



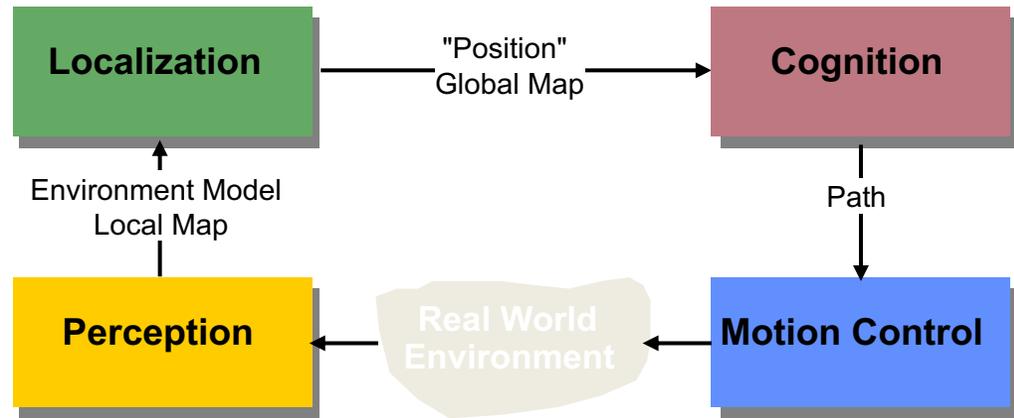
上海科技大学
ShanghaiTech University

CS283: Robotics Spring 2026: Perception

Sören Schwertfeger

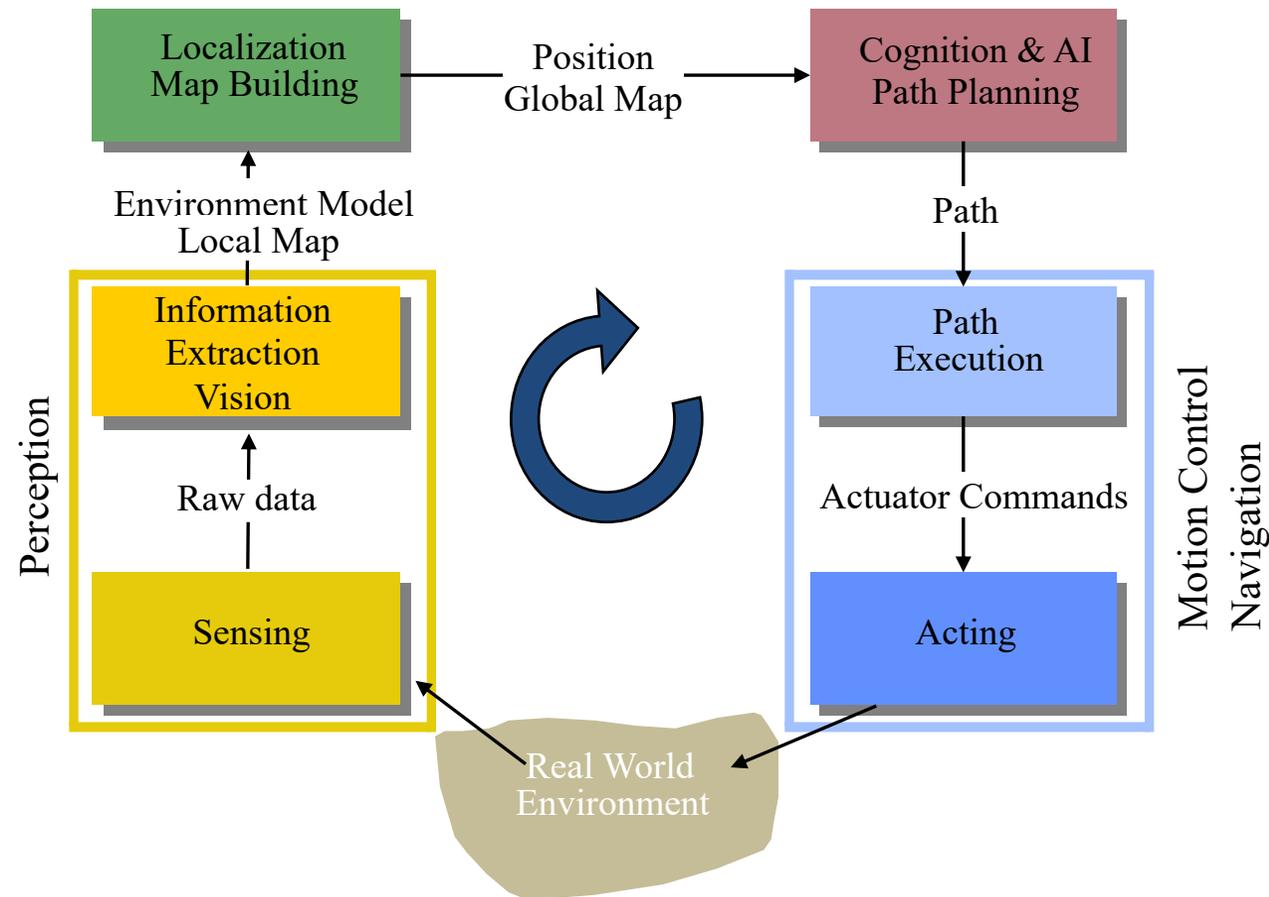
ShanghaiTech University

PERCEPTION



Line extraction from laser scans
Vision

General Control Scheme for Mobile Robot Systems



LINE EXTRACTION

Split and merge

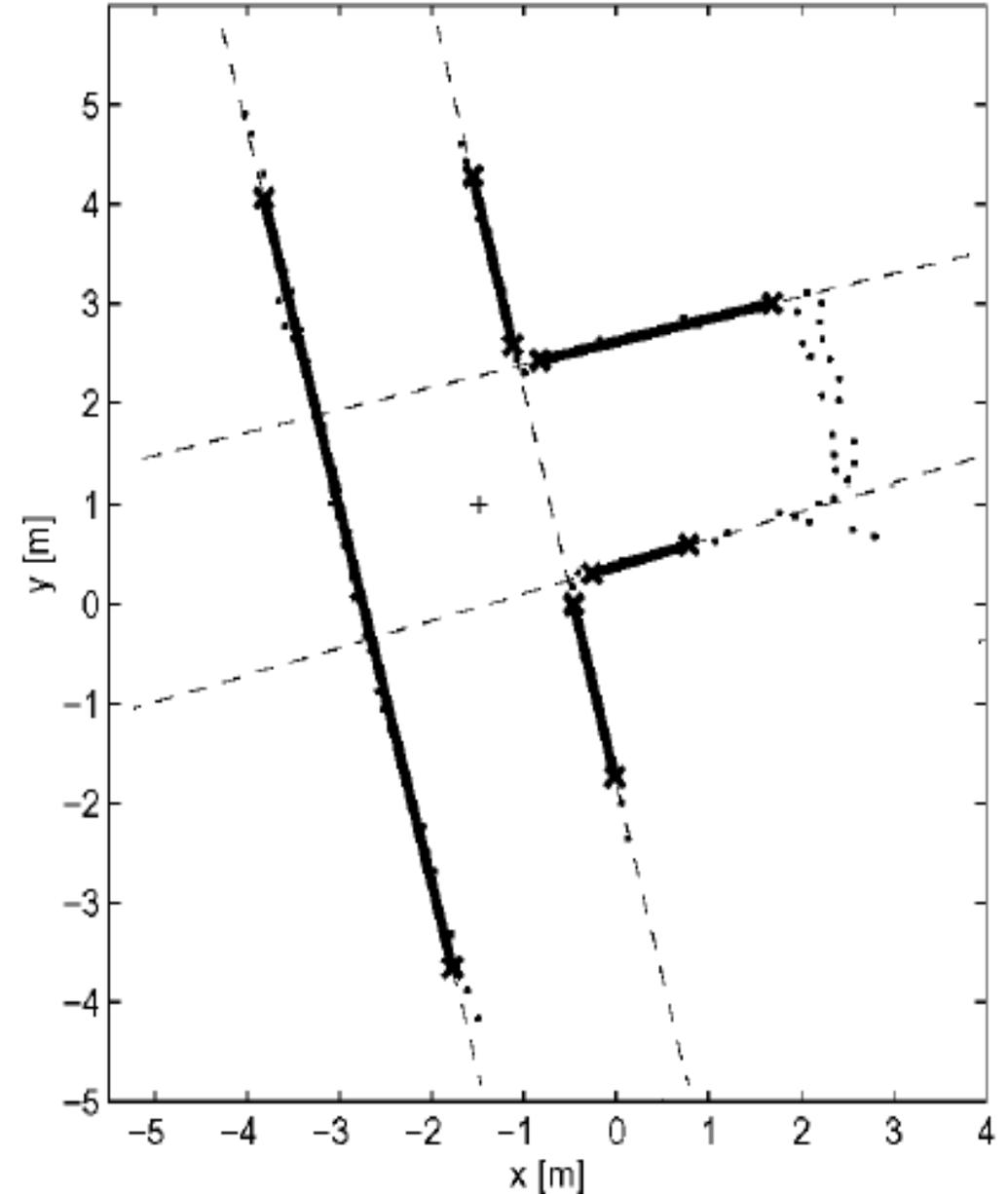
Linear regression

RANSAC

Hough-Transform

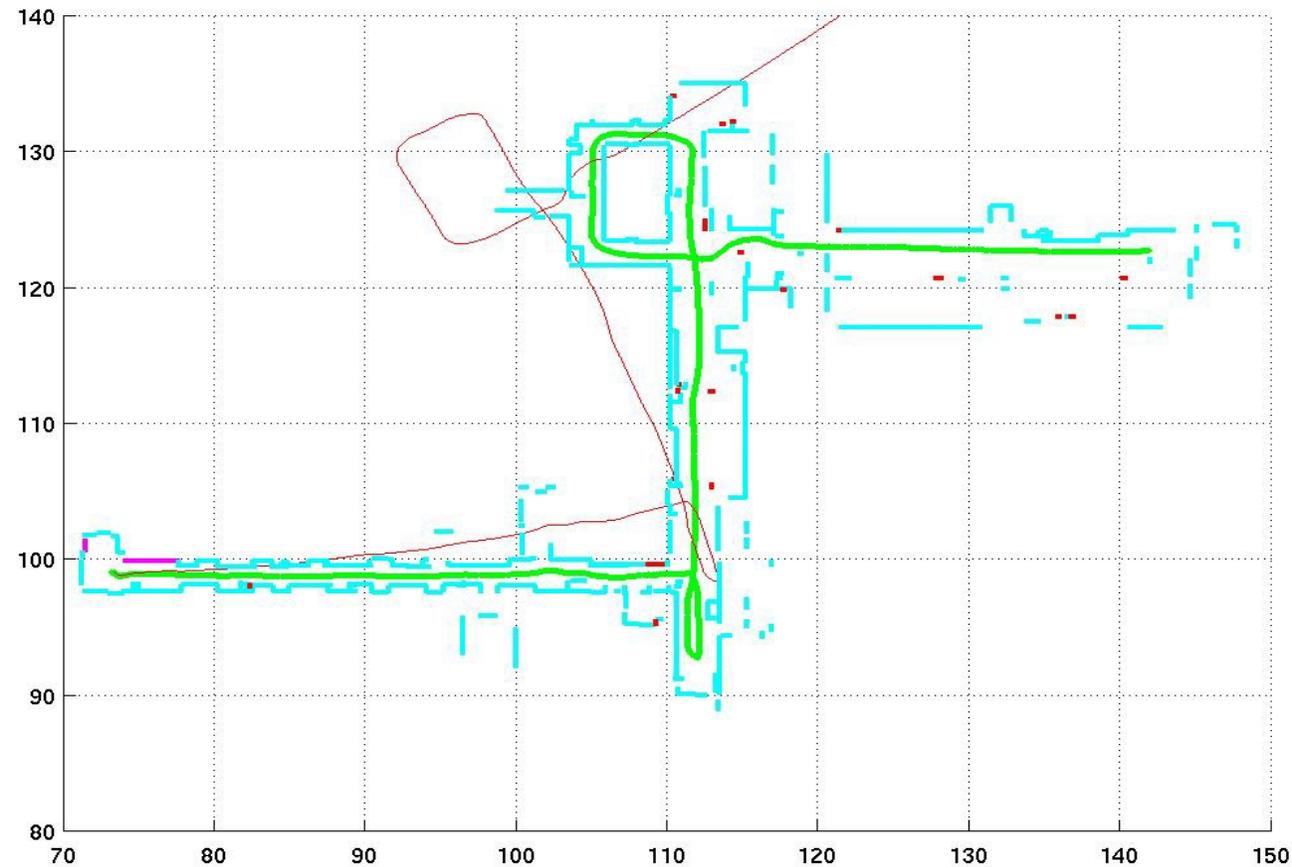
Line Extraction: Motivation

- Laser Range Scan
 - Example: 360 deg – black points
 - Example: dashed lines: desired line extractions
 - Use detected lines for:
 - Scan registration (find out transform between frames of two consecutive LRF scans – change due to robot motion)
- OR
- Mapping using line representation



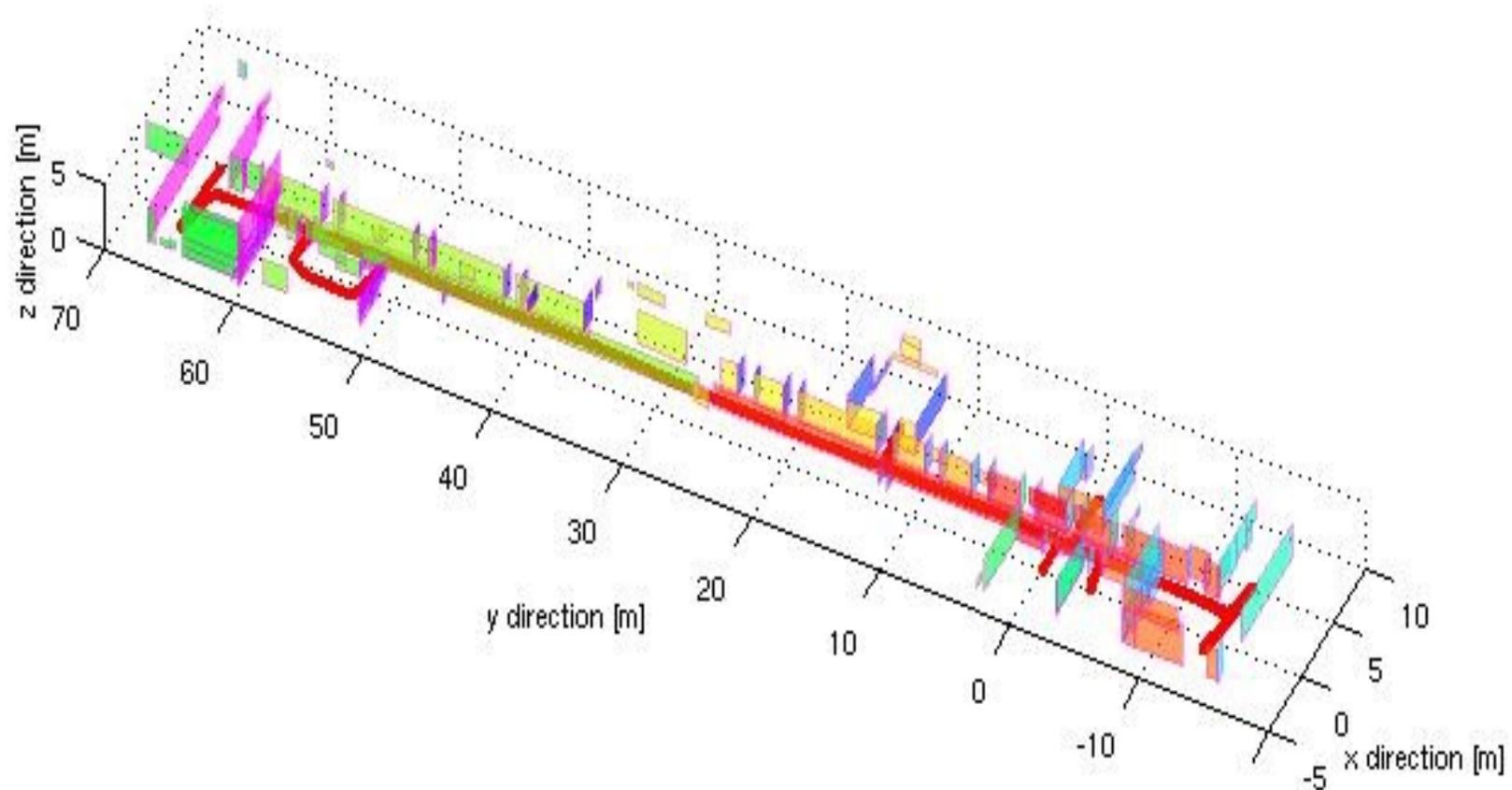
Line Extraction: Motivation

- Map of hallway built using line segments



Line Extraction: Motivation

- Map of the hallway built using orthogonal planes constructed from line segments

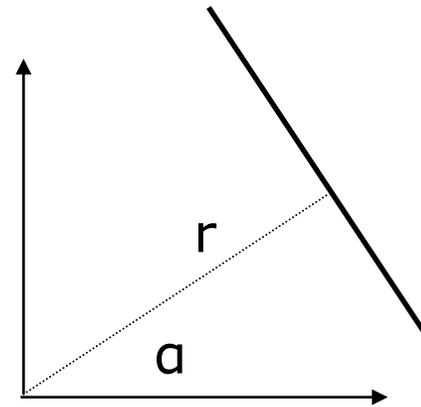
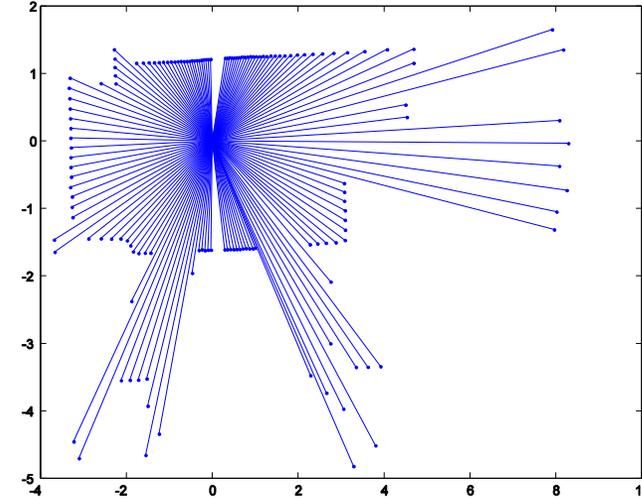


Line Extraction: Motivation

- Why laser scanner:
 - Dense and accurate range measurements
 - High sampling rate, high angular resolution
 - Good range distance and resolution.
- Why line segment:
 - The simplest geometric primitive
 - Compact, requires less storage
 - Provides rich and accurate information
 - Represents most office-like environment.

Line Extraction: The Problem

- Scan point in polar form: (ρ_i, θ_i)
- Assumptions:
 - Gaussian noise
 - Negligible angular uncertainty
- Line model in polar form:
 - $x \cos \alpha + y \sin \alpha = r$
 - $-\pi < \alpha \leq \pi$
 - $r \geq 0$

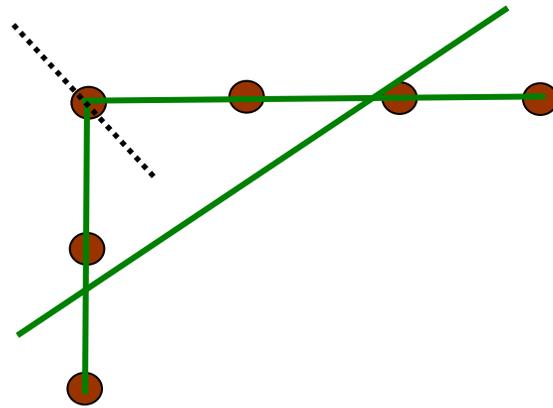


Line Extraction: The Problem (2)

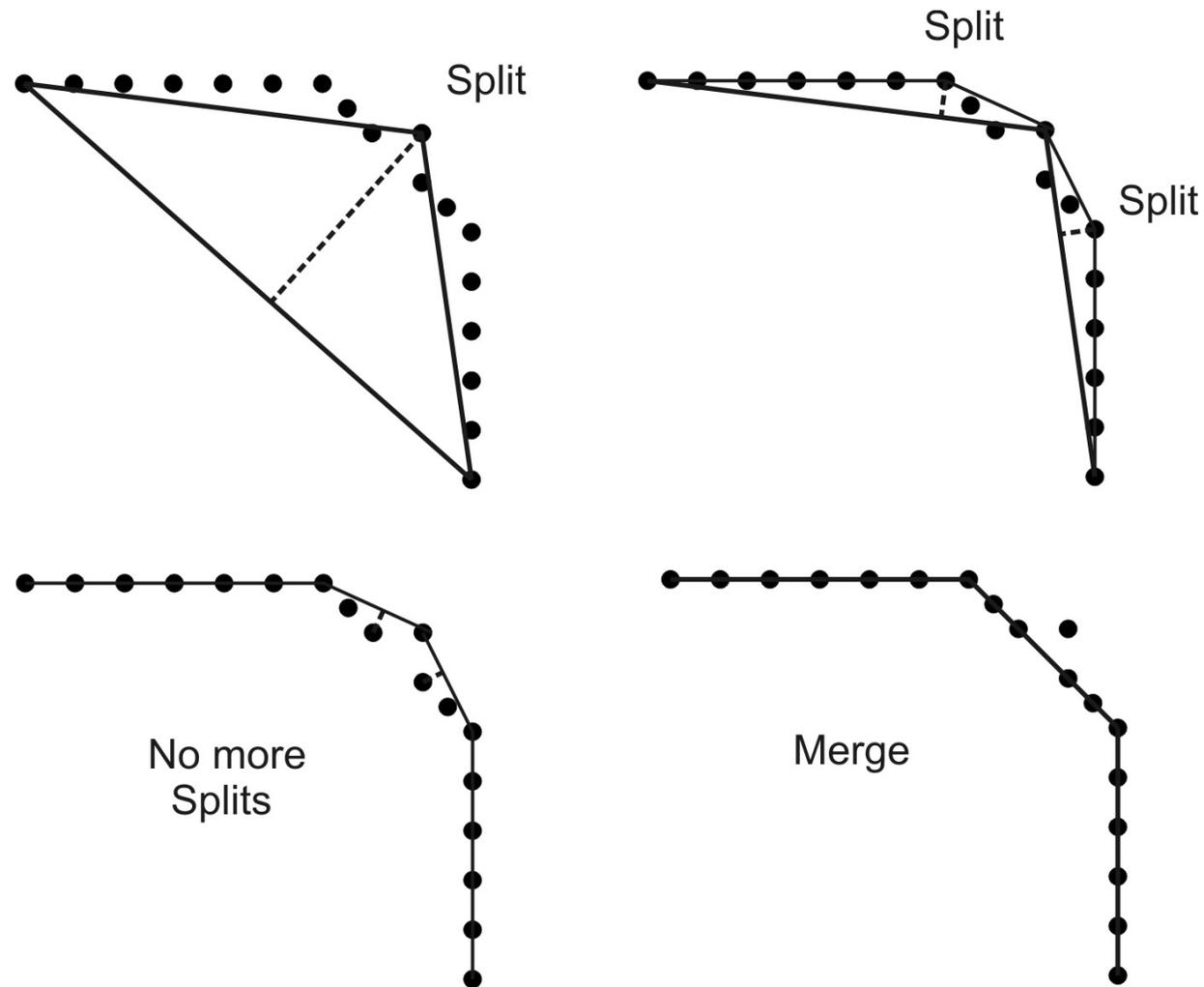
- Three main problems:
 - How many lines ?
 - Which points belong to which line ?
 - This problem is called SEGMENTATION
 - Given points that belong to a line, how to estimate the line parameters ?
 - This problem is called LINE FITTING
- The Algorithms we will see:
 - 1.Split and merge
 - 2.Linear regression
 - 3.RANSAC
 - 4.Hough-Transform

Algorithm 1: Split-and-Merge (standard)

- The most popular algorithm which is originated from computer vision.
- A recursive procedure of fitting and splitting.
- A slightly different version, called Iterative-End-Point-Fit, simply connects the end points for line fitting.



Algorithm 1: Split-and-Merge (Iterative-End-Point-Fit)

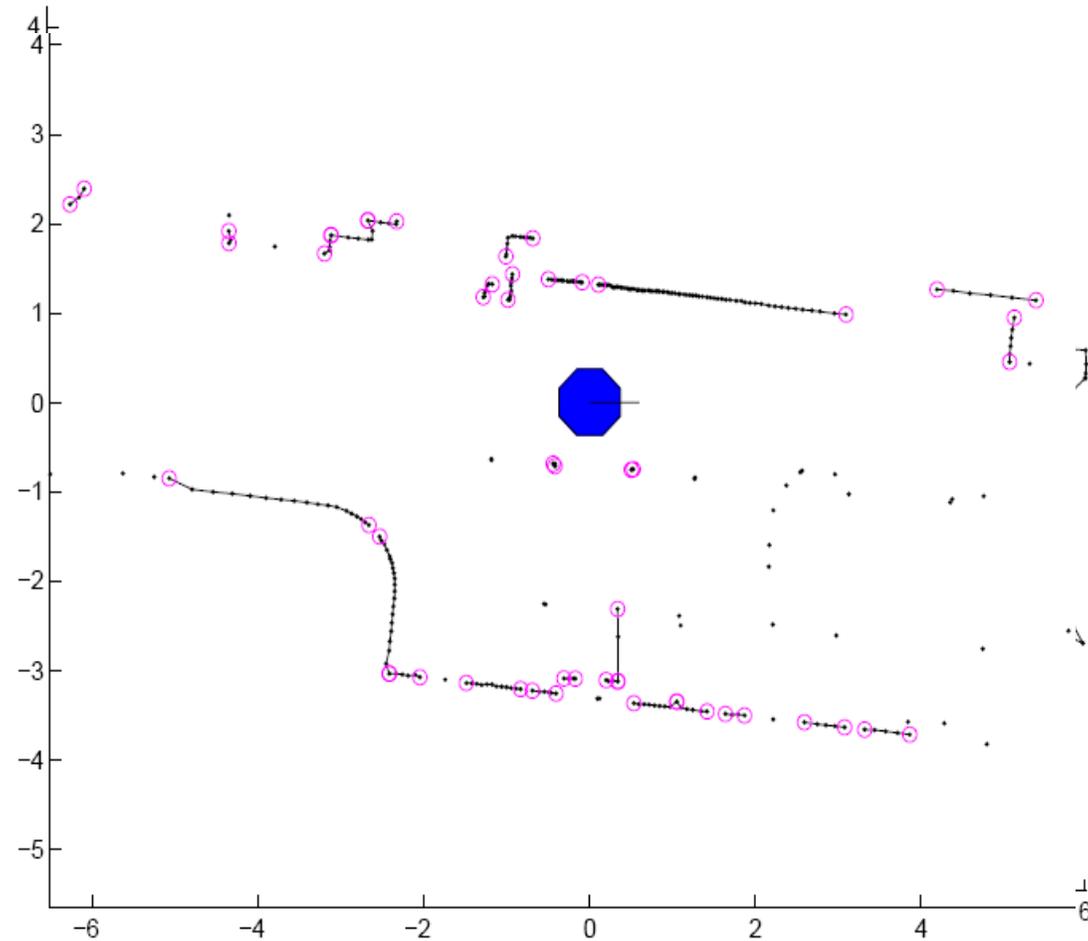


Algorithm 1: Split-and-Merge

Algorithm 1: *Split-and-Merge*

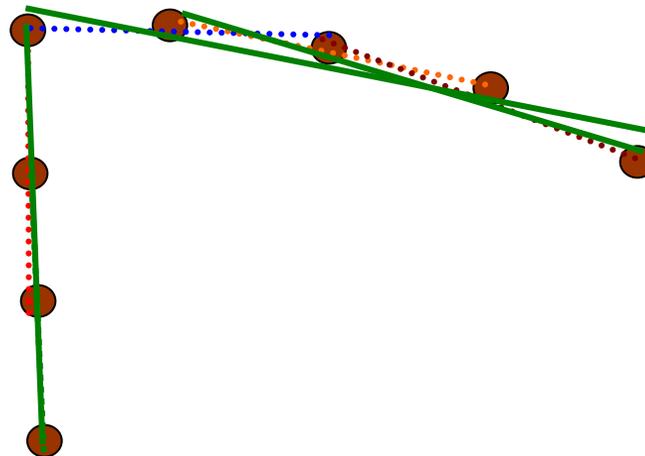
1. Initial: set s_1 consists of N points. Put s_1 in a list L
 2. Fit a line to the next set s_i in L
 3. Detect point P with maximum distance d_P to the line
 4. If d_P is less than a threshold, continue (go to step 2)
 5. Otherwise, split s_i at P into s_{i1} and s_{i2} , replace s_i in L by s_{i1} and s_{i2} , continue (go to 2)
 6. When all sets (segments) in L have been checked, merge collinear segments.
-

Algorithm 1: Split-and-Merge: Example application



Algorithm 2: Line-Regression

- Uses a “sliding window” of size Nf
- The points within each “sliding window” are fitted by a segment
- Then adjacent segments are merged if their line parameters are close



$Nf = 3$

Algorithm 2: Line-Regression

Algorithm 2: *Line-Regression*

1. Initialize sliding window size N_f
 2. Fit a line to every N_f consecutive points (a window)
 3. Compute a line fidelity array, each is the sum of Mahalanobis distances between every three adjacent windows
 4. Construct line segments by scanning the fidelity array for consecutive elements having values less than a threshold, using an AHC algorithm
 5. Merge overlapped line segments and recompute line parameters for each segment
-

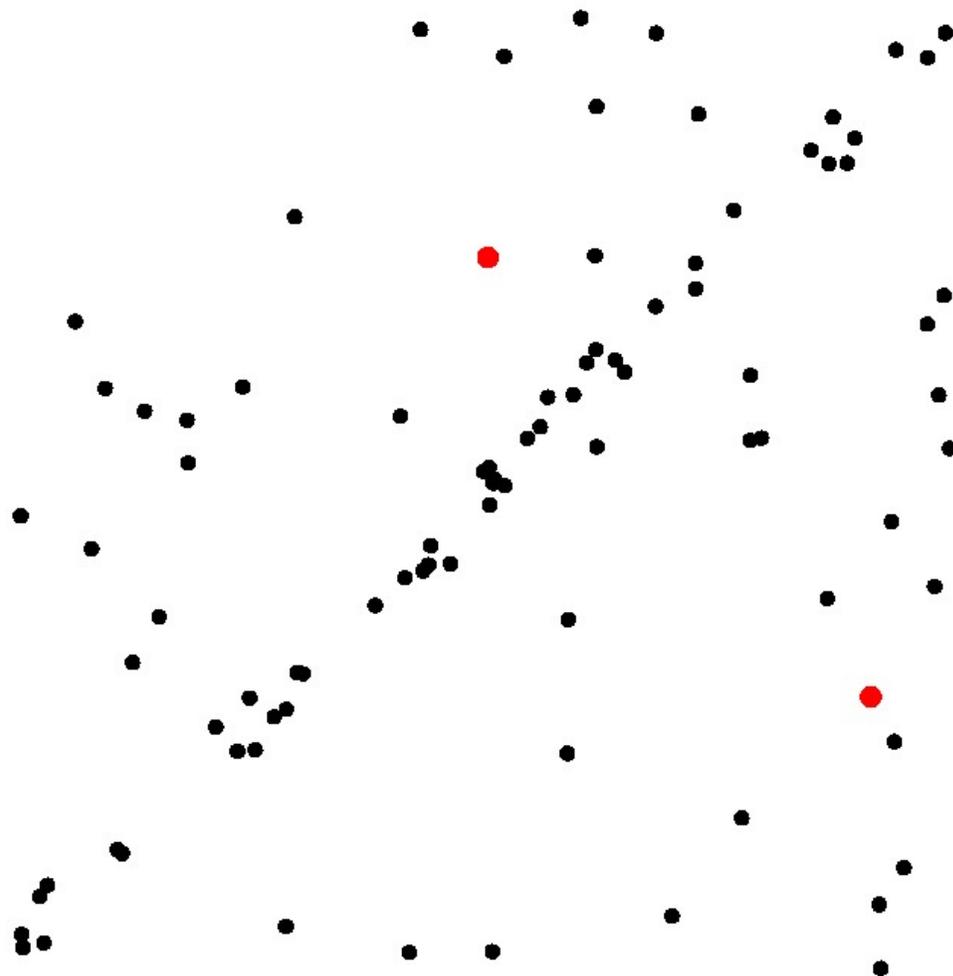
Algorithm 3: RANSAC

- Acronym: **Random Sample Consensus.**
- Generic & robust fitting algorithm of models with outliers
 - Outliers: points which do not satisfy a model
- RANSAC: apply to any problem where:
 - identify the inliers
 - which satisfy a predefined mathematical model.
- Typical robotics applications:
 - line extraction from 2D range data (sonar or laser);
 - plane extraction from 3D range data
 - structure from motion
- RANSAC:
 - iterative method & non-deterministic
- Drawback: A nondeterministic method, results are different between runs.

Algorithm 3: RANSAC

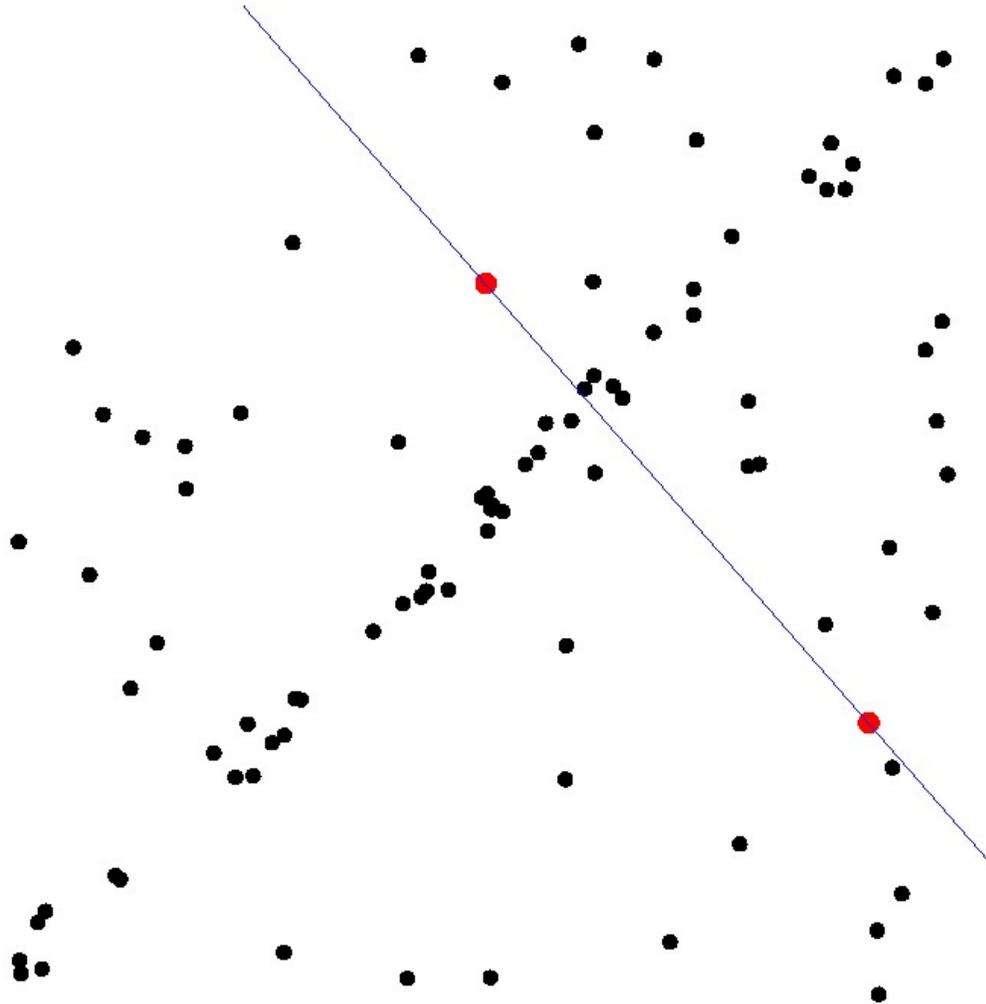


Algorithm 3: RANSAC



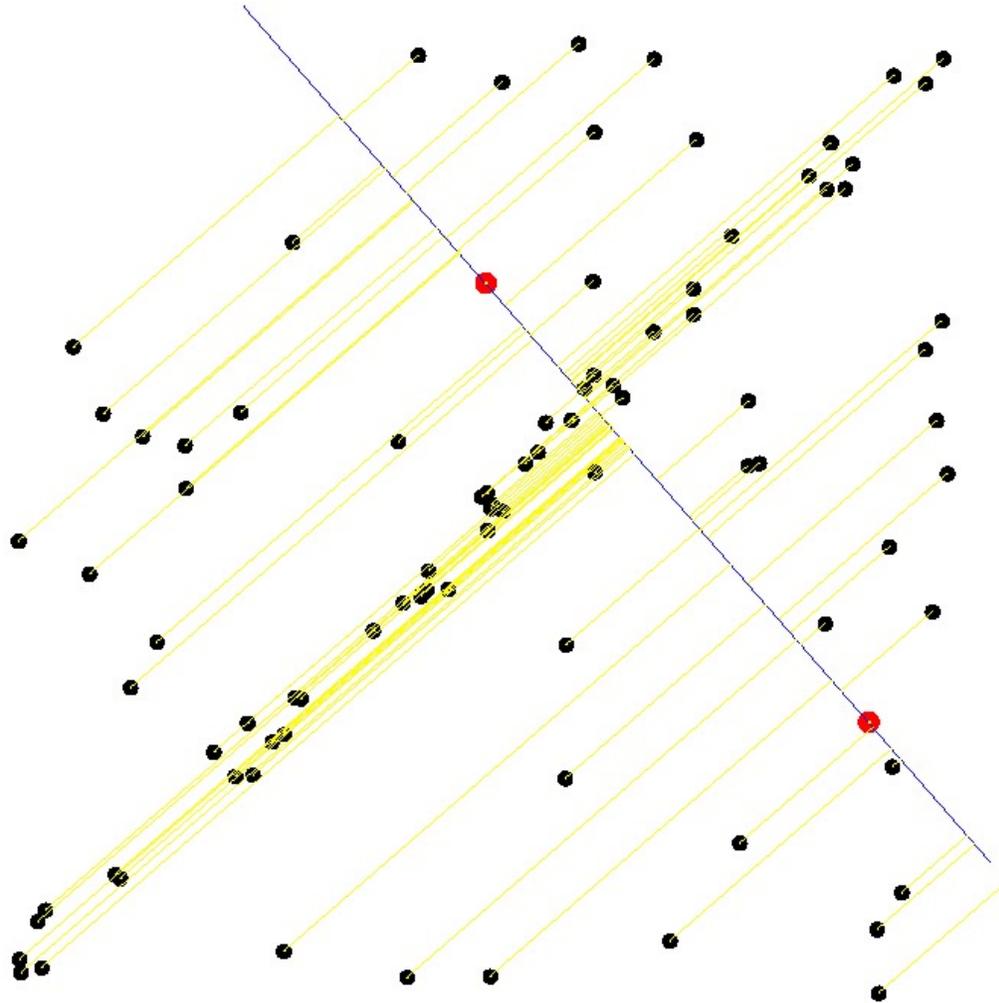
- **Select sample of 2 points at random**

Algorithm 3: RANSAC



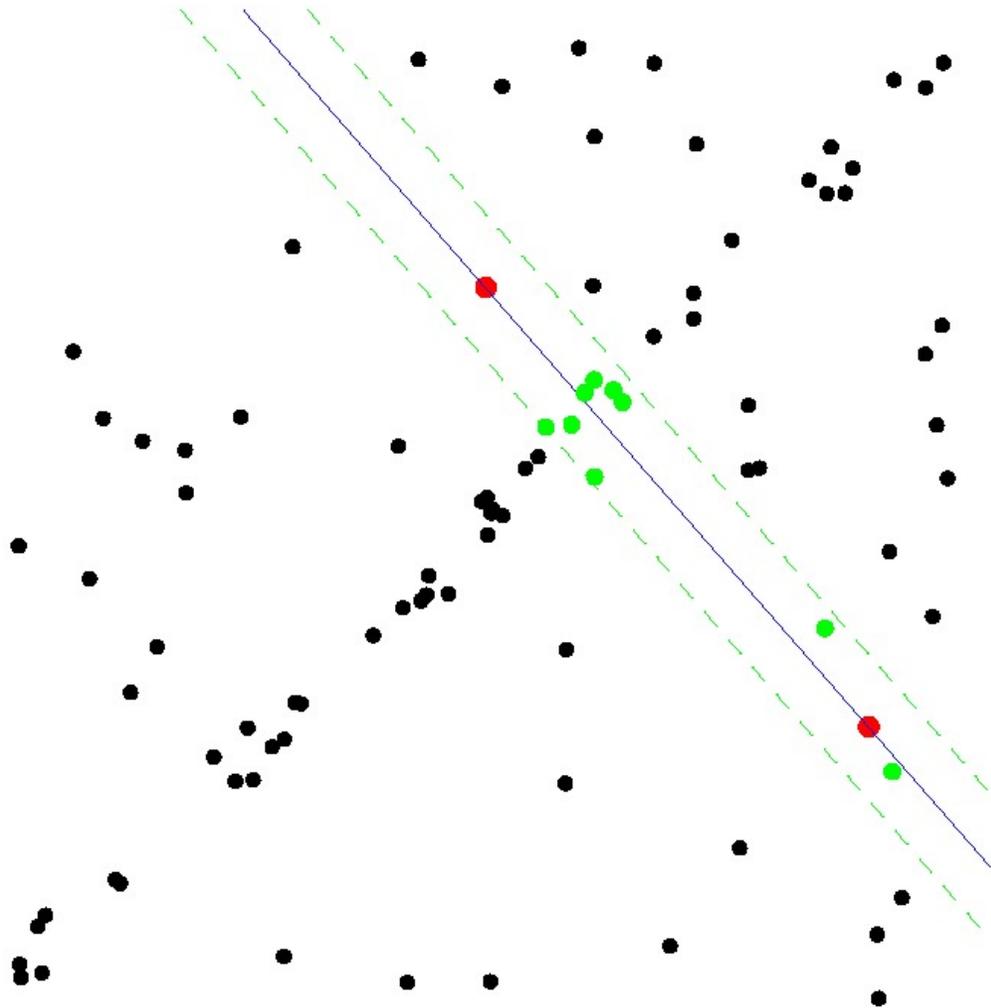
- Select sample of 2 points at random
- **Calculate model parameters that fit the data in the sample**

RANSAC



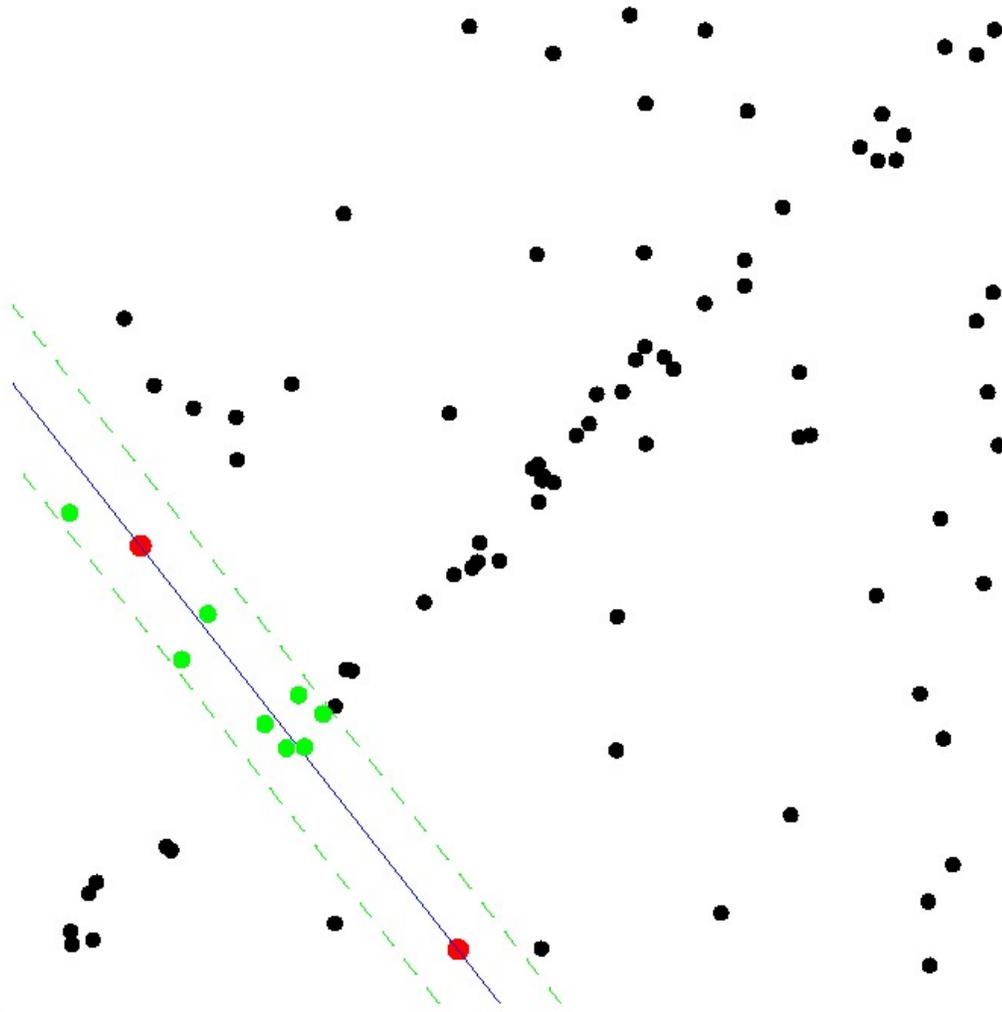
- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- **Calculate error function for each data point**

Algorithm 3: RANSAC



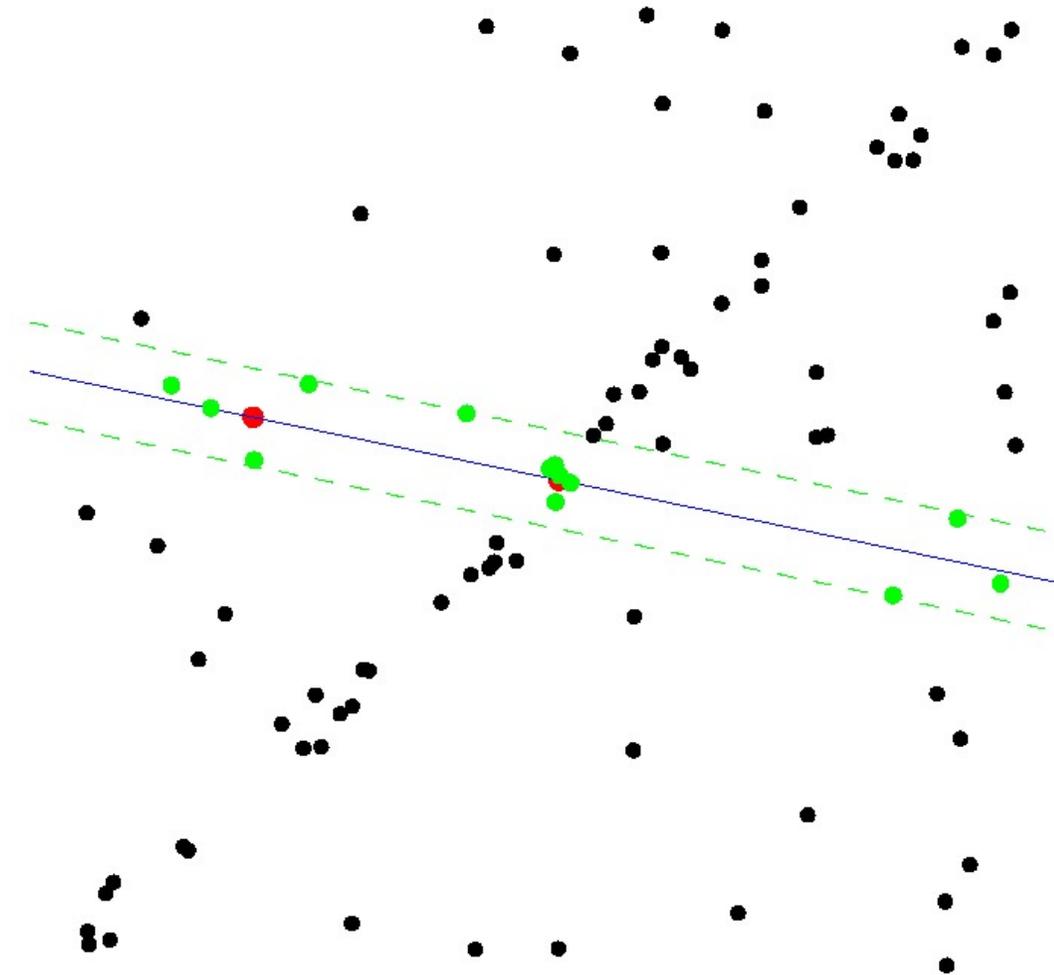
- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- Calculate error function for each data point
- **Select data that support current hypothesis**

Algorithm 3: RANSAC



- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- Calculate error function for each data point
- Select data that support current hypothesis
- **Repeat sampling**

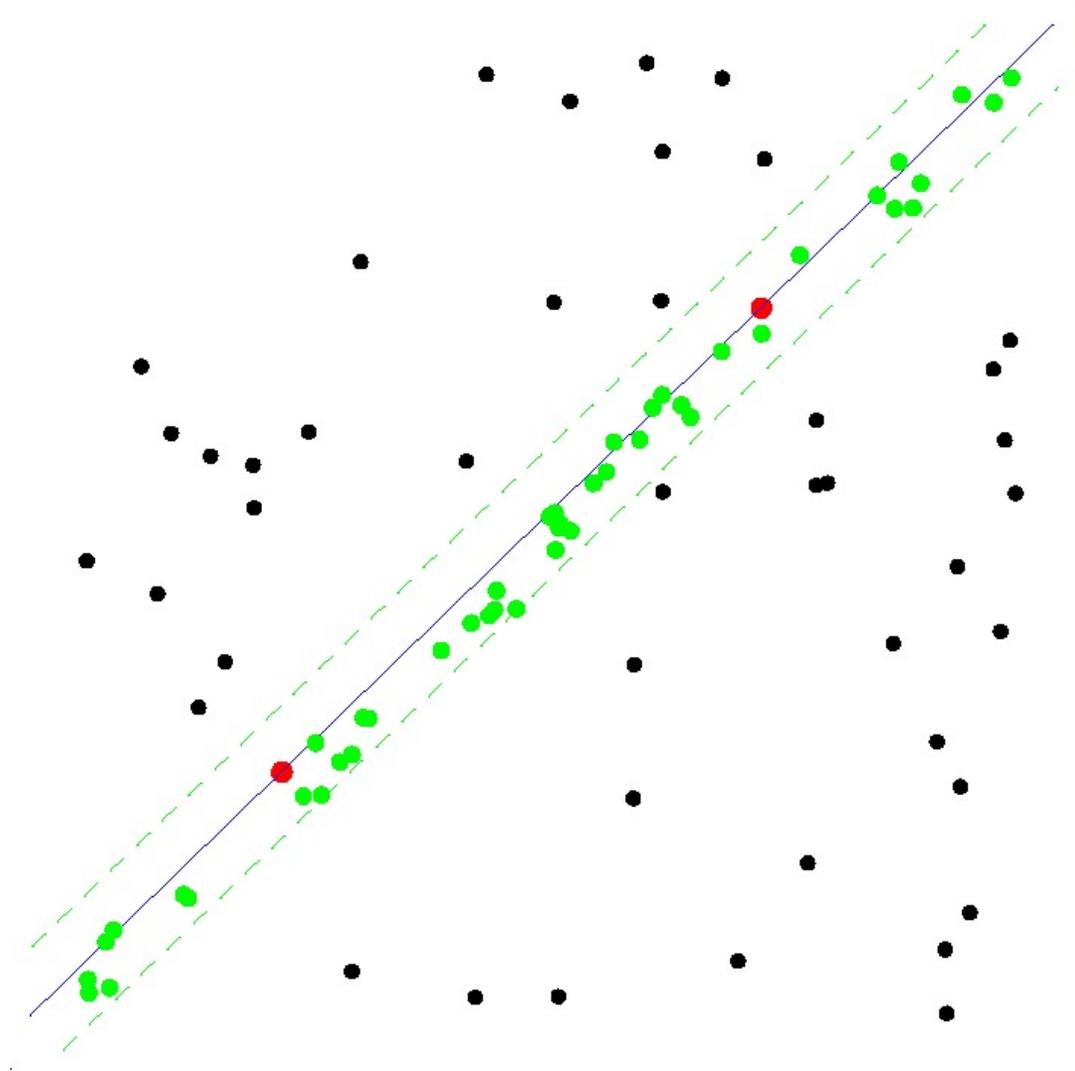
Algorithm 3: RANSAC



- Select sample of 2 points at random
- Calculate model parameters that fit the data in the sample
- Calculate error function for each data point
- Select data that support current hypothesis
- **Repeat sampling**

Algorithm 3: RANSAC

ALL-OUTLIER SAMPLE



Algorithm 3: RANSAC

Algorithm 4: *RANSAC*

1. Initial: let A be a set of N points
 2. **repeat**
 3. Randomly select a sample of 2 points from A
 4. Fit a line through the 2 points
 5. Compute the distances of all other points to this line
 6. Construct the inlier set (i.e. count the number of points with distance to the line $< d$)
 7. Store these inliers
 8. **until** Maximum number of iterations k reached
 9. The set with the maximum number of inliers is chosen as a solution to the problem
-

Algorithm 3: RANSAC

How many iterations does RANSAC need?

- Because we cannot know in advance if the observed set contains the maximum number of inliers, the ideal would be to check all possible combinations of 2 points in a dataset of N points.
- The number of combinations is given by $N(N-1)/2$, which makes it computationally unfeasible if N is too large. For example, in a laser scan of 360 points we would need to check all $360*359/2= 64,620$ possibilities!
- Do we really need to check all possibilities or can we stop RANSAC after iterations? The answer is that indeed we do not need to check all combinations but just a subset of them if we have a rough estimate of the percentage of inliers in our dataset
- This can be done in a probabilistic way

Algorithm 4: Hough-Transform

- Hough Transform uses a voting scheme

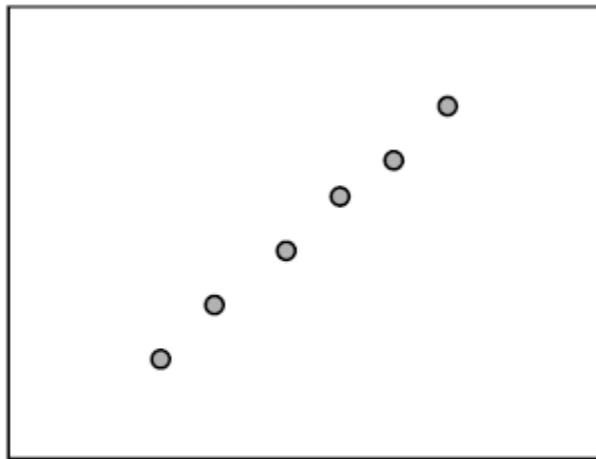
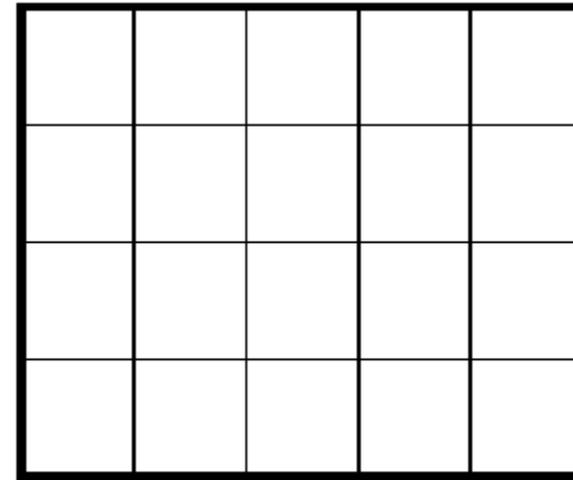
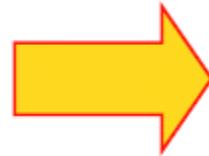


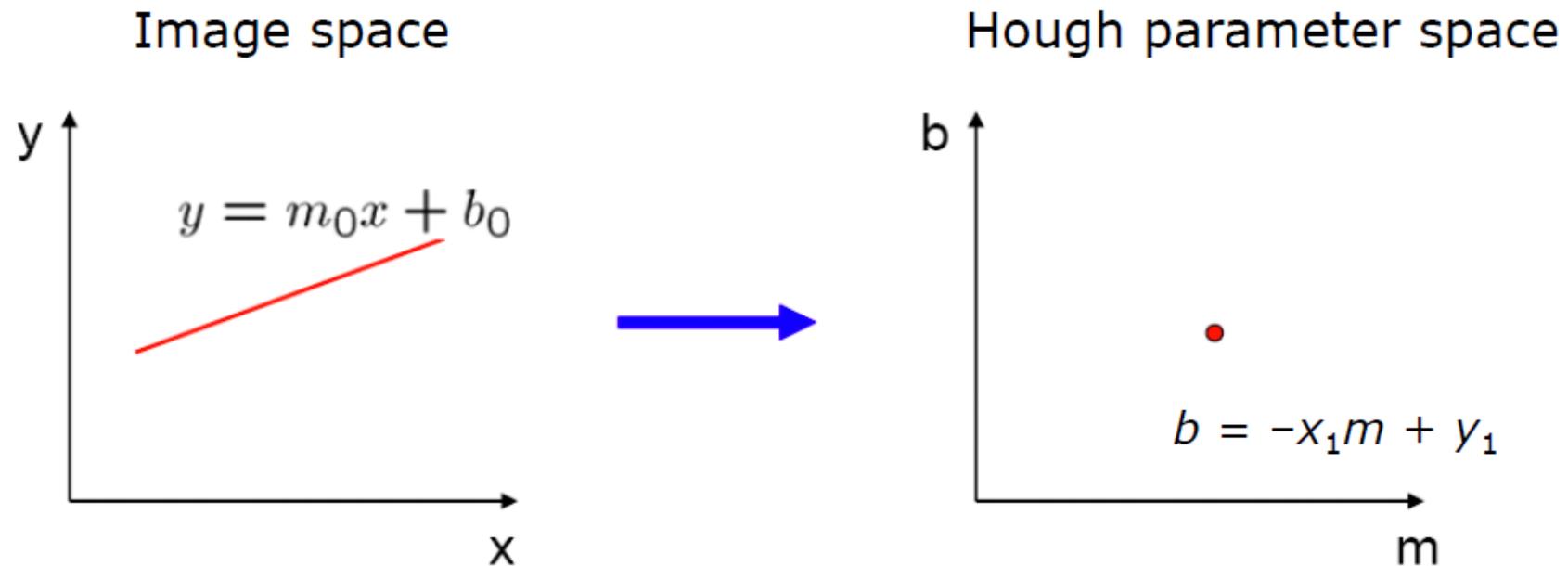
Image space



Hough parameter space

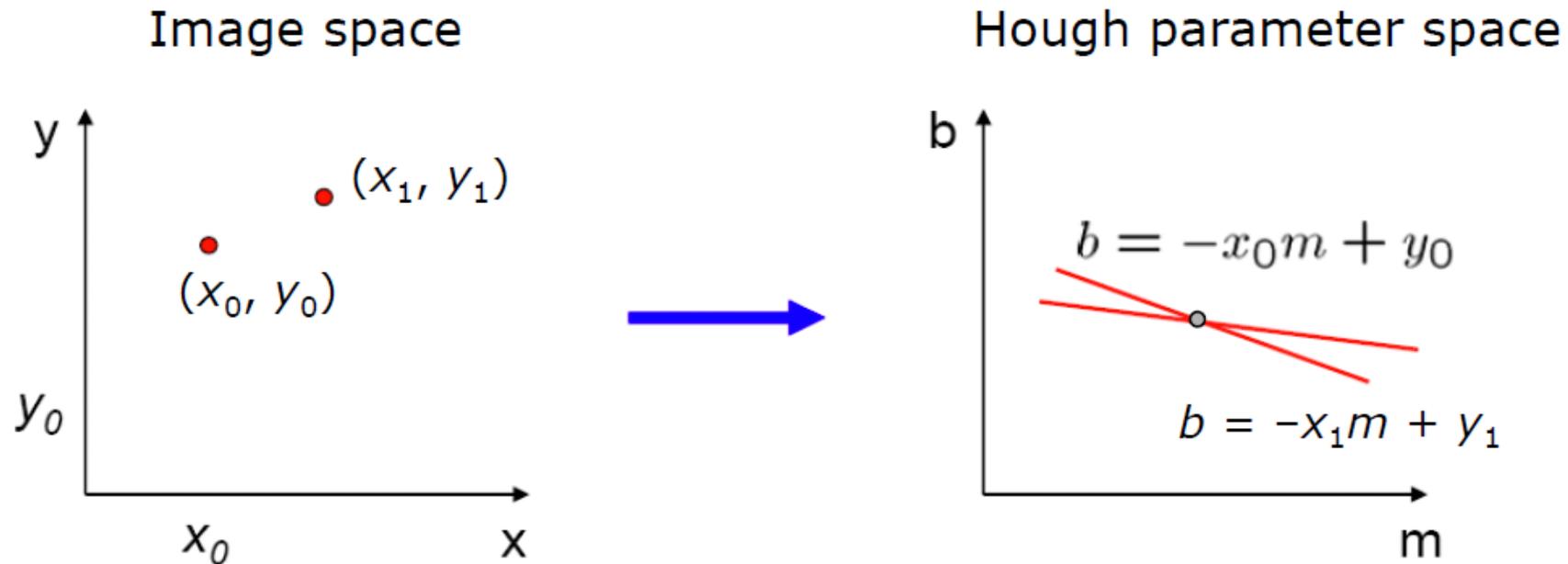
Algorithm 4: Hough-Transform

- A line in the image corresponds to a point in Hough space



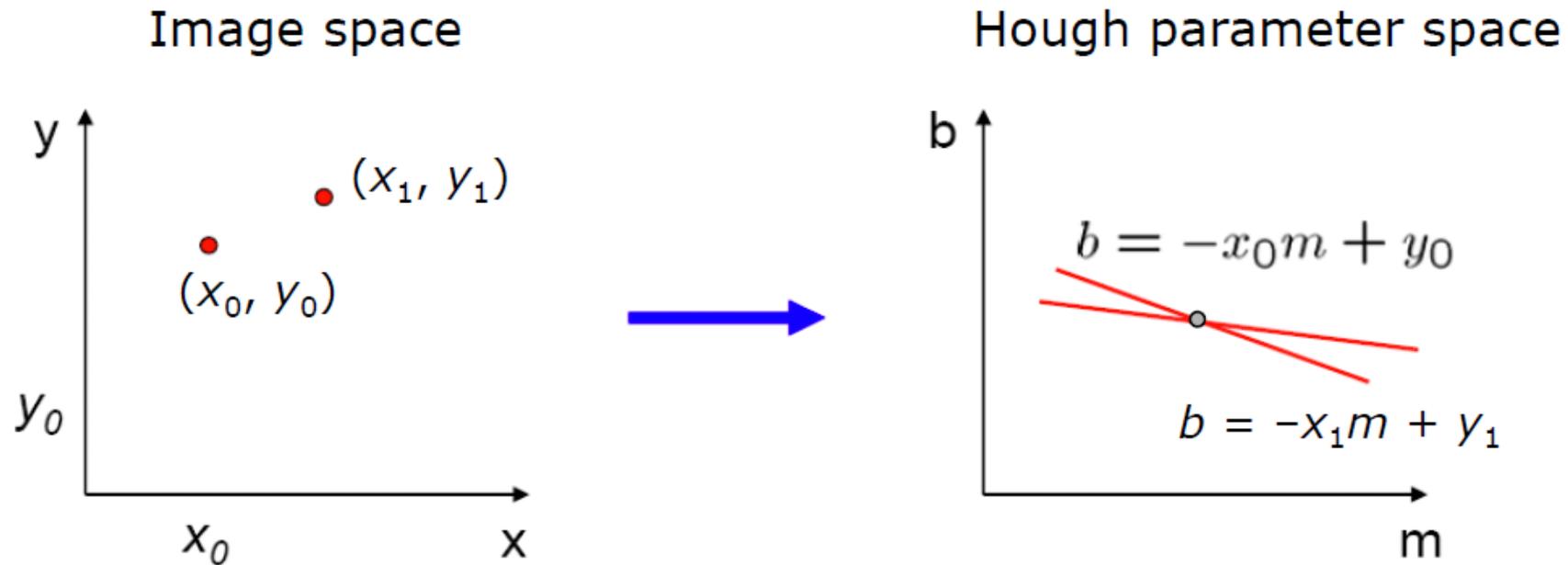
Algorithm 4: Hough-Transform

- What does a point (x_0, y_0) in the image space map to in the Hough space?



Algorithm 4: Hough-Transform

- Where is the line that contains both (x_0, y_0) and (x_1, y_1) ?
 - It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$

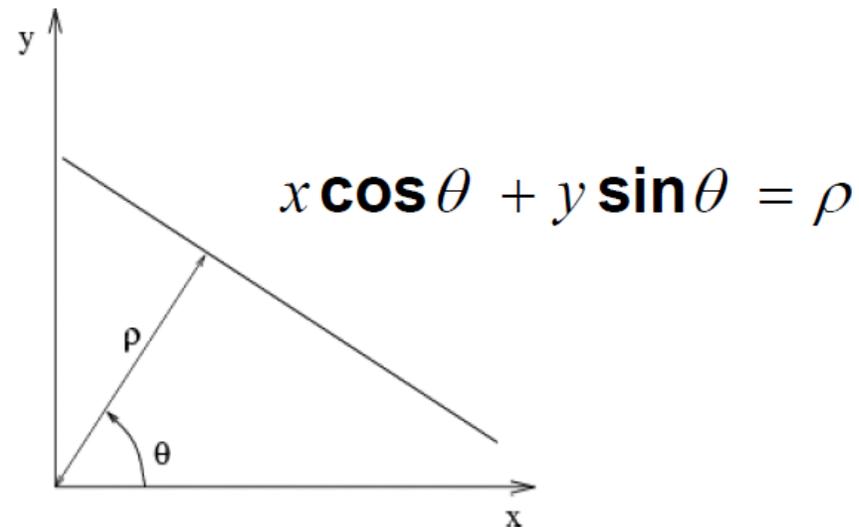


Algorithm 4: Hough-Transform

- Problems with the (m,b) space:
 - Unbounded parameter domain
 - Vertical lines require infinite m

Algorithm 4: Hough-Transform

- Problems with the (m,b) space:
 - Unbounded parameter domain
 - Vertical lines require infinite m
- Alternative: polar representation

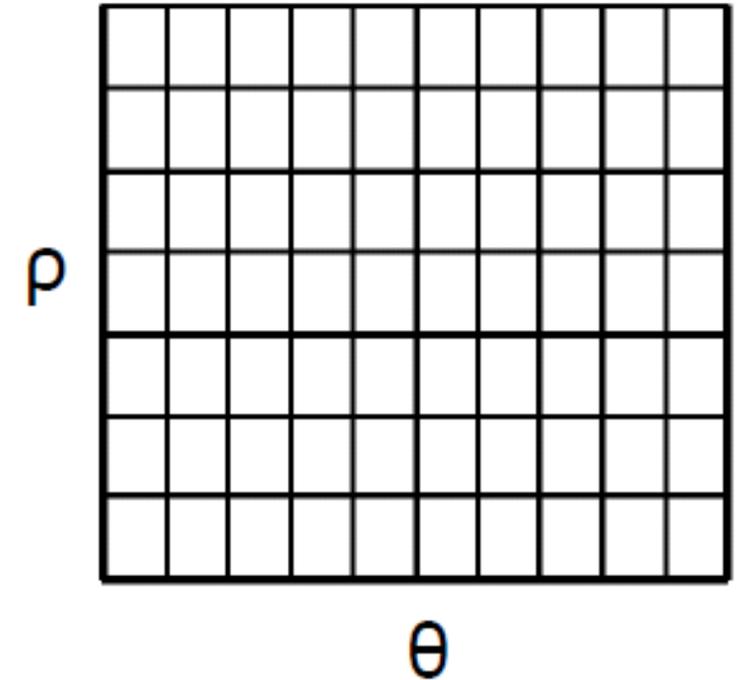


Each point will add a sinusoid in the (θ, ρ) parameter space

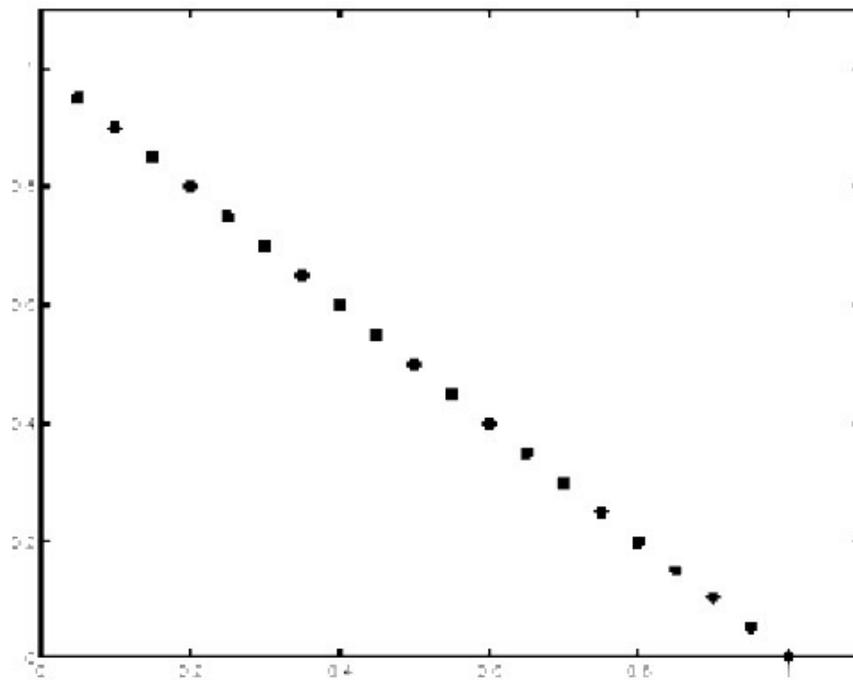
Algorithm 4: Hough-Transform

1. Initialize accumulator H to all zeros
2. For each edge point (x,y) in the image
 - For $\theta = 0$ to 180 (with a step size of e.g. 18)
 - $\rho = x \cos \theta + y \sin \theta$
 - $H(\theta, \rho) = H(\theta, \rho) + 1$
 - endend
3. Find the values of (θ, ρ) where $H(\theta, \rho)$ is a local maximum
4. The detected line in the image is given by $\rho = x \cos \theta + y \sin \theta$

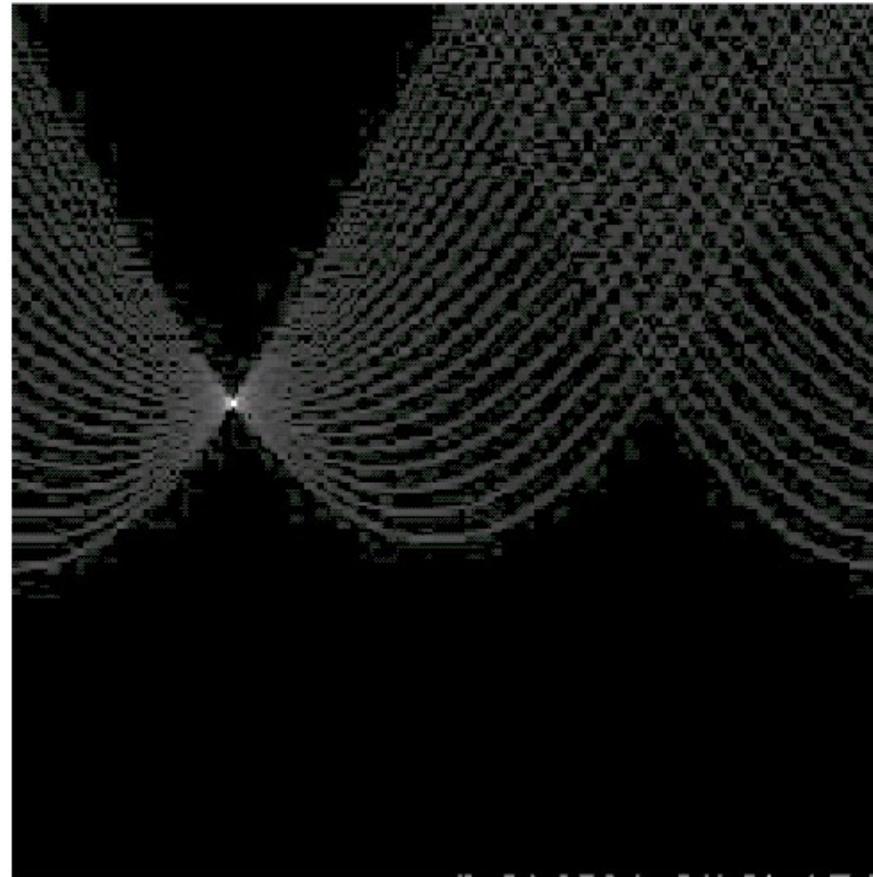
H: accumulator array (votes)



Algorithm 4: Hough-Transform



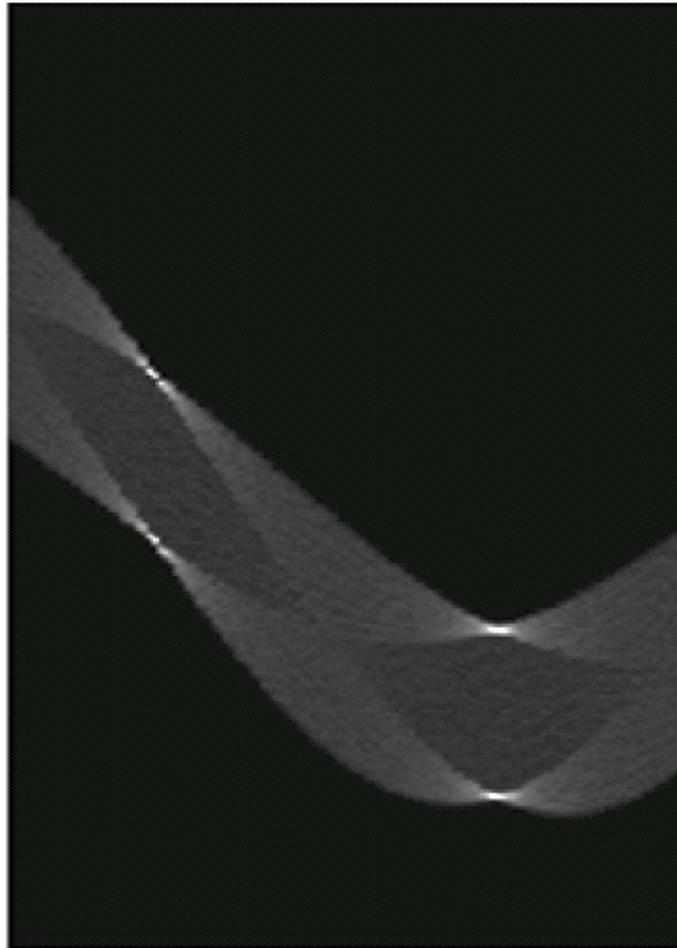
features



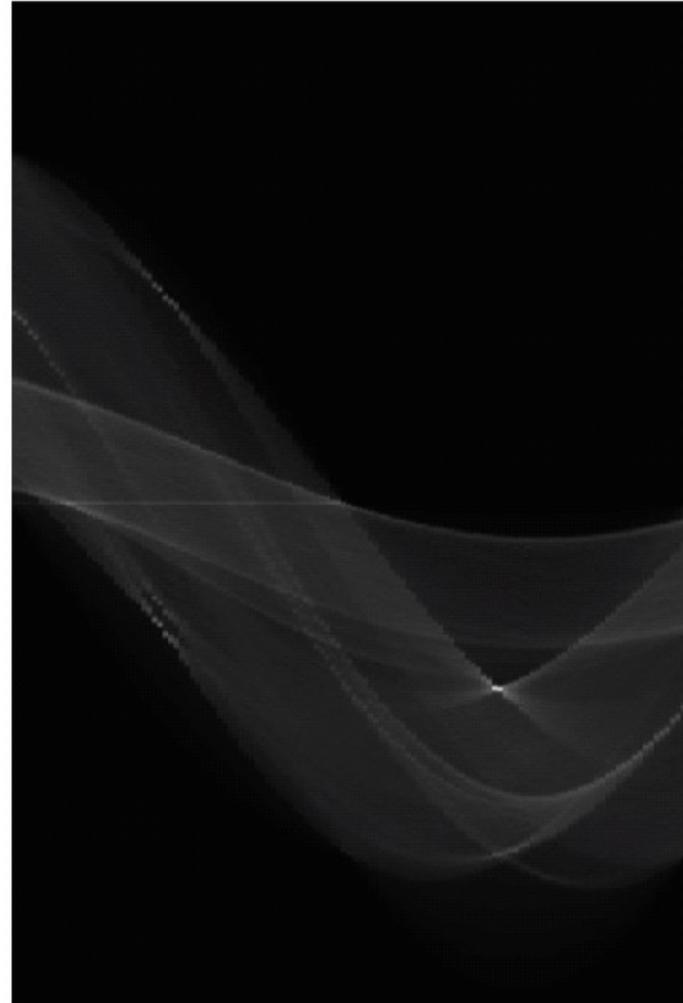
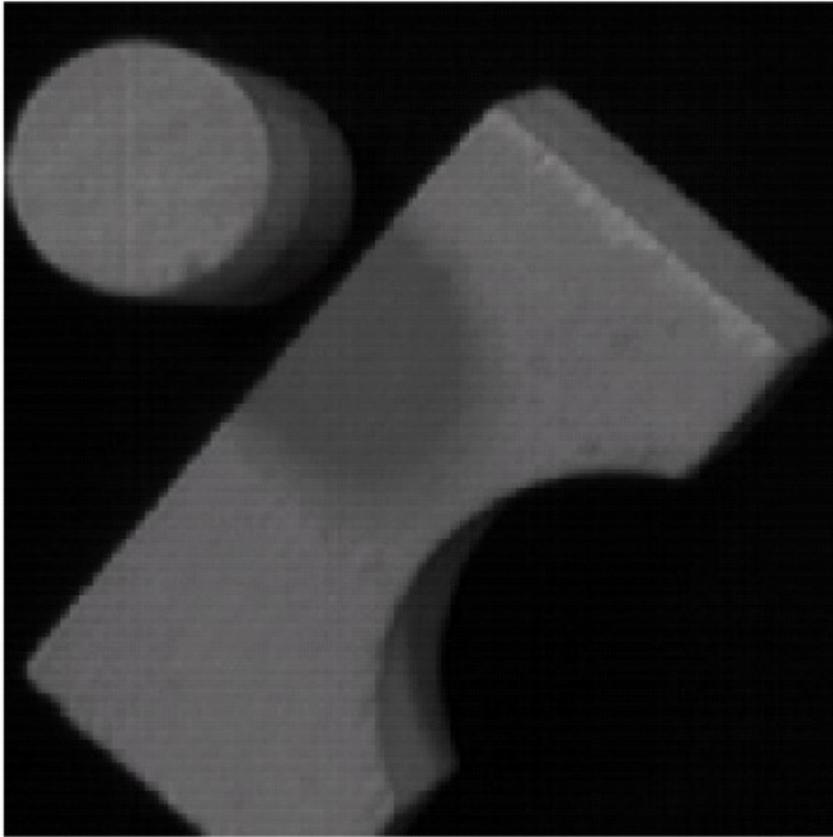
votes

Algorithm 4: Hough-Transform

Square

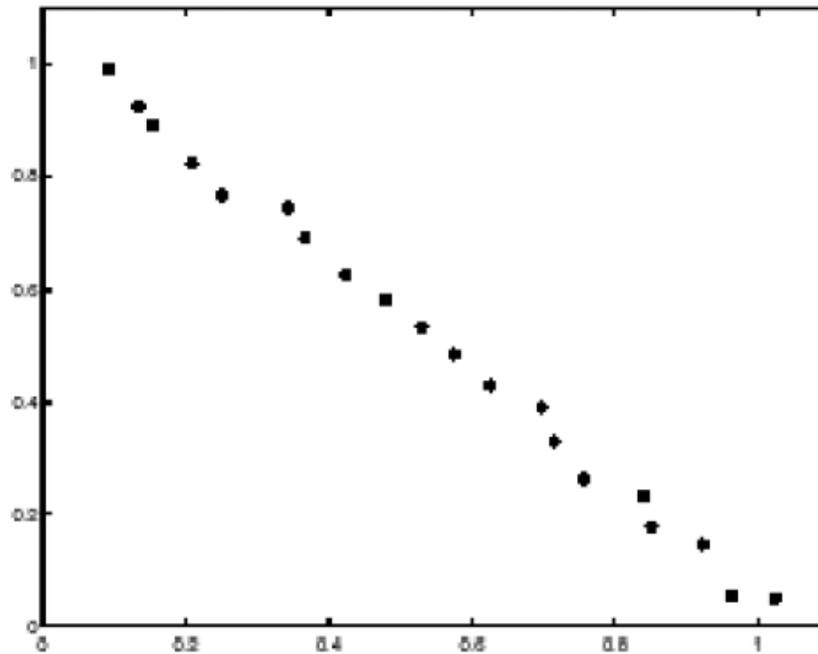


Algorithm 4: Hough-Transform

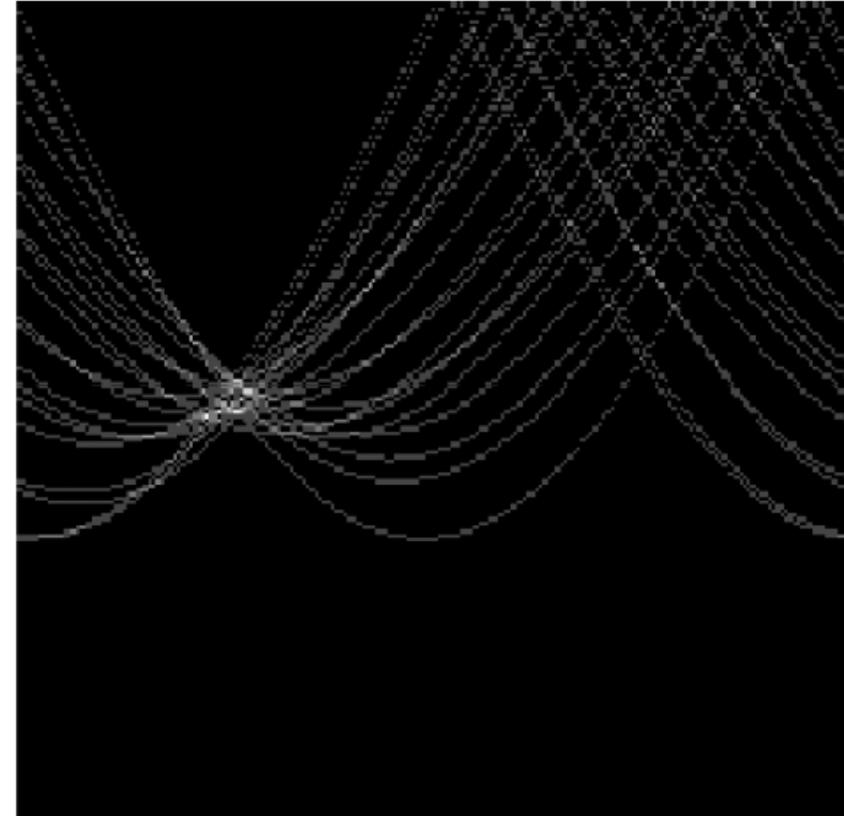


Algorithm 4: Hough-Transform

Effect of Noise



features



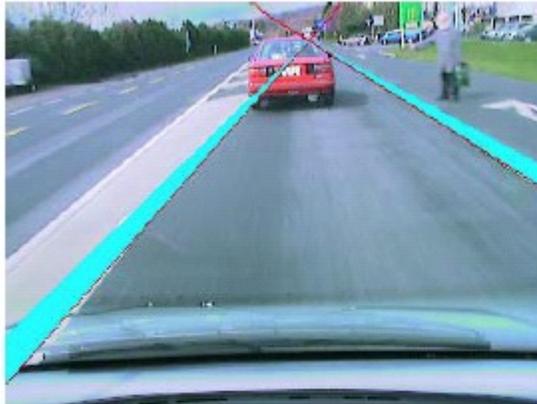
votes

- Peak gets fuzzy and hard to locate

Algorithm 4: Hough-Transform

Application: Lane detection

Inner city traffic



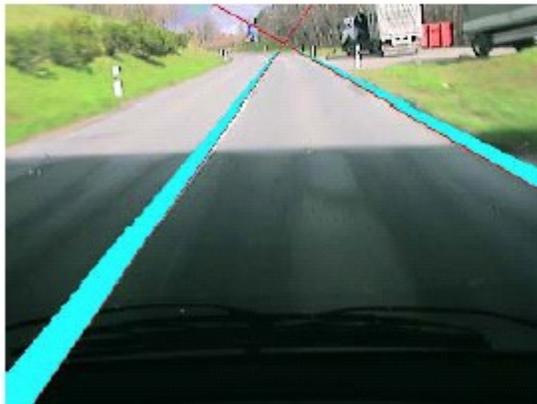
Ground signs



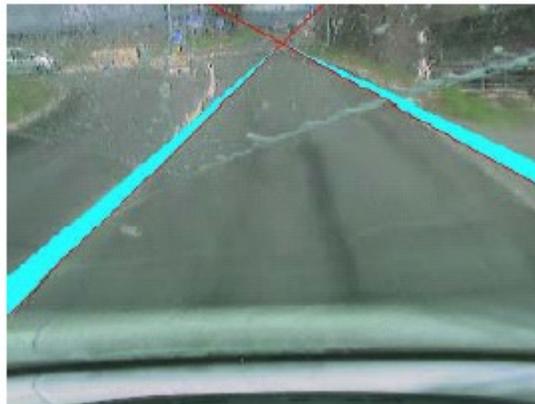
Country-side lane



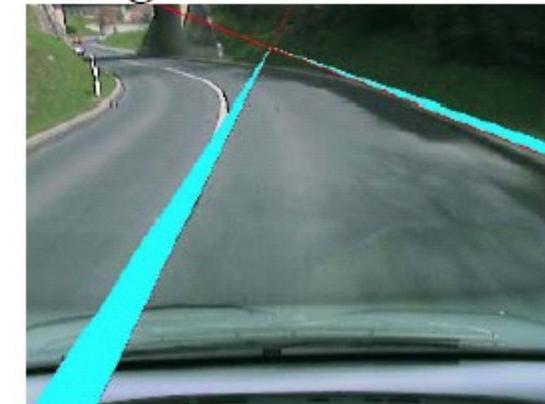
Tunnel exit



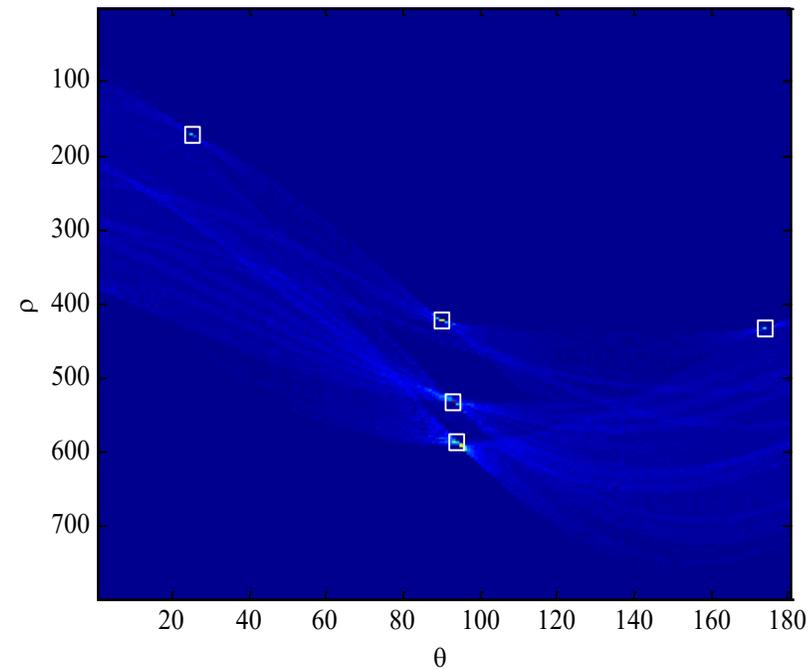
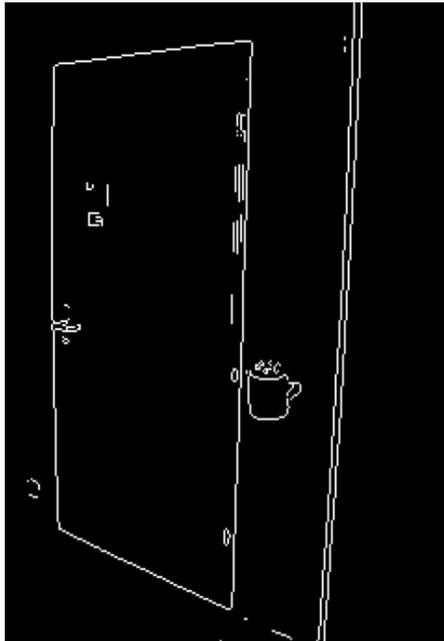
Obscured windscreen



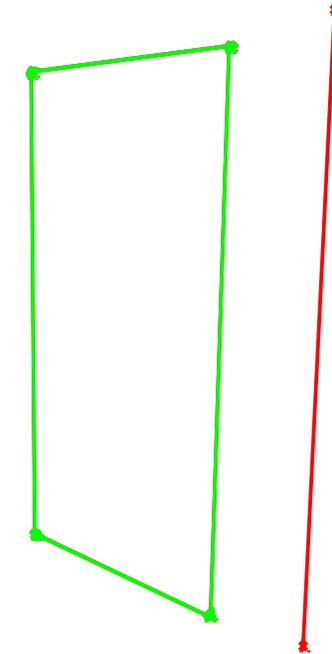
High curvature



Example – Door detection using Hough Transform



Hough Transform



Hough Transform: other features

Lines:

$$p = (d, \nu)$$
$$g(x, y, p) := x \cdot \cos(\nu) + y \cdot \sin(\nu) - d$$

Circles:

$$p = (x_0, y_0, r)$$
$$g(x, y, p) := (x - x_0)^2 + (y - y_0)^2 - r^2$$

Ellipses:

$$p = (x_0, y_0, a, b, \psi)$$
$$g(x, y, p) := \frac{\left[(x - x_0) \cdot \cos(\psi) + (y - y_0) \cdot \sin(\psi) \right]^2}{a^2} + \frac{\left[(y - y_0) \cdot \cos(\psi) - (x - x_0) \cdot \sin(\psi) \right]^2}{b^2} - 1$$

Hough Transform

- Advantages

- Noise and background clutter do not impair detection of local maxima
- Partial occlusion and varying contrast are minimized

- Negatives

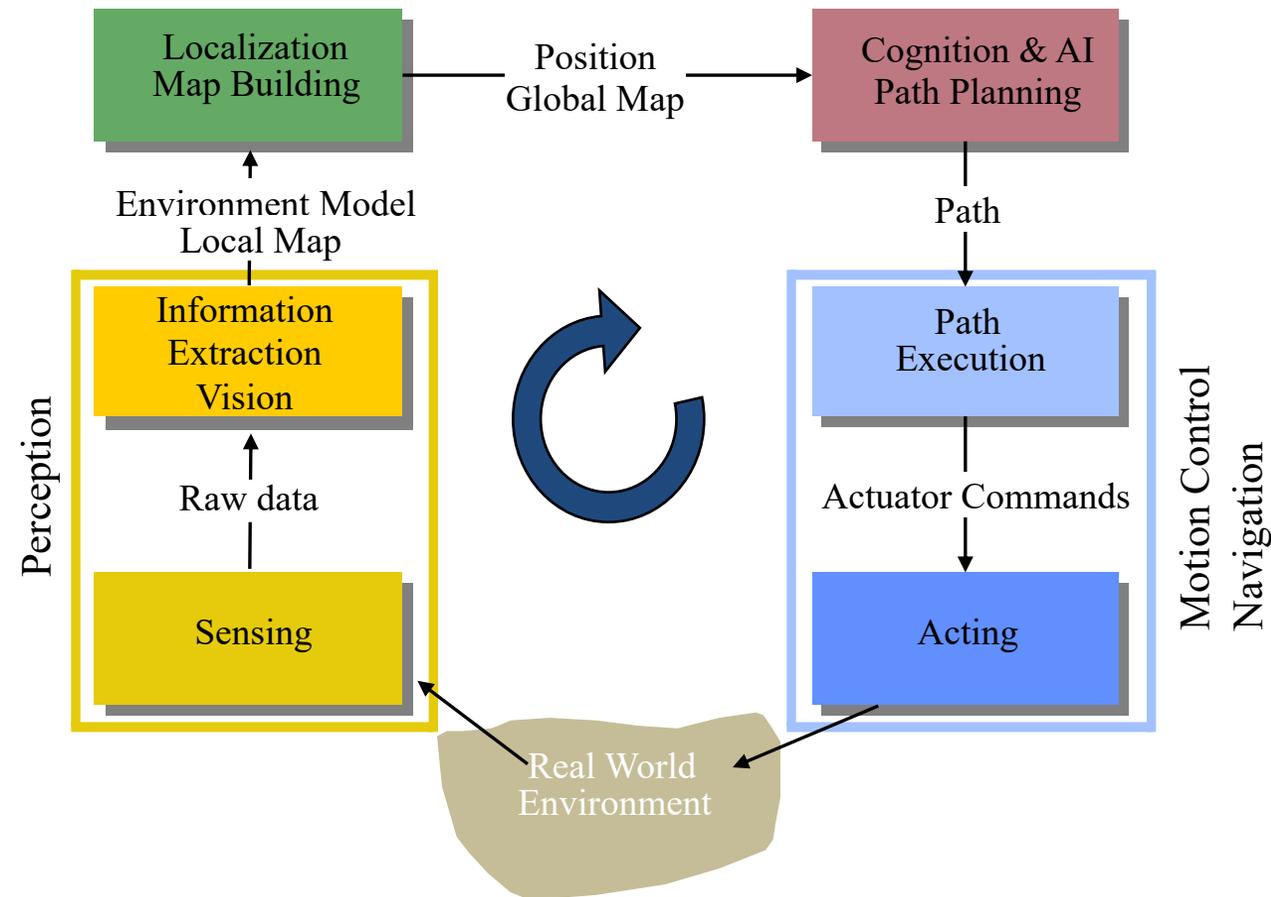
- Requires time and space storage that increases exponentially with the dimensions of the parameter space

Comparison Line Detection

- Deterministic methods perform better with laser scans
 - Split-and-merge, Line-Regression, Hough transform
 - Make use of the sequencing property of scan points.
- Nondeterministic methods can produce high False Positives
 - RANSAC
 - Do not use the sequencing property
 - But it can cope with outliers
- Overall:
 - Split-and-merge is the fastest, best real-time application

MAP REPRESENTATION

General Control Scheme for Mobile Robot Systems



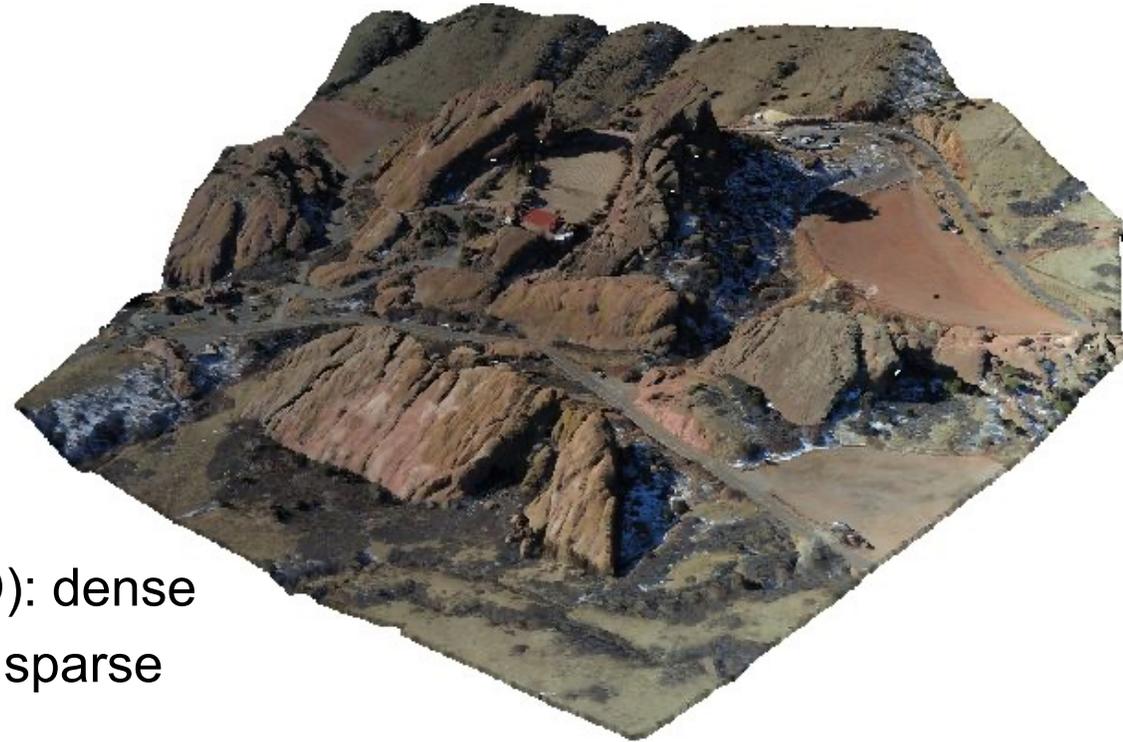
Map Representation: what is “saved” in the map

- Points (surface of objects, buildings): 2D or 3D
 - What: x,y or x,y,z coordinates;
Optional: intensity; maybe RGB; maybe descriptor;
temperature; ...
 - From range sensors (laser, ultrasound, stereo, RGB-D): dense
 - From cameras (structure from motion; feature points): sparse
 - Variant: kd-tree
- Grid-map: 2D or 3D
 - Option: probabilistic grid map
 - Option: elevation map
 - Option: cost map
 - Option: Truncated Signed Distance Field
 - Option: Normal Distributions Transform (NDT)
 - Variant: Quad-tree; Oct-tree
- Higher-level Abstractions
 - Lines; Planes; Mesh
 - Curved: splines; Superquadrics
- Semantic Map
 - Assign semantic meaning to entities of a map representation from above
 - E.g. wall, ceiling, door, furniture, car, human, tree, ...
- Topologic Map
 - High-level abstraction: places and connections between them
- Hierarchical Map
 - Combine Maps of different scales. E.g.:
 - Campus, building, floor
- Pose-Graph Based Map
 - Save (raw) sensor data in graph, annotated with the poses; generate maps on the fly
- Dynamic Map
 - Capture changing environment
- Hybrid Map
 - Combination of the above

Point based map

- Point Cloud

- What: x,y or x,y,z coordinates;
Optional: intensity; maybe RGB; maybe descriptor; temperature; ...
- From range sensors (laser, ultrasound, stereo, RGB-D): dense
- From cameras (structure from motion; feature points): sparse
- Variant: kd-tree
- Structured point cloud: points are ordered in a 2D grid -> could calculate depth image from it
- PCL: point cloud library: Used in ROS; <https://pointclouds.org/>
- `sensor_msgs/PointCloud` `sensors_msgs/PointCloud2`
- Excellent viewer: CloudCompare <https://www.danielgm.net/cc/>
- File formats: PLY (bin & ASCII); XYZ (ASCII); PCD (from pcl); LAS (terrestrial scanning); ...

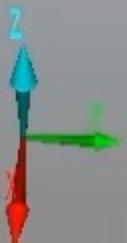


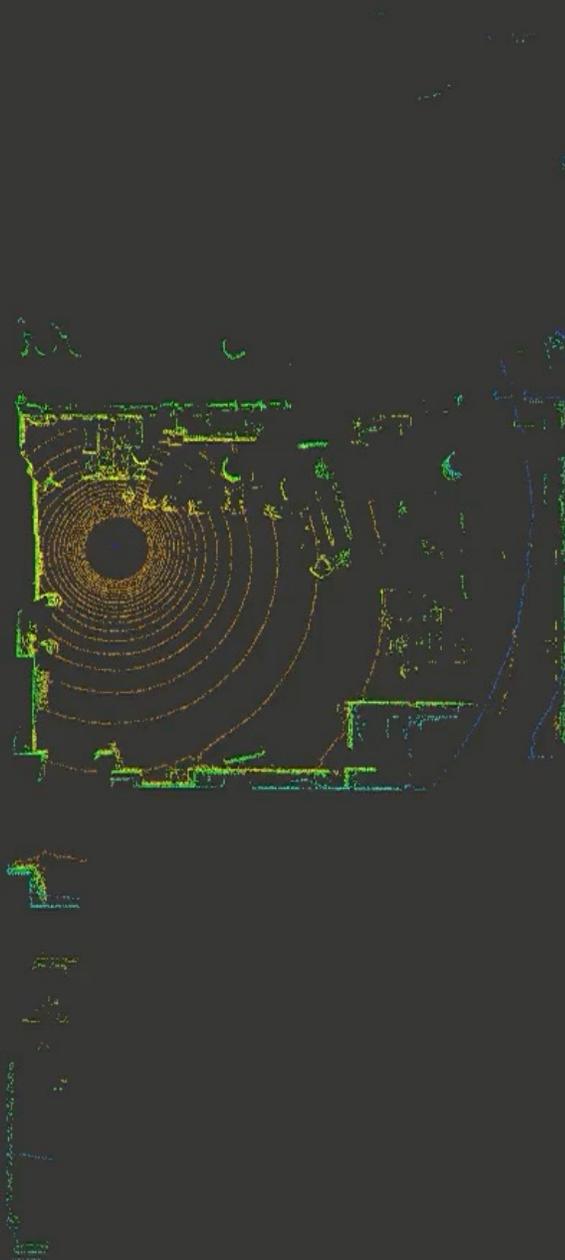
RGBD-Inertial Trajectory Estimation and Mapping for Ground Robots

Zeyong Shan, Ruijian Li and Sören Schwertfeger



Source code: <https://github.com/STAR-Center/VINS-RGBD>



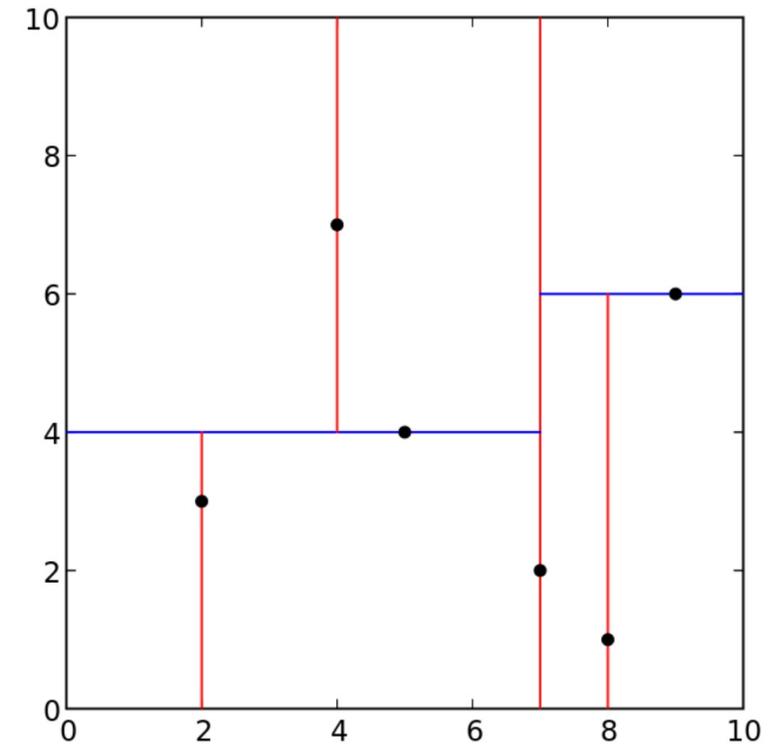
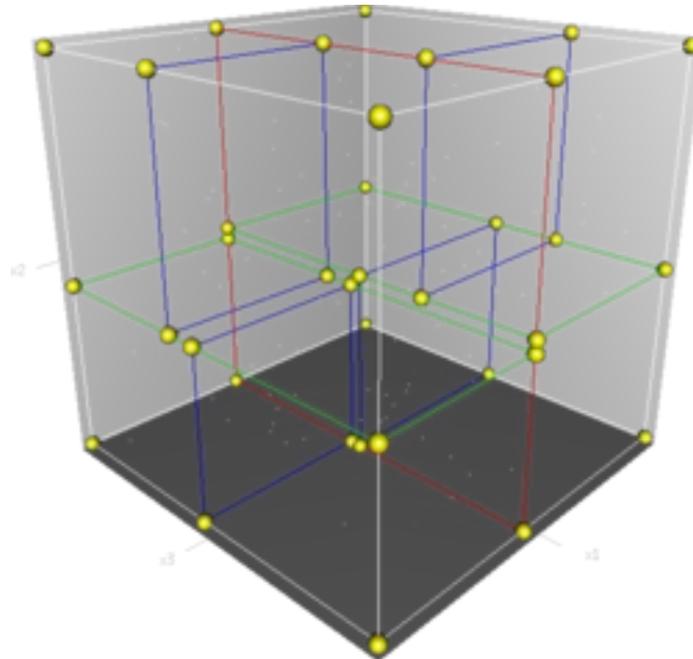




k-d tree

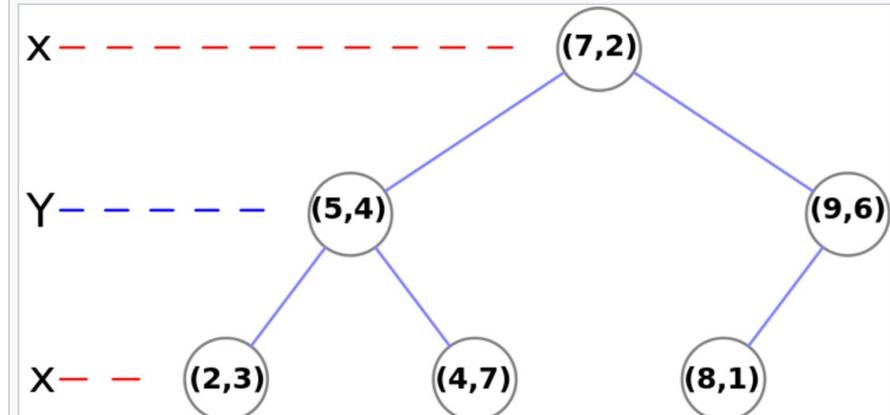
- k-dimensional binary search tree
- Robotics: typically 3D or 2D
- Every level of tree:
 - For a different axis (e.g. x,y,z,x,y,z,x,y,z) (\Rightarrow split space with planes)
 - Put points in left or right side based on median point (w.r.t. its value of on the current axis) \Rightarrow
 - Balanced tree
- Fast neighbor search \rightarrow ICP!

Algorithm	Average	Worst case
Space	$O(n)$	$O(n)$
Search	$O(\log n)$	$O(n)$
Insert	$O(\log n)$	$O(n)$
Delete	$O(\log n)$	$O(n)$



k-d tree decomposition for the point set

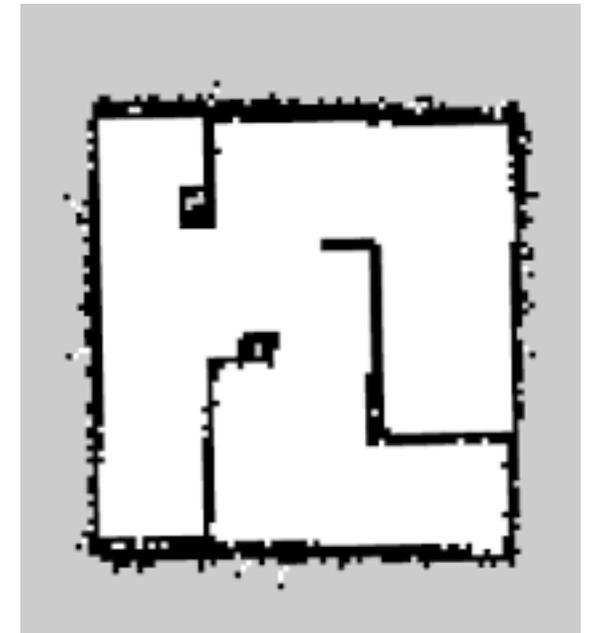
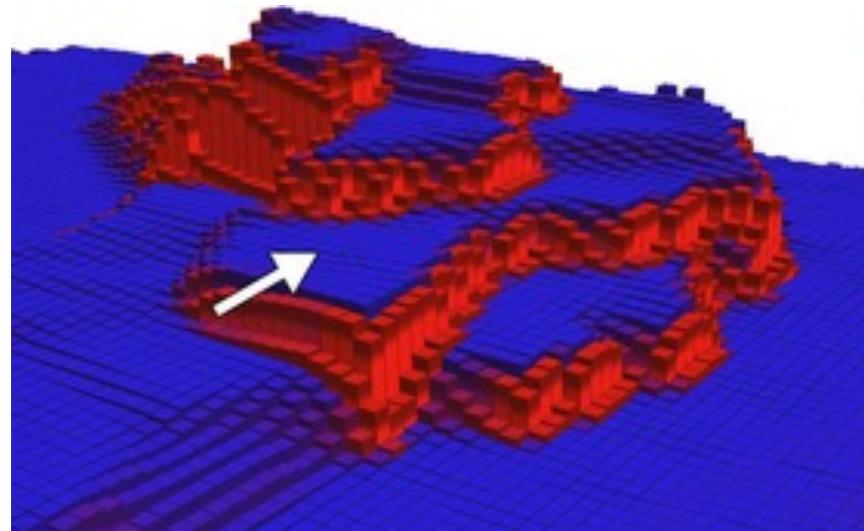
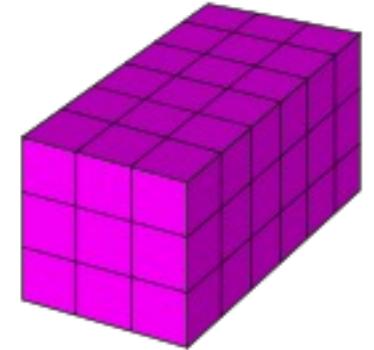
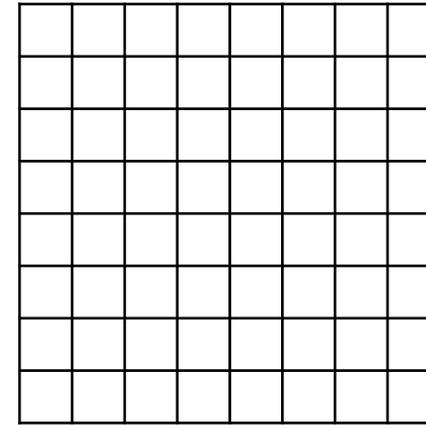
$(2,3), (5,4), (9,6), (4,7), (8,1), (7,2)$.



The resulting k-d tree.

Grid Maps

- Grid-map: 2D or 3D cartesian grid
- What to save in the cell:
 - Binary: Free; Occupied;
 - Colored: +Unknown; +Searched; +Path; ...
 - Probability of being occupied 0...1.0
 - Height above ground: Elevation Map
 - Cost map: used for planning (covered later)

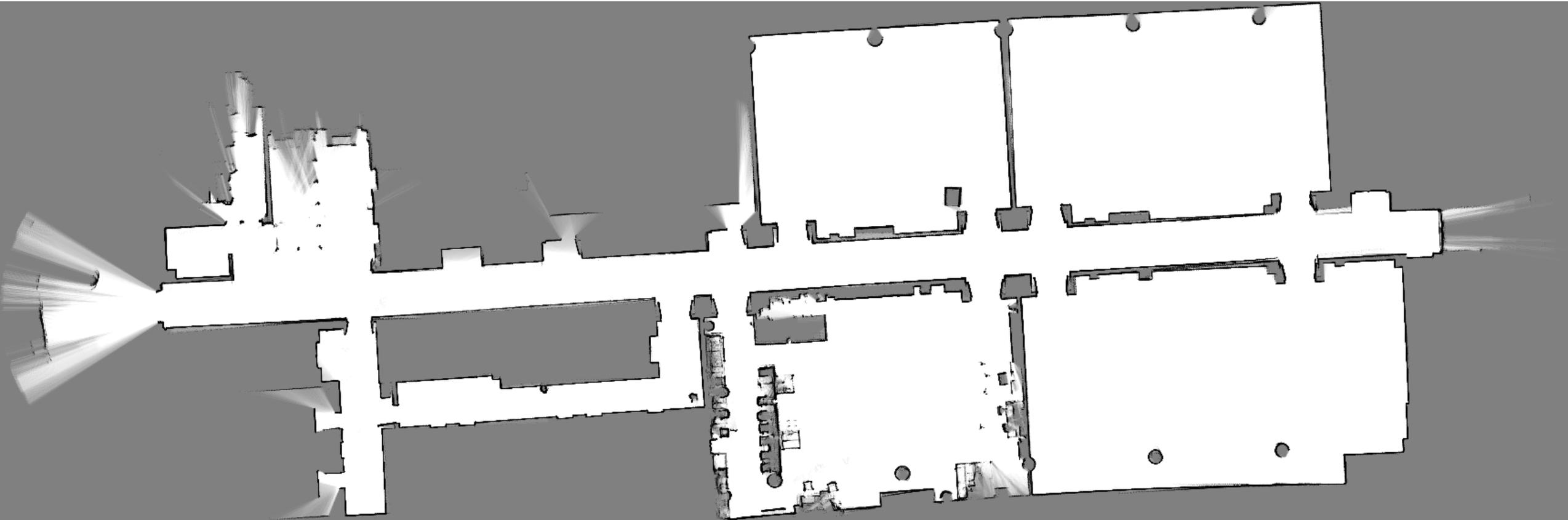


Grid Maps

- Resolution: e.g. 1 pixel == 5cm
- Size: e.g. 2000 pixel x 2000 pixel
- Scale: e.g. 1:100.000 (2DoF of 3 parameters resolution, size, scale)
- Frame
 - A map has a frame. Where? In the center of the image? Top left corner?
- Time stamp
- Formats:
 - Image: png, pgm (grey scale), tiff, jpeg (lossy!)
 - GeoTIFF: georeferenced TIFF (inlined tfw World File): map projection, coordinate systems, ellipsoids, datums (see GPS slide)
 - 3D: bt (octomap)
- QGIS: open source Geographic Information System for raster (image) and vector data <http://qgis.org/en/site>

Probabilistic grid map

- 1: occupied; 0: free; 0.5: unknown
- Need error model of sensor (and of localization) to properly update cells with a scan
- Can remove dynamic (moving) objects (by observing the free space multiple times)



Normal Distributions Transform (NDT)

- Sparse Gaussian mixture model
- Each cell has NDT Computed from covariance matrix of points inside the cell
- Useful for scan matching/ registration

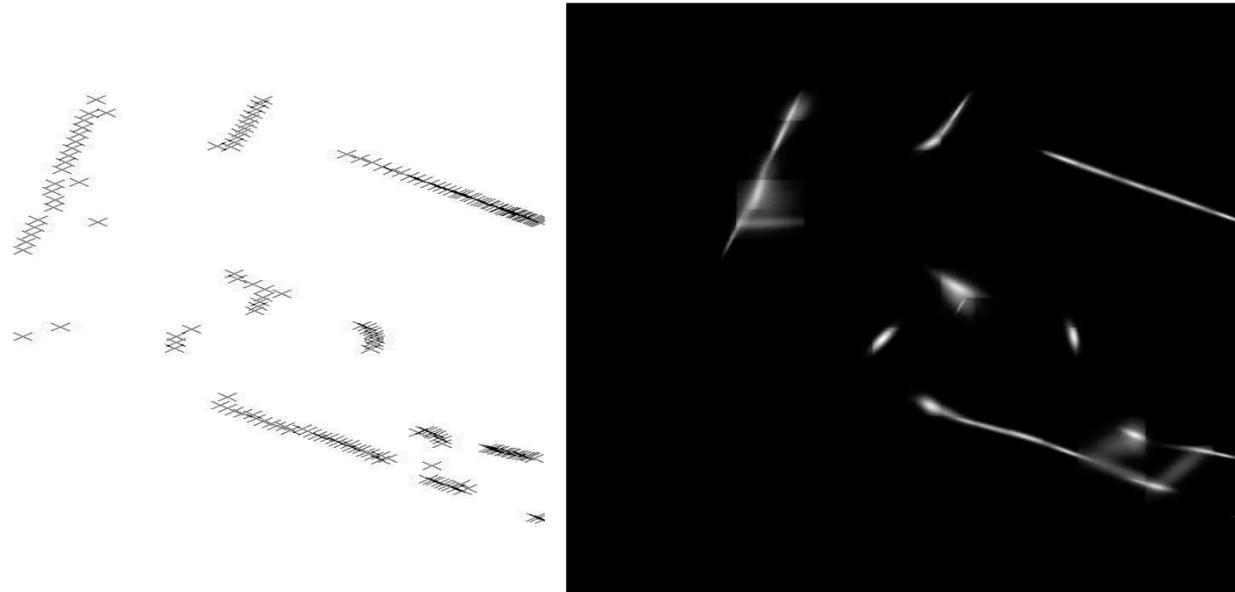
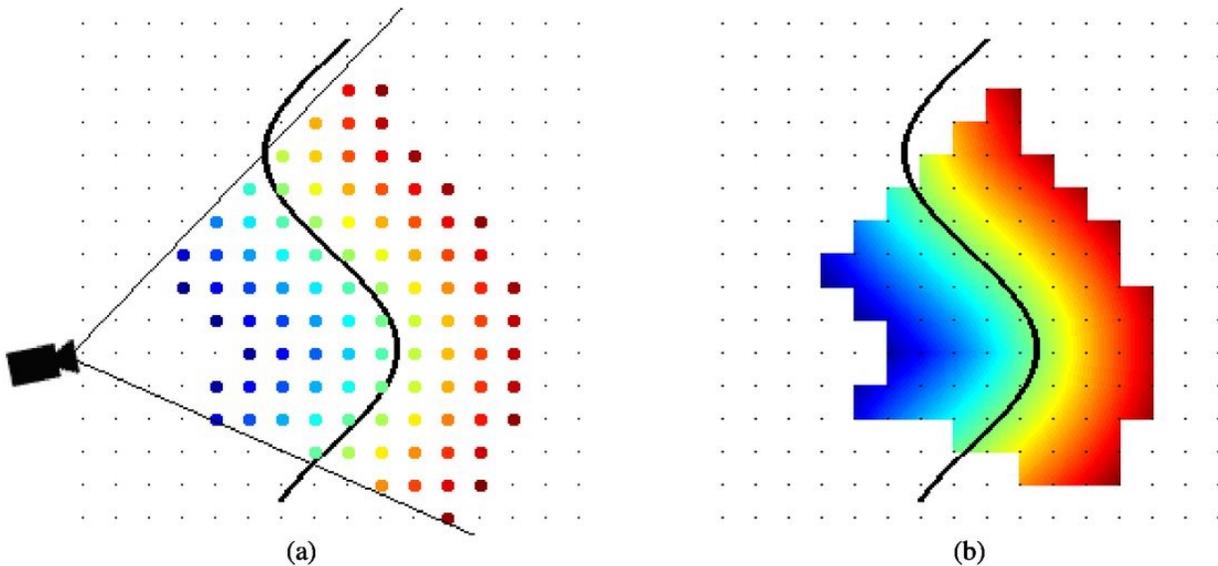


Fig. 1. An example of the NDT: The original laser scan and the resulting probability density.

Truncated Signed Distance Field

- Distance to surface:
- positive in free space; negative behind
- Up to a certain distance (truncated)
- Useful for collision checking; planning; meshing
- Can be modeled using Gaussian random variable in each cell



<https://www.dci.org/2020/03/20/active-exploration-in-signed-distance-fields/>
<https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9091/909117/Real-time-reconstruction-of-depth-sequences-using-signed-distance-functions/10.1117/12.2054158.short?SSO=1&tab=ArticleLinkFigureTable>

