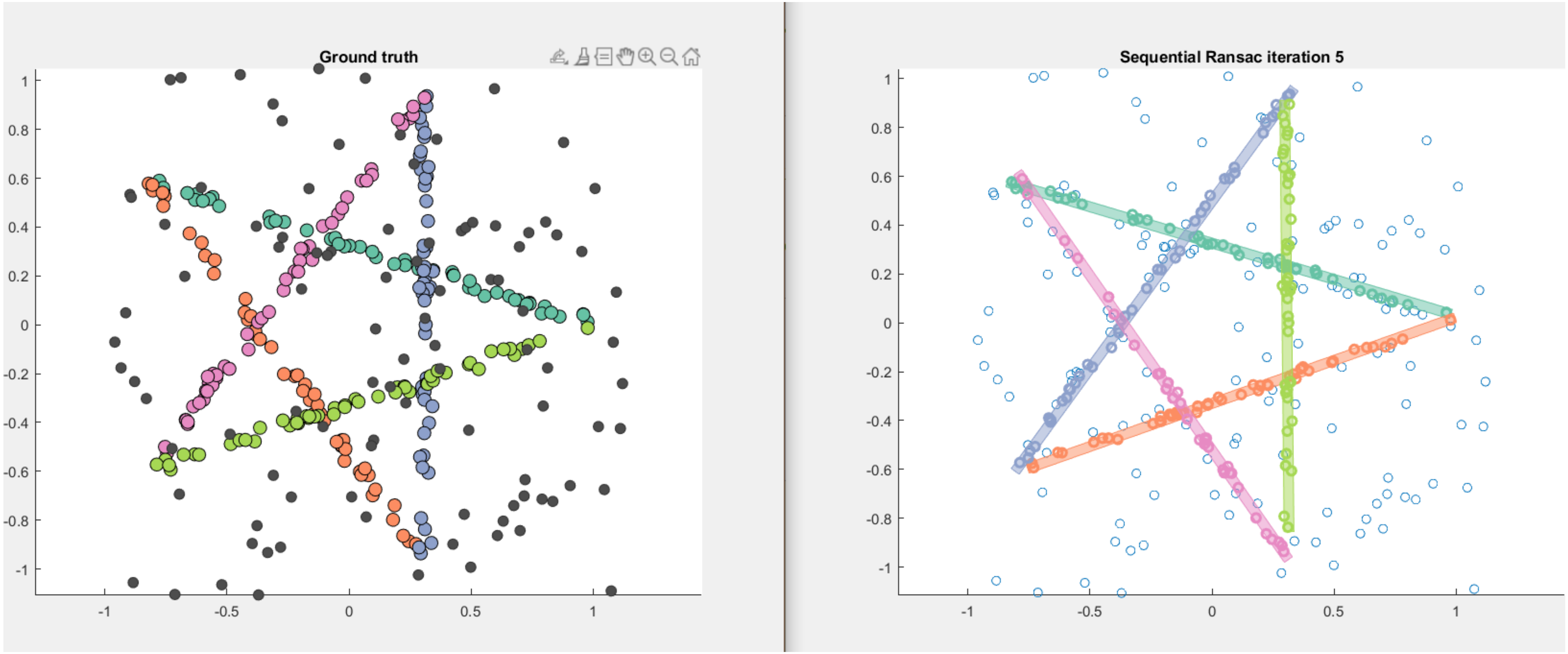
   introducing different amount of outliers
2. Check the limitations of sequential ransac and test different stopping criteria (number of models retrieved, minimum consensus of the last model found)
3. Implement Ransac (thus run sequential Ransac) to fit circles

Only 10 outliers



G. Boracchi

100 outliers



G. Boracchi

The stairs case.. Sequential ransac terribly fails