Features

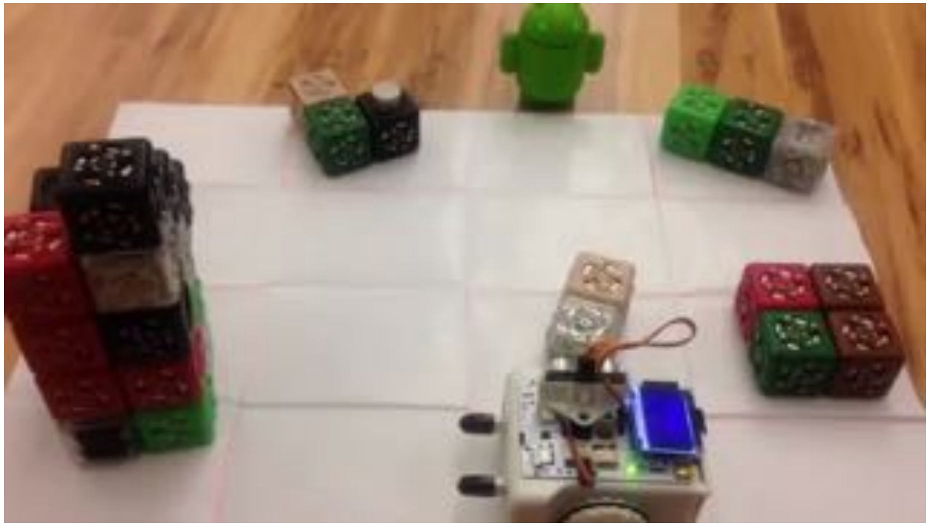
Chapter 6

Last week: Sensors

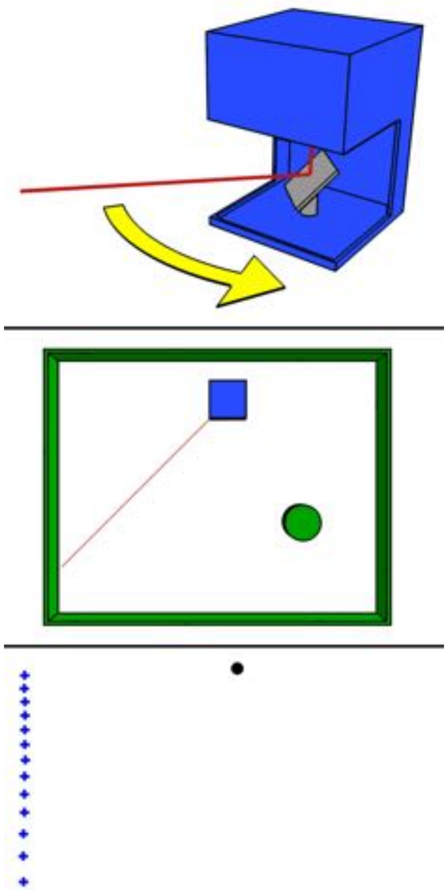
- Different sensor technologies
  - Distance, velocity, acceleration
  - Light, sound, magnetic field, …
- Precision, accuracy, bandwidth, dynamic range and resolution
- No single sensor for any application
- Running example: robot navigation

Last Friday: Dijkstra on Sparki

- Please implement till next Friday
- This Friday: moving the robot from waypoint to waypoint
- Brute-force implementation of Dijkstra or google (30 lines of code)



How much data does a laser scanner produce?



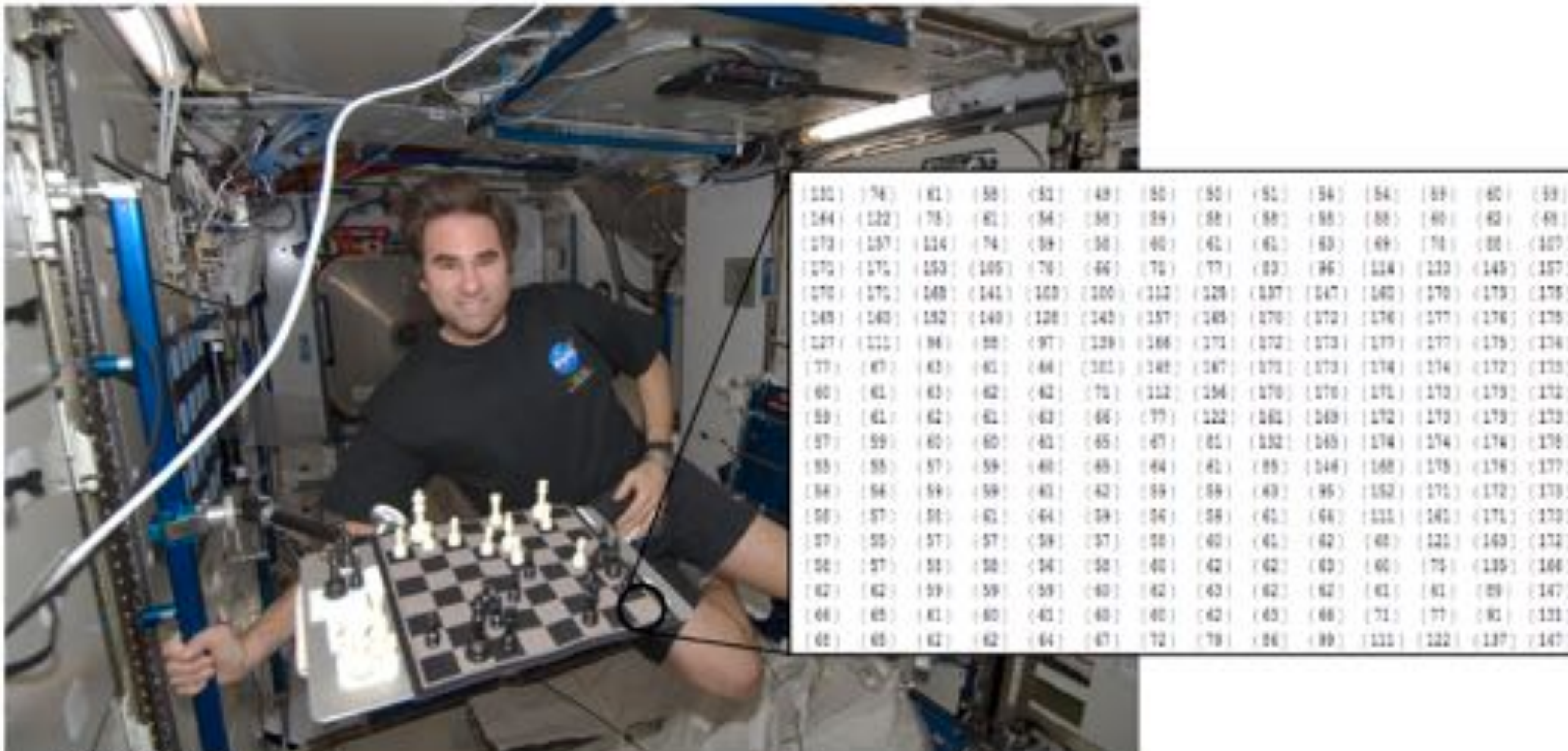
Specifications	
Power source	5V +/-5%
Current consumption	0.5A (Rush current 0.8A)
Detection range	0.02 to approximately 4m
Laser wavelength	785nm, Class 1
Scan angle	240°
Scan time	100msec/scan (10.0Hz)
Resolution	1mm
Angular Resolution	0.36°
Interface	USB 2.0, RS232
Weight	5.0 oz (141 gm)

Specification	URG-04LX
Power source	Regulated 5V ±5%
Interface	RS232, USB
Detection Distance	20 to 4000 (mm)
Guaranteed Accuracy (min to 1m)	±10mm
Guaranteed Accuracy (1m to max)	1% of detected distance

Hokuyo URG

What to do with so much data? (Cameras are even worse!)

Cameras

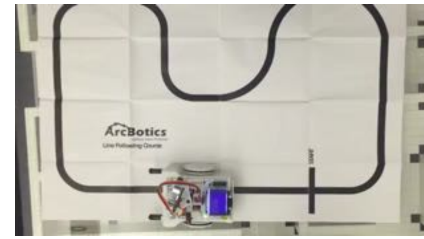


ISS017E011574

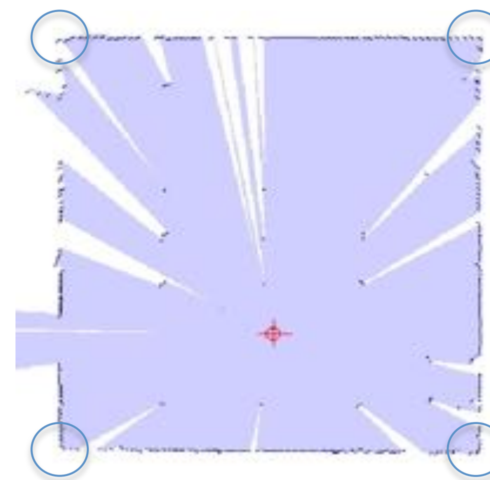
What kind of high-level information could we possibly extract to aid in robot navigation?

Ideas

- Detect *walls* to constrain pose estimate
- Detect *corners* to recognize places ("loop closure")
- Estimate speed by observing how the environment changes
- …



Simple loop closure



Sensor data needs to be broken down into *features*. All sensors provide data that contains features.

What features could we extract from Sparki?

- Accelerometer
- Magnetometer
- Rate gyroscope
- Floor sensor
- Ultrasound sensor
- IR receiver
- Light sensor

Other features

- Being picked up, falling of the table, …
- N, S, E, W
- Rotating / not rotating
- Cross on the floor, …
- Lines, corners, …
- In kitchen, in living room, …
- Light on / light off

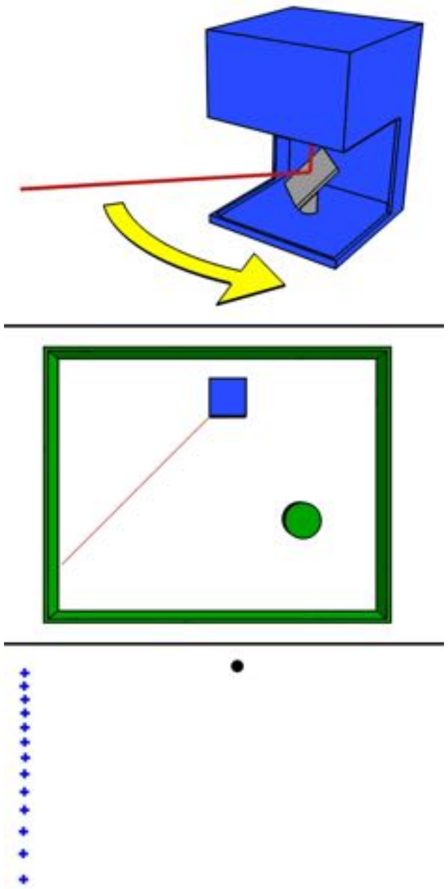
Writing code to extract such features often is hard.
Reliability of detection will be treated with later…

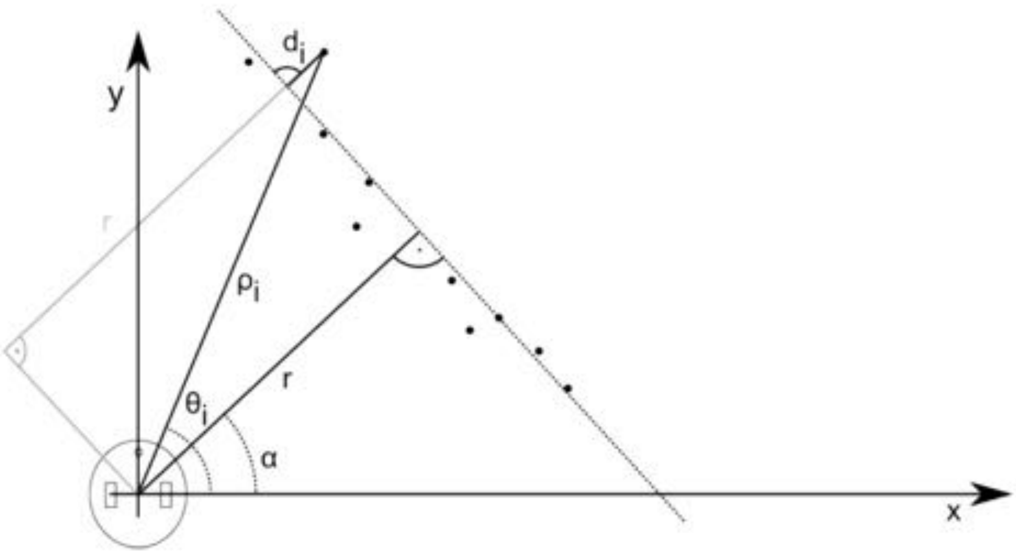
Today

- Extracting line and corner features
- Least-squares
- RANSAC
- Split-and-merge algorithm

Obvious applications to mapping and navigation, but very, very general algorithms

Lines





$$\rho_i \cos(\theta_i - \alpha) - r = d_i.$$

$$S_{r,\alpha} = \sum_i d_i^2 = \sum_i (\rho_i \cos(\theta_i - \alpha) - r)^2$$

How to find the optimal parameters alpha and r?

“Least-squares”

$$S_{r,\alpha} = \sum_i d_i^2 = \sum_i (\rho_i \cos(\theta_i - \alpha) - r)^2$$

$$\frac{\partial S}{\partial \alpha} = 0 \qquad \frac{\partial S}{\partial r} = 0$$

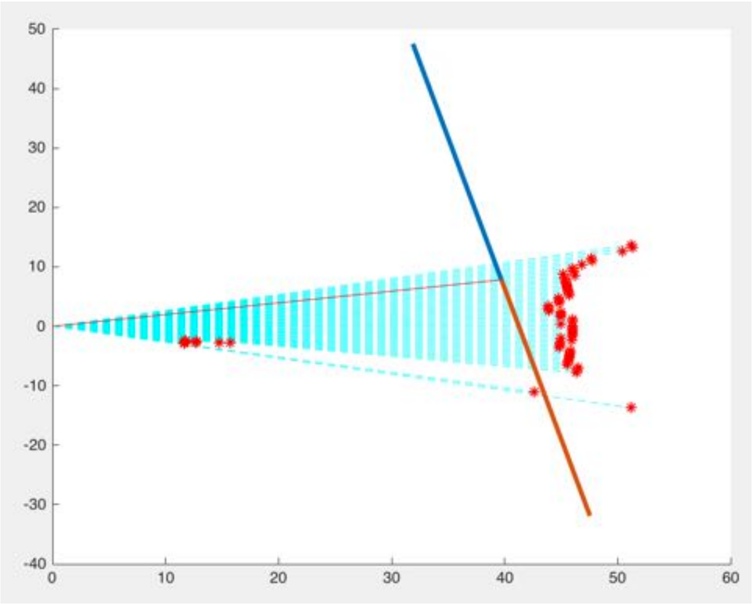
$$\alpha = \frac{1}{2} atan \left(\frac{\frac{1}{N}\sum \rho_i^2 sin2\theta_i - \frac{2}{N^2}\sum\sum \rho_i\rho_j cos\theta_i sin\theta_j}{\frac{1}{N}\sum \rho_i^2 cos2\theta_i - \frac{1}{N^2}\sum\sum \rho_i\rho_j cos(\theta_i + \theta_j)} \right)$$

$$r = \frac{\sum \rho_i cos(\theta_i - \alpha)}{N}$$

Matlab demo

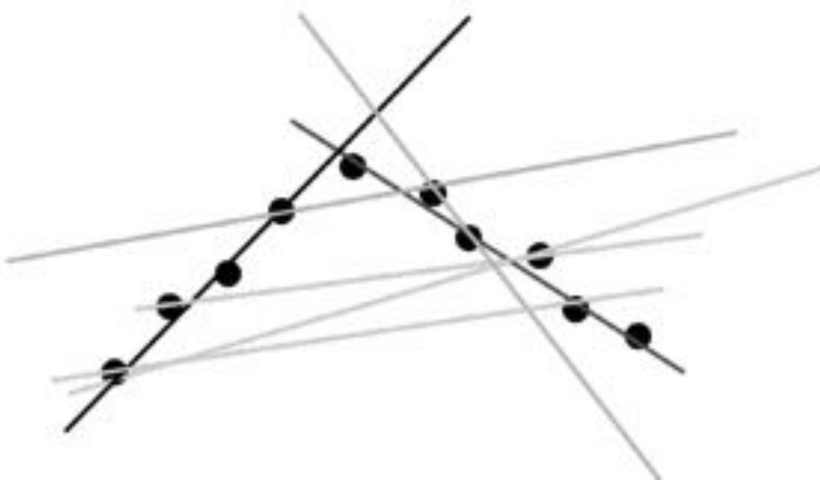
Problems

- Big problems with noise
- Computational complexity quite high $O(N^2)$
- Cannot deal with multiple lines at once
- Solution: appropriate segmentation



Ultrasound data from Sparki
(the "line" was perfect)

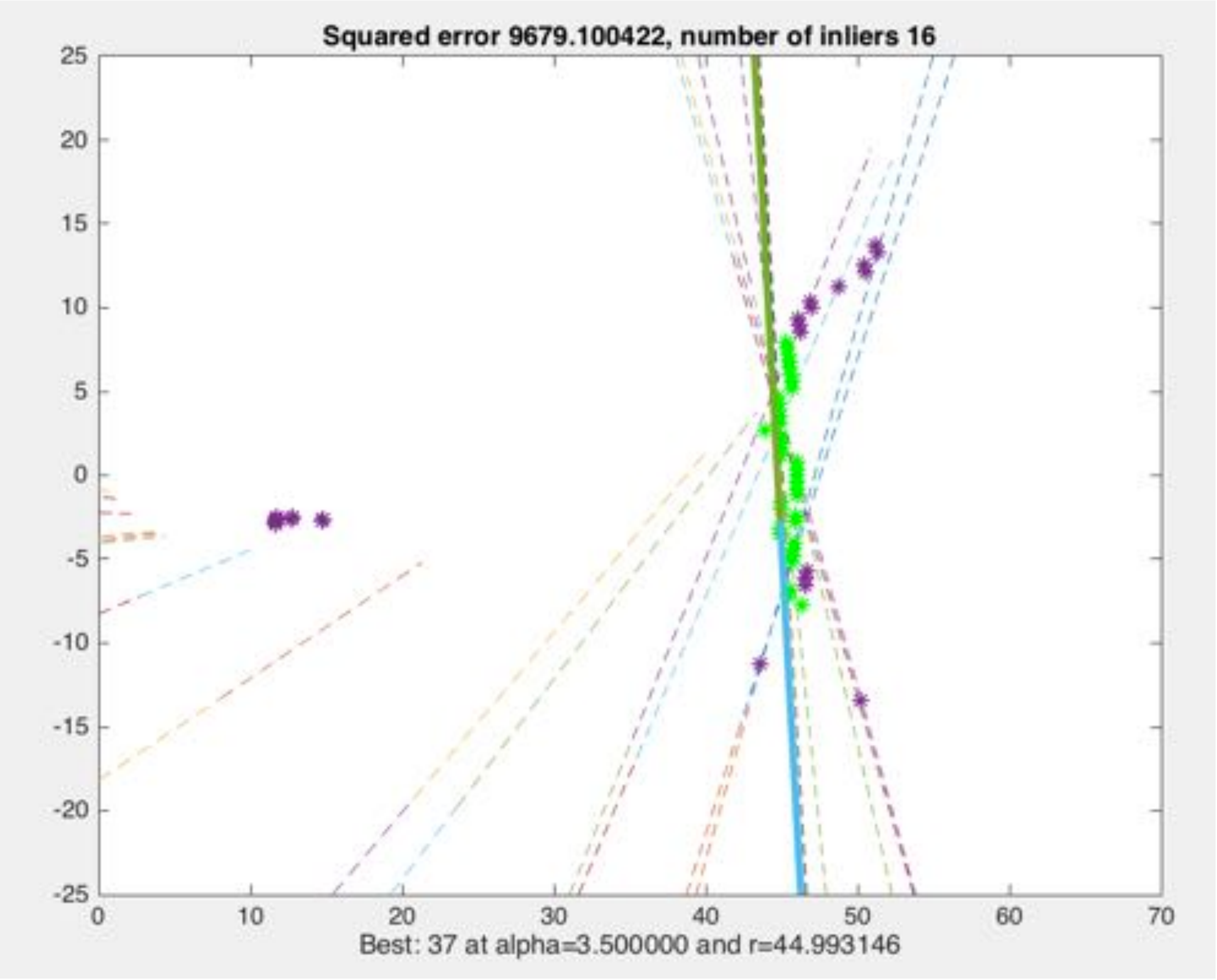
Random Sample and Consensus (RANSAC)



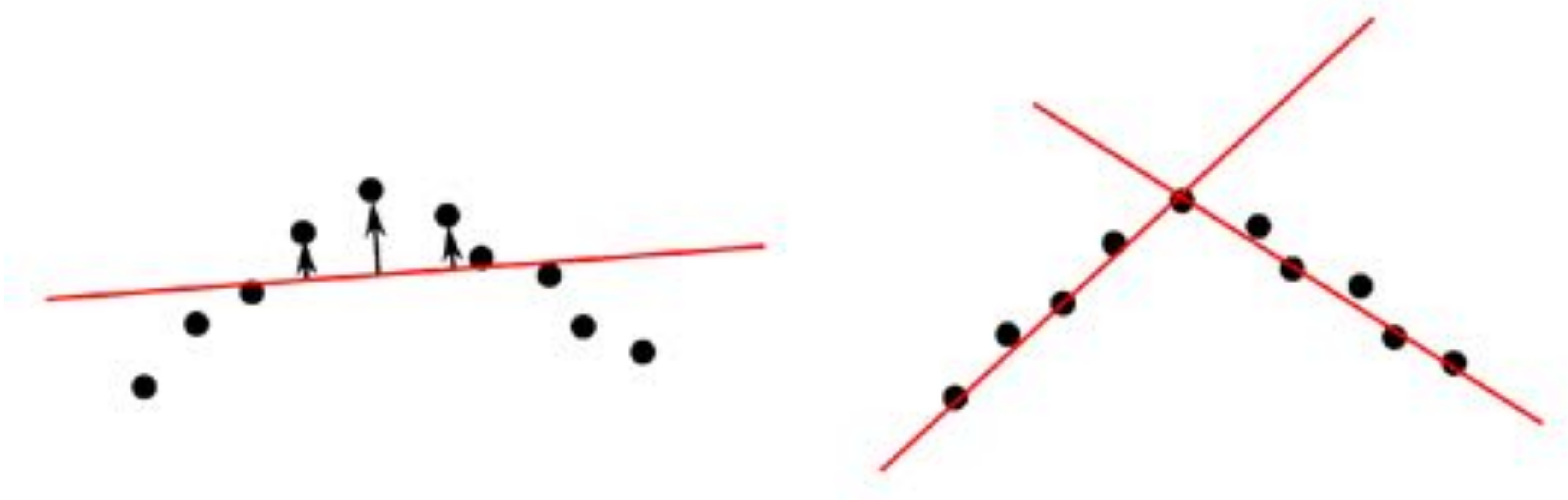
1. Sample random line from two points
2. Calculate number of "inliers"
3. Check if better than previous
   1. Yes: store
   2. No: discard
4. Repeat X times

Inliers are points sufficiently close to a line

Example


Squared error 9679.100422, number of inliers 16
25
20
15
10
5
0
-5
-10
-15
-20
-25
0
10
20
30
40
50
60
70
Best: 37 at alpha=3.500000 and r=44.993146

Split-and-Merge Algorithm



1. Select point with highest error
2. Split dataset at this location

Other applications of RANSAC

- Image stitching (iPhone panorama function)
- Map alignment
- Data clustering
- Any regression
- …

Summary

- Features are a smart way to reduce data coming from sensors
- Features are task-relevant high-level information
  - Location of lines
  - Location of corners
  - Location of objects
  - …
- Least-squares gives optimal solutions
- RANSAC deals with outliers
- Feature extraction is an optimization problem with a probabilistic outcome