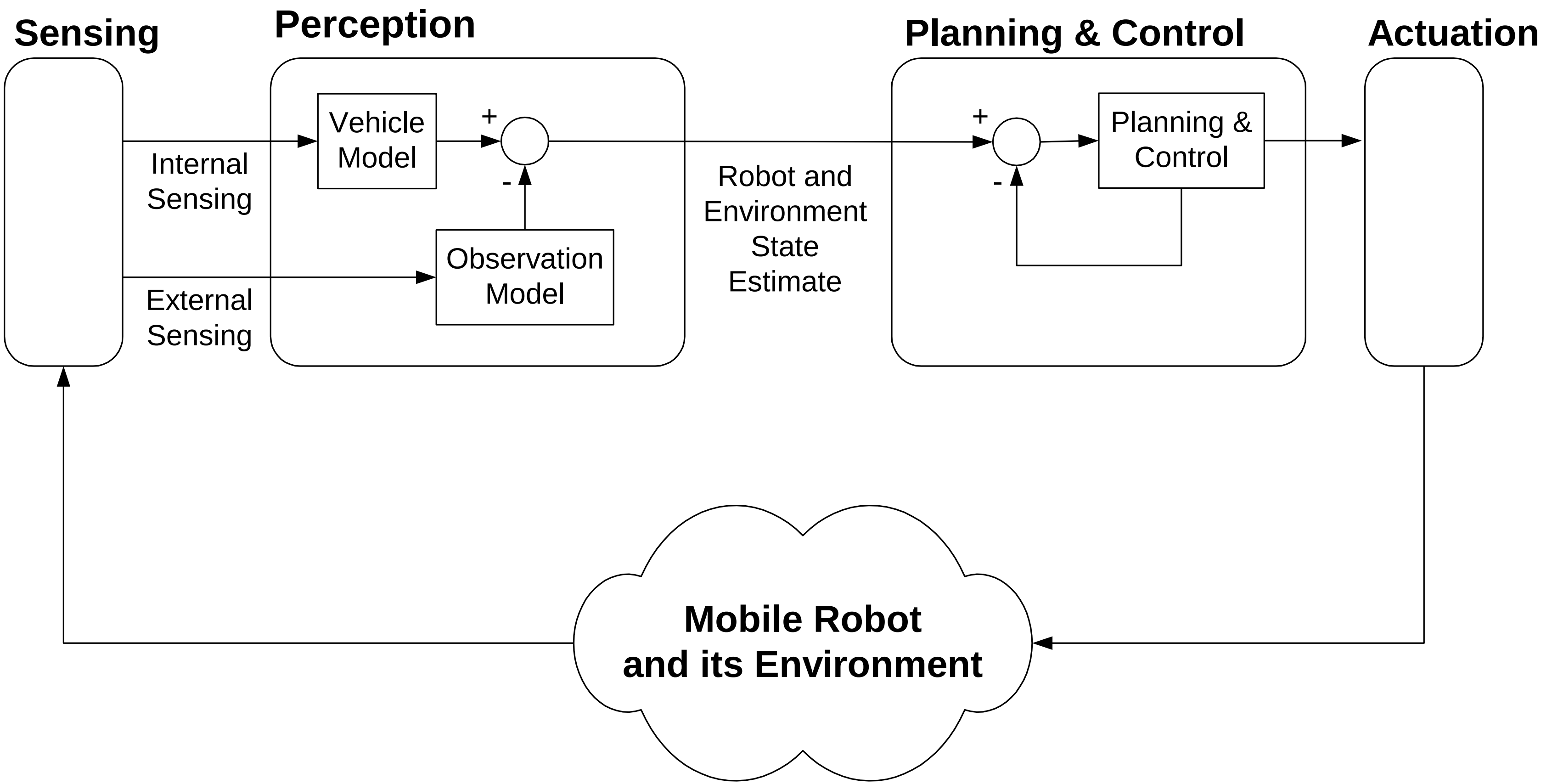
UTS:CAS
CENTRE FOR AUTONOMOUS SYSTEMS

49274 ADVANCED ROBOTICS SENSORS

TERESA VIDAL CALLEJA

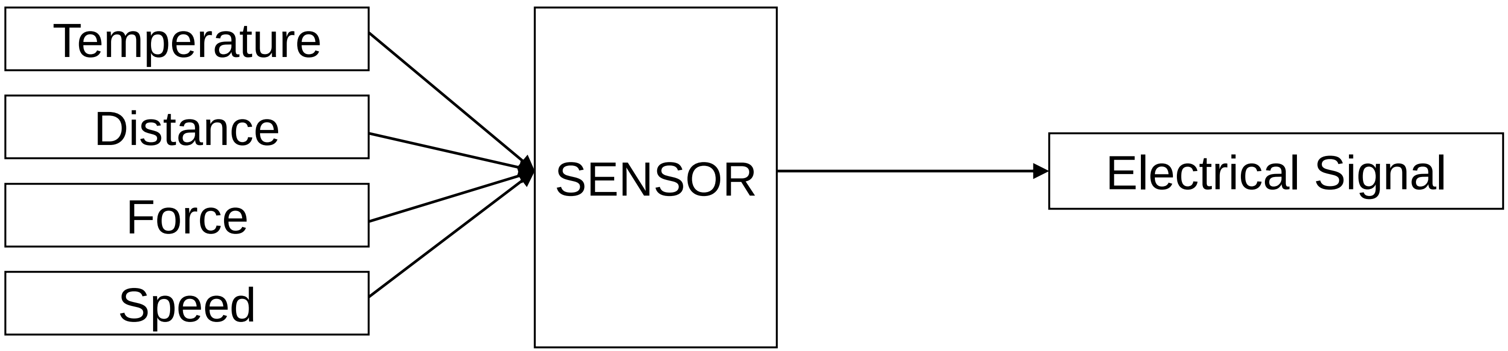
UNIVERSITY OF TECHNOLOGY SYDNEY

MOBILE ROBOT ARCHITECTURE

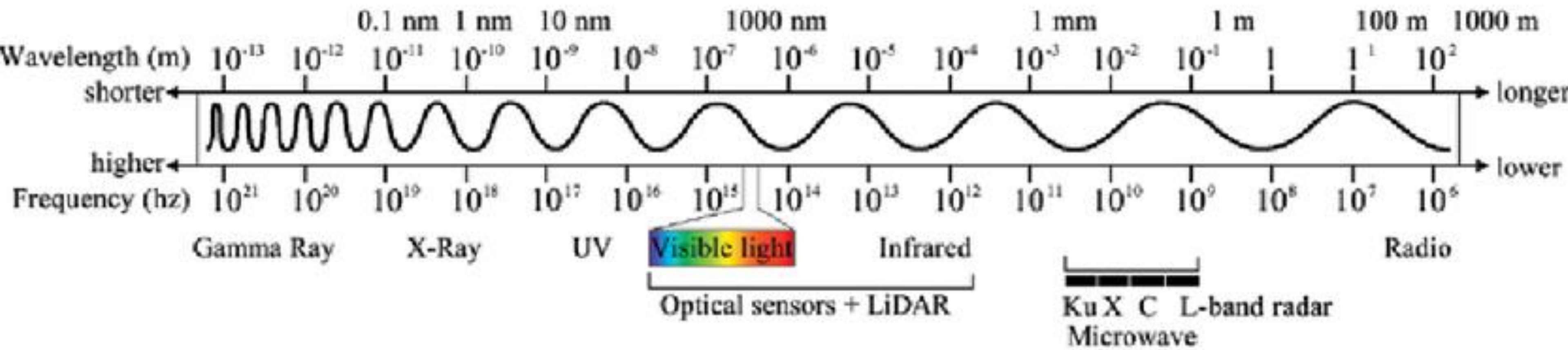


SENSORS

- A device for measuring some physical quantity
- The sensor usually converts from the measurement space to an electrical signal



ELECTROMAGNETIC SPECTRUM



DEFINITIONS

Accuracy: The agreement between the actual value and the measured value

Resolution: The change in measured variable to which the sensor will respond

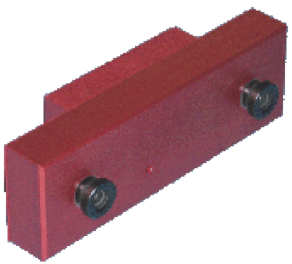
Repeatability: Variation of sensor measurements when the same quantity is measured several times

Range: Upper and lower limits of the variable that can be measured

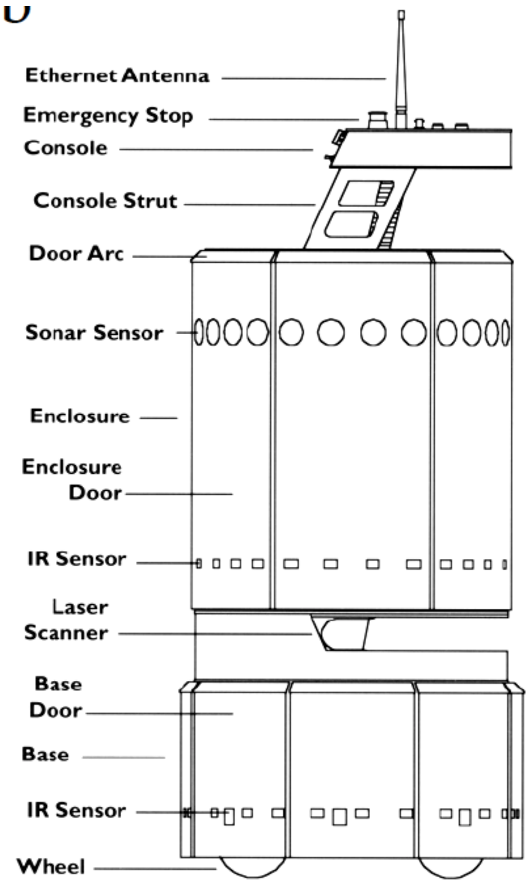
Sensitivity: How the measured value is affected by the environment (temperature, etc)

Linearity: How linear is the quantity

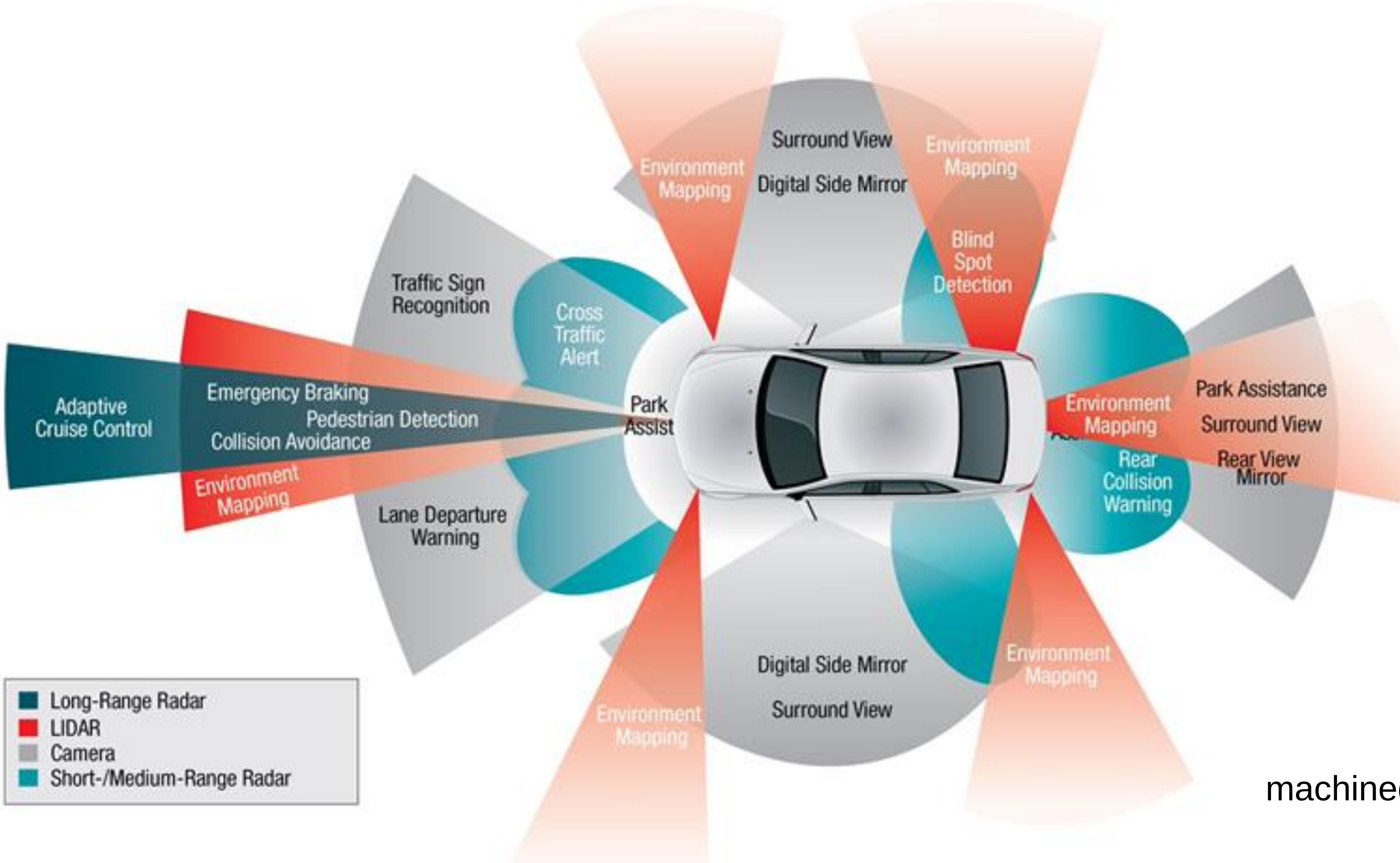
SENSORS



SENSORS IN MOBILE ROBOTS



SENSORS IN AUTONOMOUS CARS



machinedesign.com

SENSOR CLASSIFICATION

Proprioceptive sensors measure values internal to the system (robot)

motor speed, wheel load, robot arm joint angles, battery voltage

Exteroceptive sensors acquire information from the robot’s environment

distance measurements, light intensity, sound amplitude

Passive sensors measure ambient environmental energy entering the sensor

Active sensors emit energy into the environment, then measure the environmental reaction

SENSORS CLASSIFICATION

General Classification (typical use)	Sensor Sensor System	PC: Propriocep. EC: Exteroceptive	P: Passive A: Active
Wheel/motor sensors (wheel/motor speed and position)	Brush Encoders	PC	P
	Potentiometers	PC	P
	Synchros, Resolvers	PC	A
	Optical Encoders	PC	A
	Magnetic Encoders	PC	A
	Inductive Encoders	PC	A
	Capacitive Encoders	PC	A
Heading sensors (orientation of the robot in relation to a fixed reference frame)	*Compass*	EC	P
	Gyroscopes	PC	P
	Inclinometers	EC	P/A
Ground based beacons (localization in a fixed reference frame)	*GPS*	EC	A
	Active optical or RF beacons	EC	A
	Active ultrasonic beacons	EC	A
	Reflective beacons	EC	A
Active ranging (reflectivity, time-of-flight and geometric triangulation)	Reflectivity sensors	EC	A
	Ultrasonic sensor	EC	A
	Laser rangefinder	EC	A
	Optical triangulation (1D)	EC	A
	Structured light (2D)	EC	A
Motion/speed sensors (speed relative to fixed or moving objects)	*Doppler radar*	EC	A
	Doppler sound	EC	A
Vision-based sensors (visual ranging, whole-image analysis, segmentation, object recognition)	*CCD/CMOS camera(s)*	EC	P
	Visual ranging packages		
	Object tracking packages		

SENSORS - OUTLINE

- Lidar
- Ultrasound (sonar)
- Cameras
- 3D Cameras
- Encoders

SENSORS - OUTLINE

- Lidar
- Ultrasound (sonar)
- Cameras
- 3D Cameras
- Encoders

Time of Flight

Intensity-based

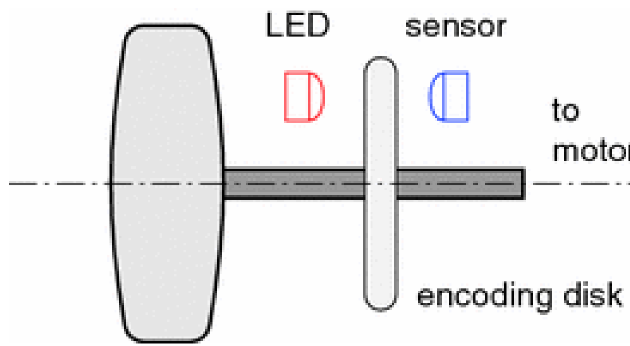
proprioceptive

ENCODERS

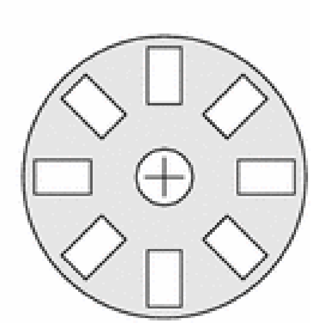
Encoders are digital sensors commonly used to provide position feedback for actuators

Consist of a glass or plastic disc that rotates between a light source (LED) and a pair of photo-detectors

Disk is encoded with alternate light and dark sectors so pulses are produced as disk rotates



Optical wheel encoder



Encoding disk

ENCODER TYPES

Incremental encoders

Pulses from LEDs are counted to provide rotary position

Two detectors are used to determine direction (quadrature)

Index pulse used to denote start point

Otherwise pulses are not unique

Absolute encoders

Those have a unique code that can be detected for every angular position

Often in the form of a “gray code”; a binary code of minimal change

Absolute encoders are much more complex and expensive than incremental encoders

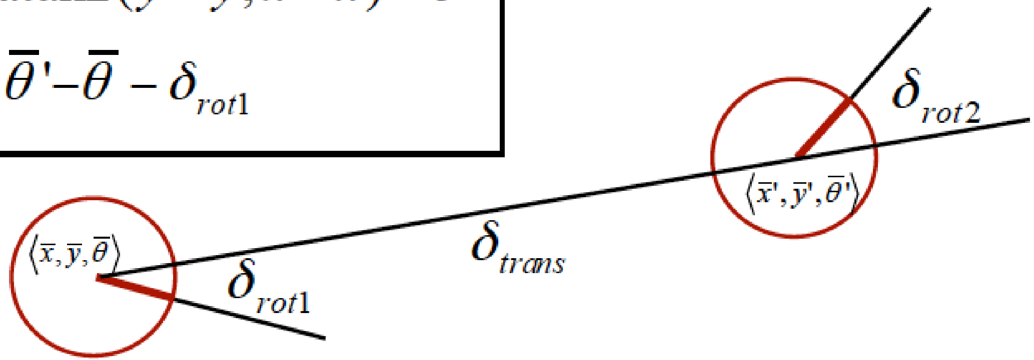
ENCODERS - ODOMETRY

Odomety Model

Robot moves from $\langle \bar{x}, \bar{y}, \bar{\theta} \rangle$ to $\langle \bar{x}', \bar{y}', \bar{\theta}' \rangle$.

Odometry information $u = \langle \delta_{rot1}, \delta_{rot2}, \delta_{trans} \rangle$.

$$\boxed{\begin{aligned} \delta_{trans} &= \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2} \\ \delta_{rot1} &= \text{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta} \\ \delta_{rot2} &= \bar{\theta}' - \bar{\theta} - \delta_{rot1} \end{aligned}}$$



ENCODERS - ODOMETRY

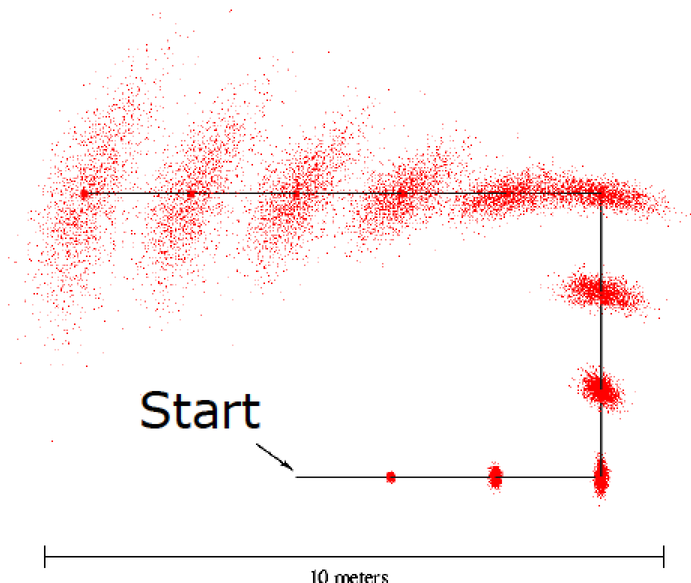
Noise model

The measured motion is given by the true motion corrupted with noise.

$$\hat{\delta}_{rot1} = \delta_{rot1} + \varepsilon_{\alpha_1 |\delta_{rot1}| + \alpha_2 |\delta_{trans}|}$$

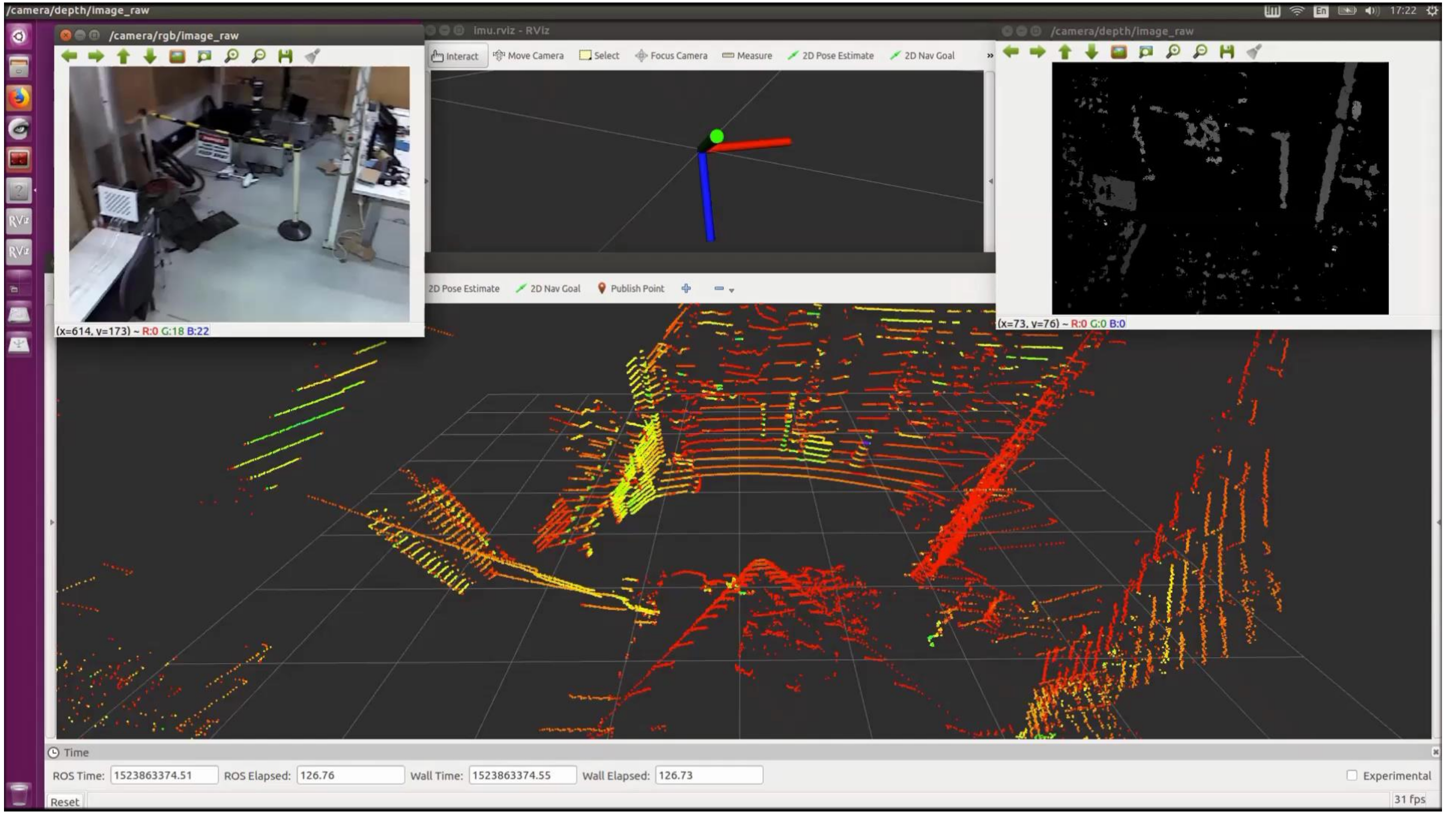
$$\hat{\delta}_{trans} = \delta_{trans} + \varepsilon_{\alpha_3 |\delta_{trans}| + \alpha_4 |\delta_{rot1} + \delta_{rot2}|}$$

$$\hat{\delta}_{rot2} = \delta_{rot2} + \varepsilon_{\alpha_1 |\delta_{rot2}| + \alpha_2 |\delta_{trans}|}$$



Noise modeled as Gaussian distribution

DEMO

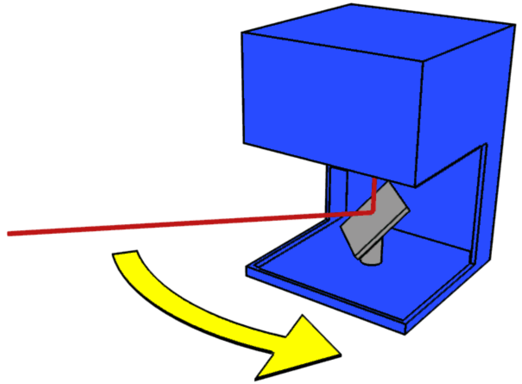

/camera/depth/image_raw
/camera/rgb/image_raw
imu.rviz - RViz
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
(x=614, y=173) ~ R:0 G:18 B:22
(x=73, y=76) ~ R:0 G:0 B:0
Time
ROS Time: 1523863374.51
ROS Elapsed: 126.76
Wall Time: 1523863374.55
Wall Elapsed: 126.73
Experimental
Reset
31 fps
17:22

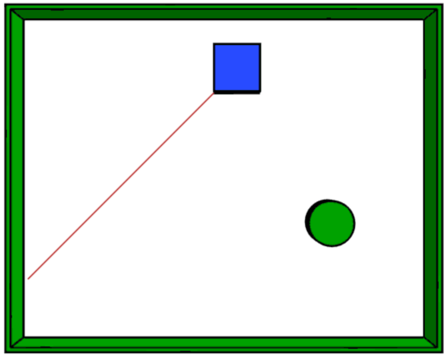
LIDAR

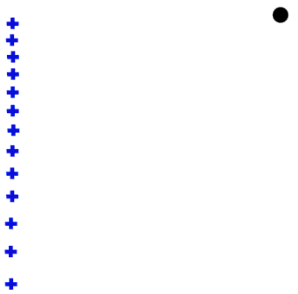
Light Detection and Ranging

Operating principal

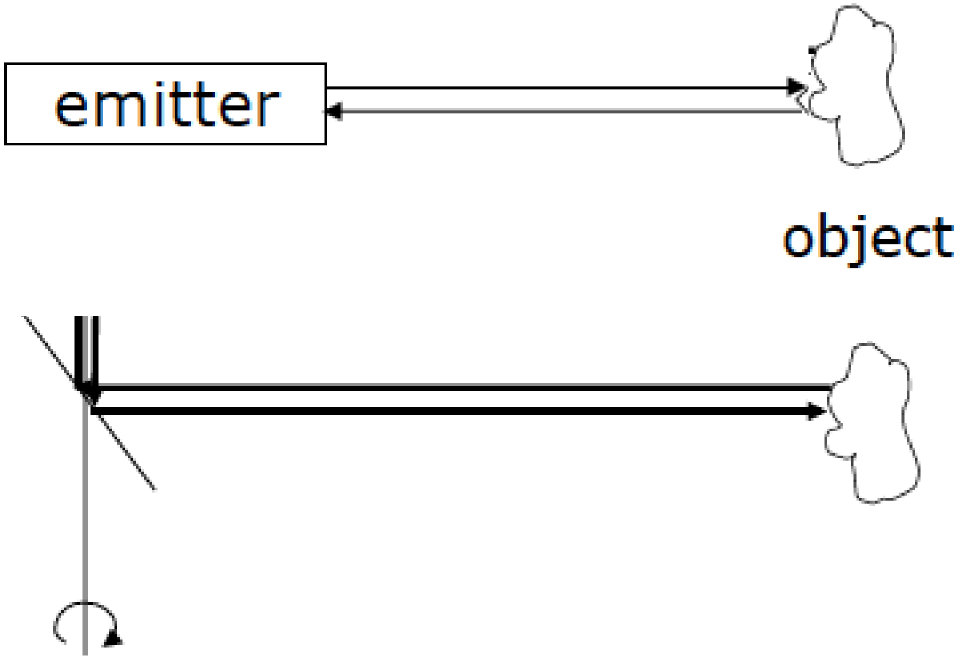
- Measuring the time of flight of laser light pulses
- The pulsed laser beam is deflected by an internal rotating mirror
- The measurement data is available in real time via serial interface







LIDAR – TIME OF FLIGHT

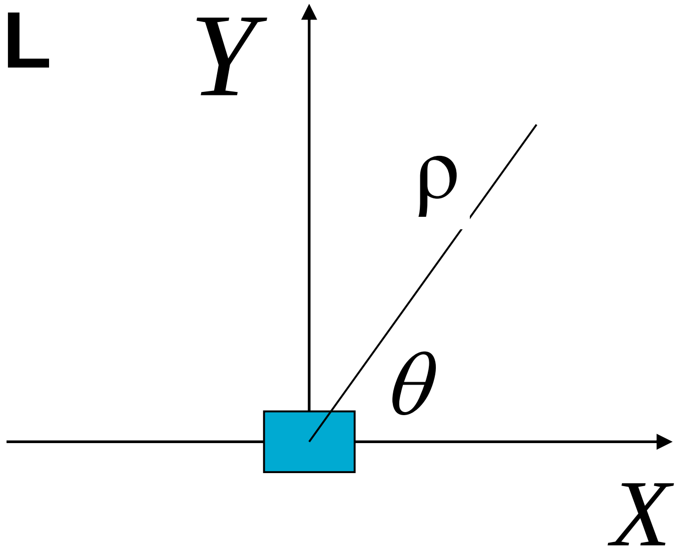


High precision

Wide field of view

Distance = (Speed of Light x Time of Flight) / 2

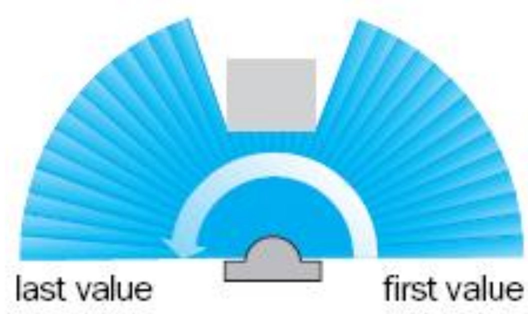
2D LIDAR SENSOR MODEL



- 2D laser sensor provides range and bearing to an object
  - Accurate range
  - Accurate bearing

$$x = \rho \cos q$$

$$y = \rho \sin q$$



LIDAR MAPPING WITH ENCODERS-ONLY

Dirk Hahnel

DATA INTERPRETATION

Range and Bearing

- How can an algorithm extract relevant information from raw sensor data?
- How is it integrated over time?
- How can an algorithm identify already discovered information (data association problem)?

Relevant problems

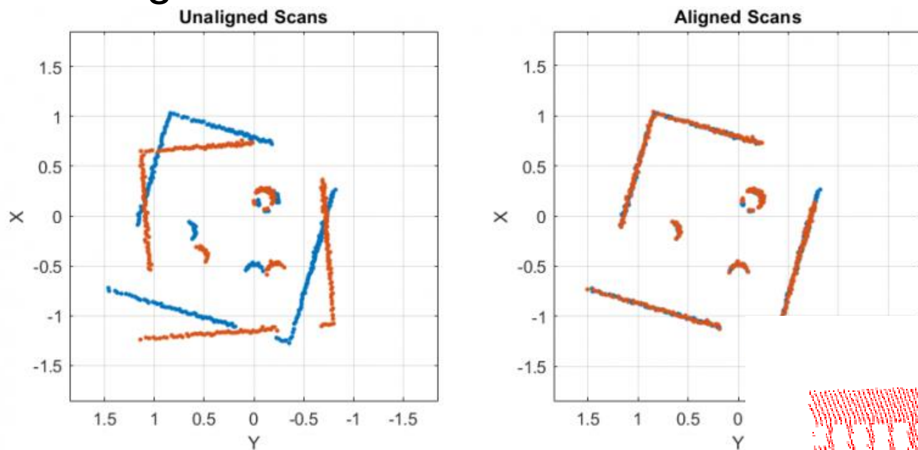
- Mapping
- Localisation
- Tracking

SCAN MATCHING

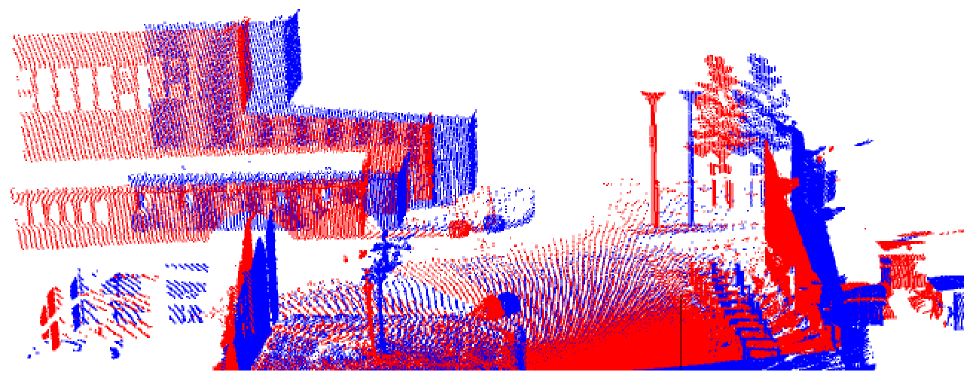
- Raw data
- Iterative Closest Points (ICP)
- Scan-to-scan
- Scan-to-map
- Feature-based (points, lines, planes)
- RANSAC for outlier rejection

SCAN MATCHING

Alignment 2D lidar data

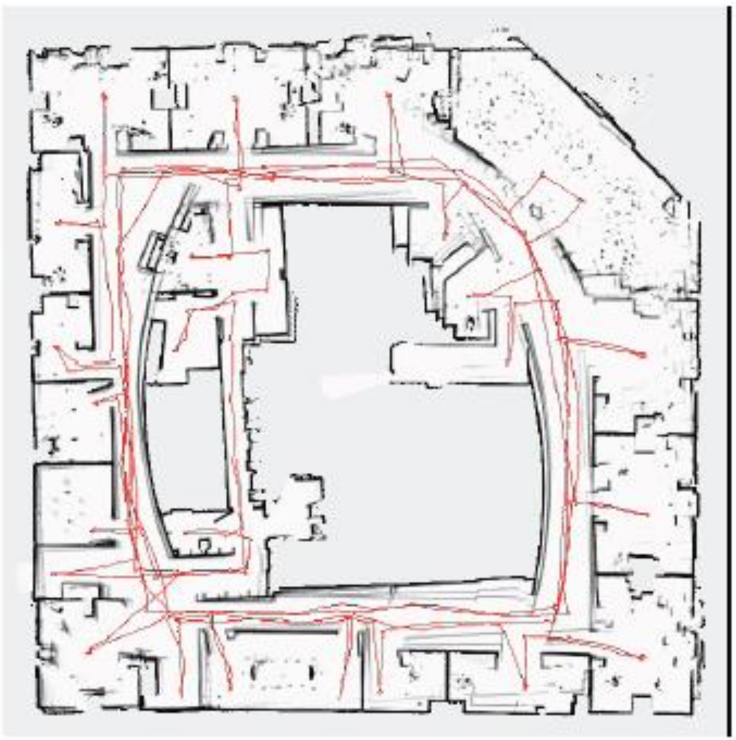


3D lidar data



SCAN MATCHING

Lidar mapping without scan matching



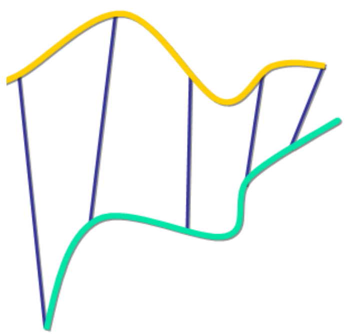
Lidar mapping with scan matching

SCAN MATCHING - PROBLEM

Given: two corresponding point sets:

$$X = \{x_1, ..., x_n\}$$

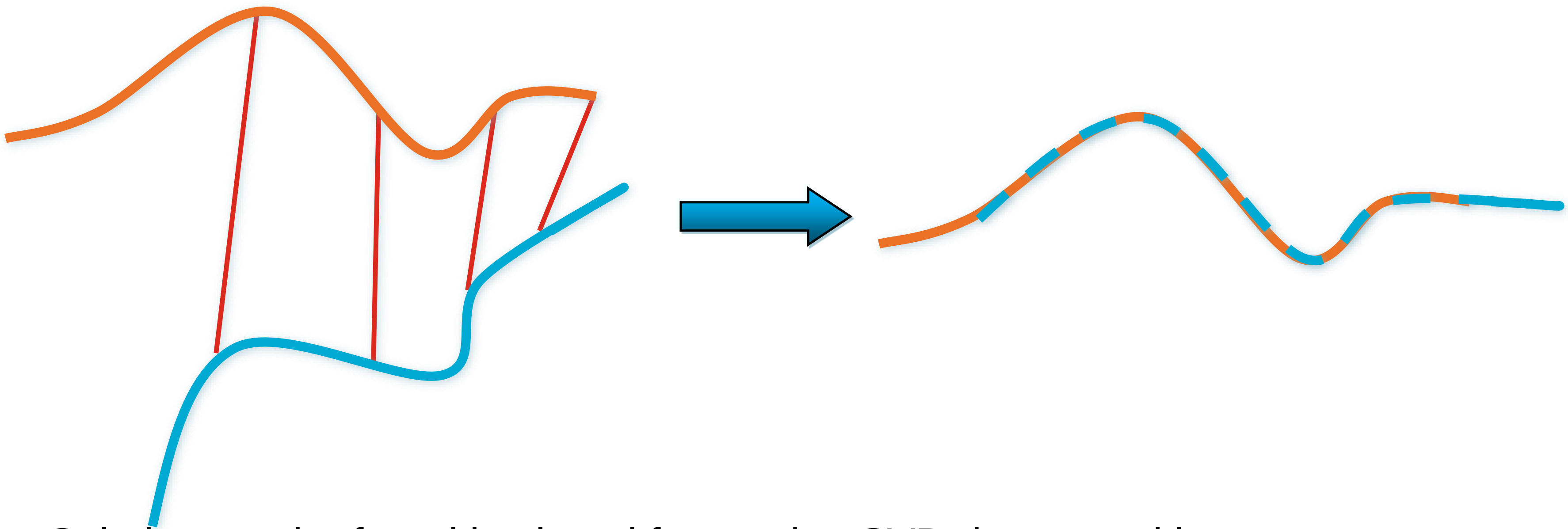
$$P = \{p_1, ..., p_n\}$$



Wanted: The Rotation R and translation t that transforms the point sets to be overlapping

SCAN MATCHING

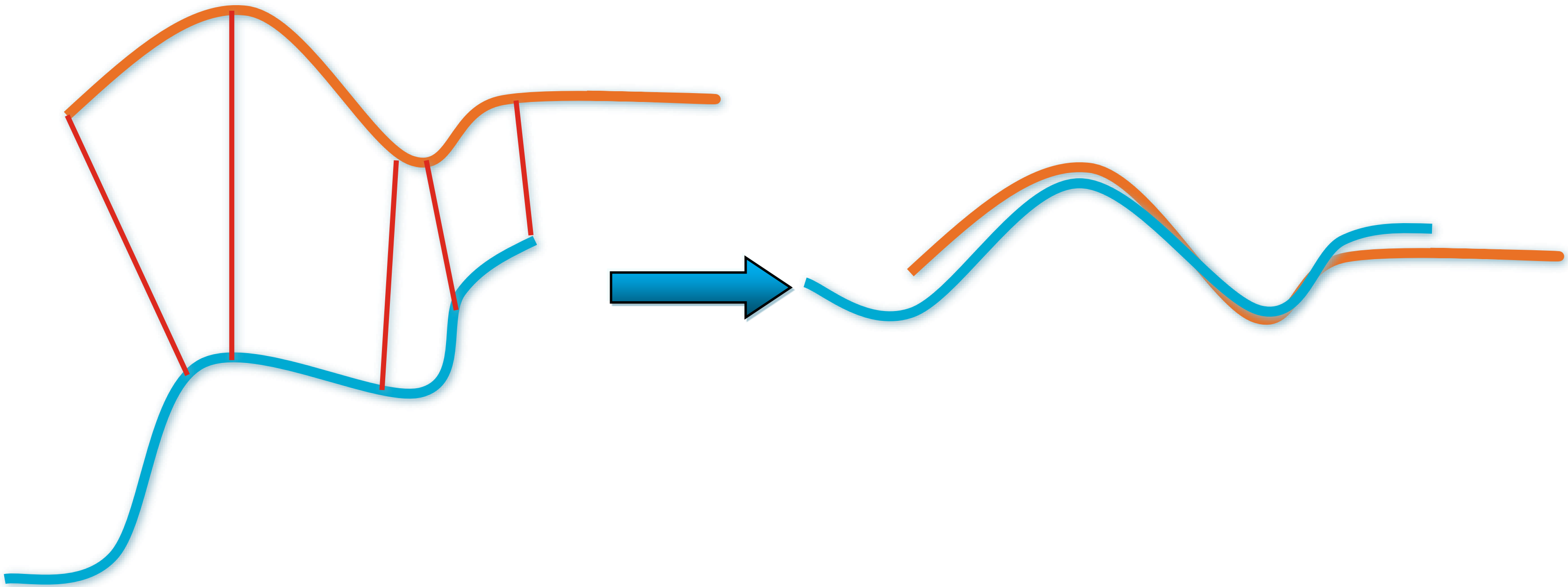
If the correct correspondences are known, can find the correct relative frame transformation (Rotation + translation)



Solution can be found in closed form using SVD decomposition

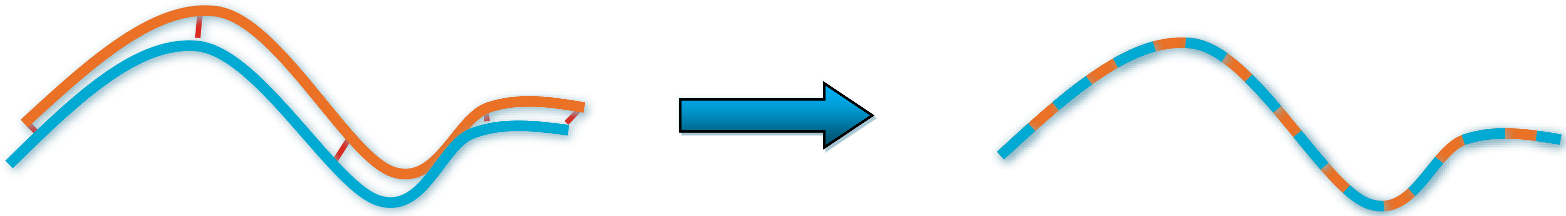
ITERATIVE CLOSEST POINTS (ICP)

- How to find correspondences: User input? Feature detection? Signatures?
- Alternative: assume closest points correspond



ICP

Converges if starting position is “close enough”



Given two corresponding point sets:

$$X = \{x_1, ..., x_n\}$$

$$P = \{p_1, ..., p_n\}$$

- Find closest points and solve
- Iterate until a given threshold in the error

We need to find the R and t that minimises the sum of the squared error:

$$E(R,t) = \frac{1}{N_p}\sum_{i=1}^{N_p} ||x_i - Rp_i - t||^2$$

Where xi and pi are corresponding points

ICP VARIANTS

Point subsets

- Point sub-sampling
- Feature based-sampling

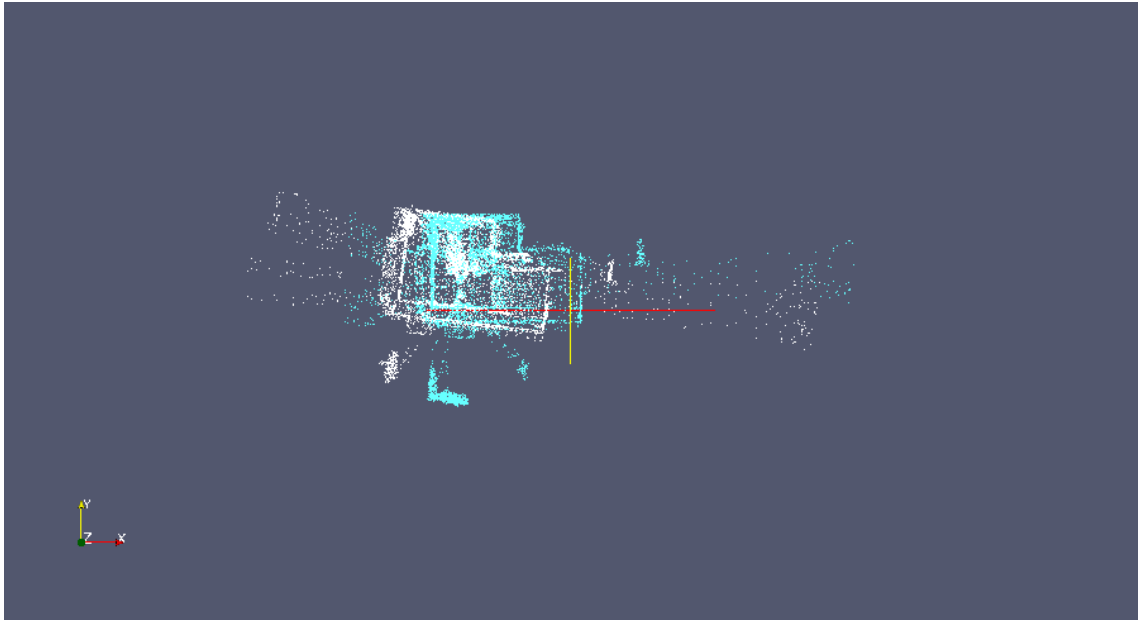
Error Metrics

- Point to plane
- Point to Line
- Plane to Plane

Outlier rejection

- RANSAC

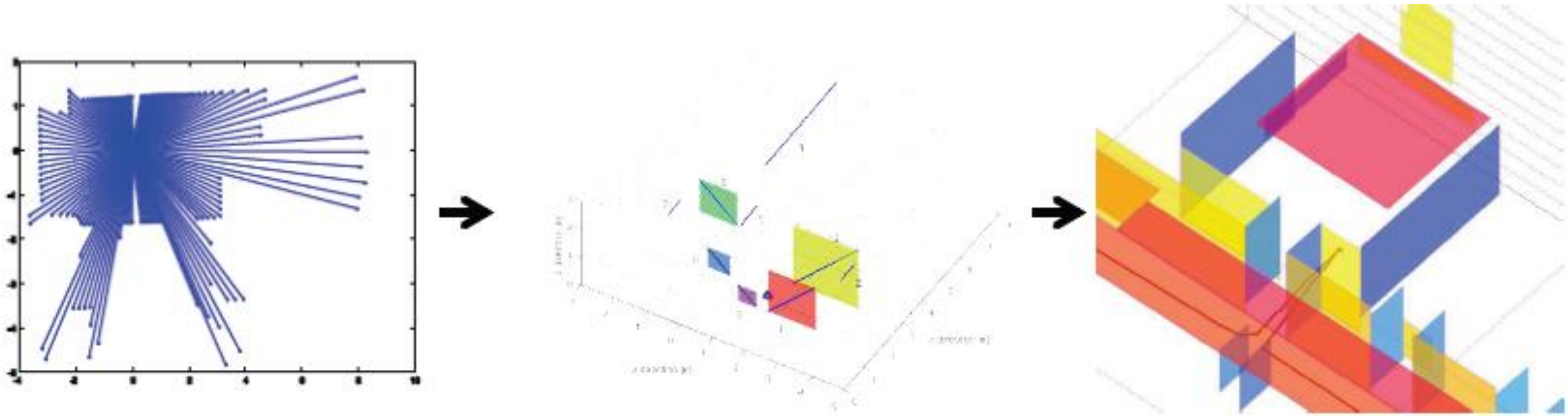
Tree search (KD-tree)



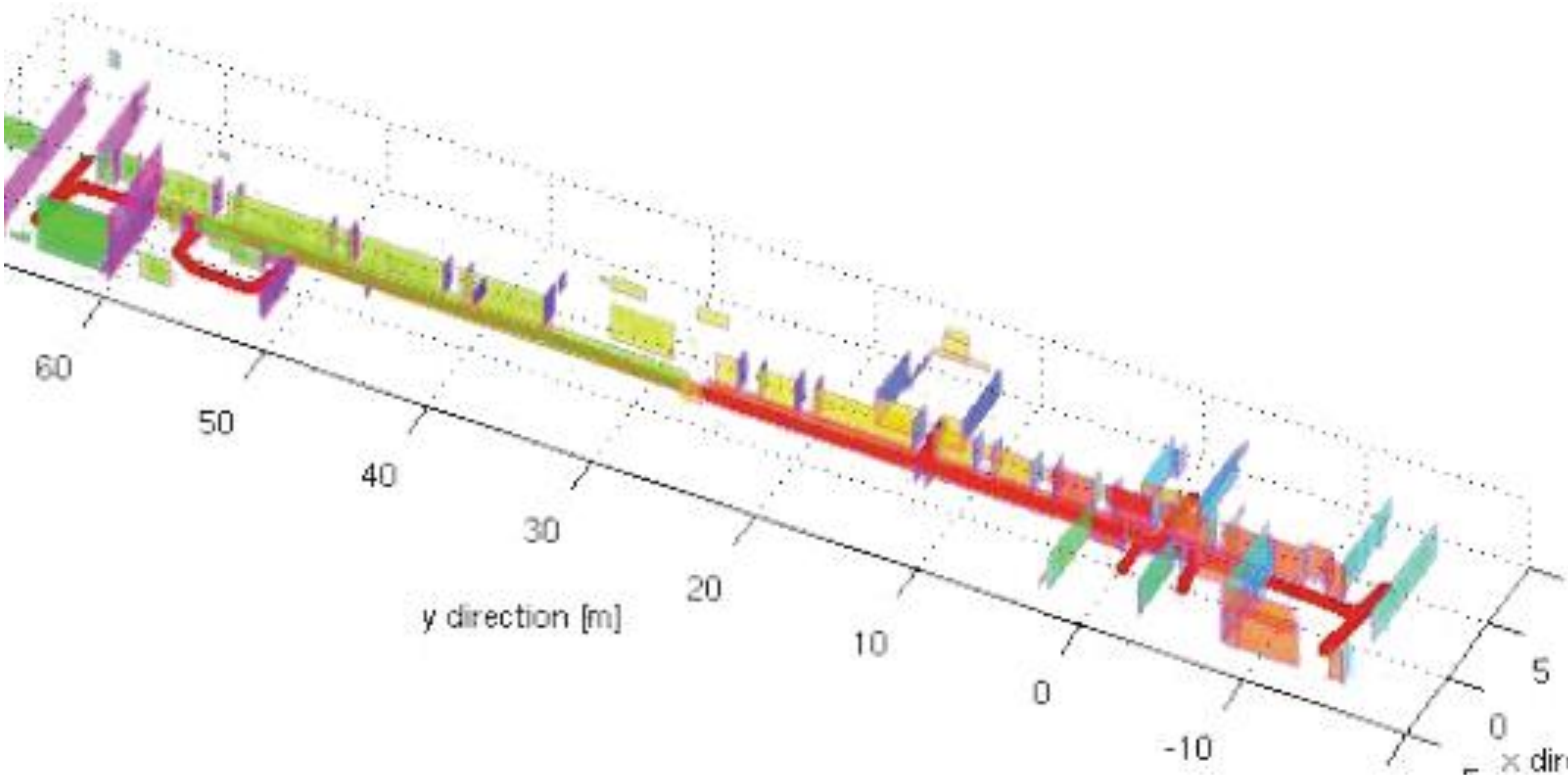
libpointmatcher.readthedocs.io

FEATURES

- Features are much more compact than raw data (all points)
- Can reflect physical or abstract objects
- Rich in information
- Can assess accuracy of feature

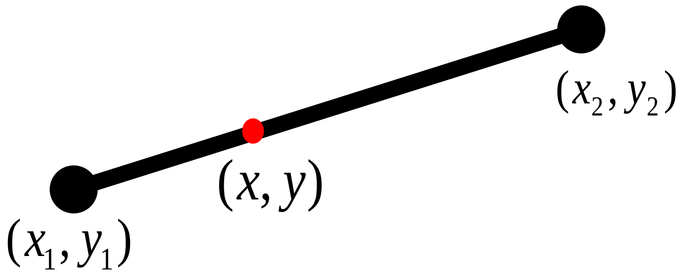


LINE EXTRACTION



LINE EXTRACTION

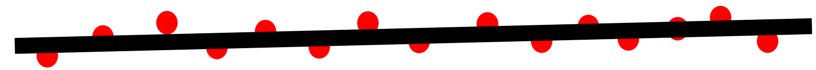
- Equation of a line



$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

$$y = mx + c$$

- Least squares solution



Line equation

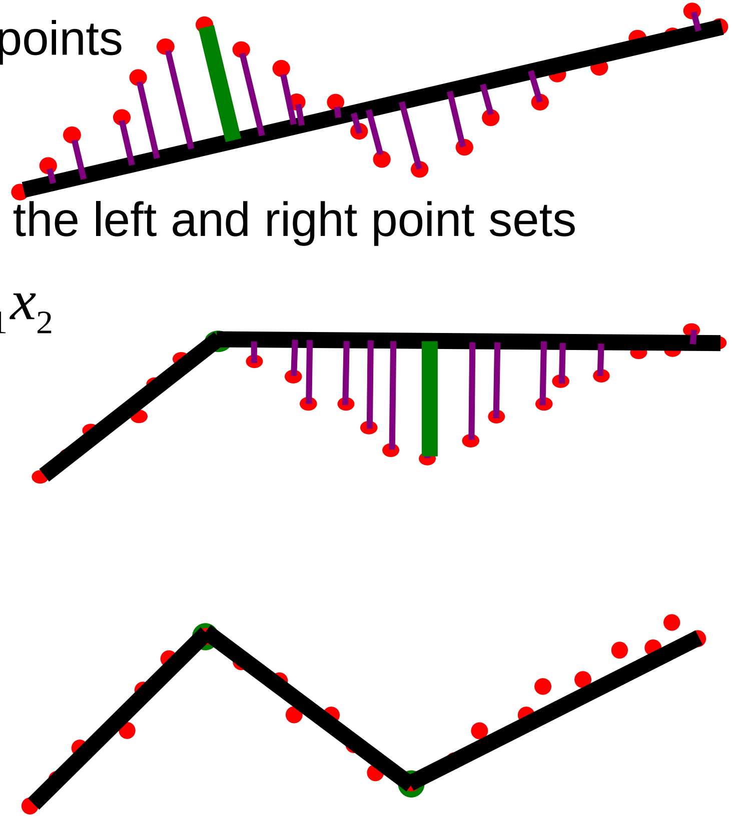
$$y - mx - c = 0$$

Error fit

$$\sum_i (y_i - mx_i - c)^2$$

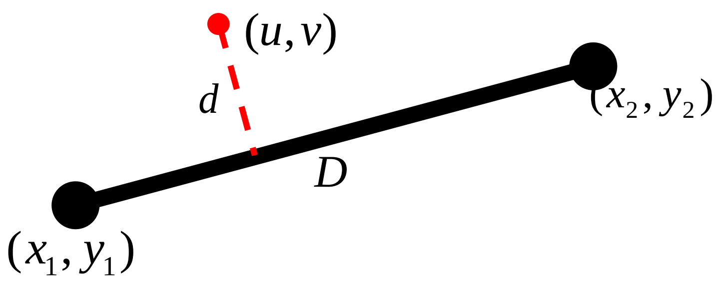
LINE SPLITING

- Obtain the line passing by the two extreme points
- Find the most distant point to the line
- If distance > threshold, split and repeat with the left and right point sets



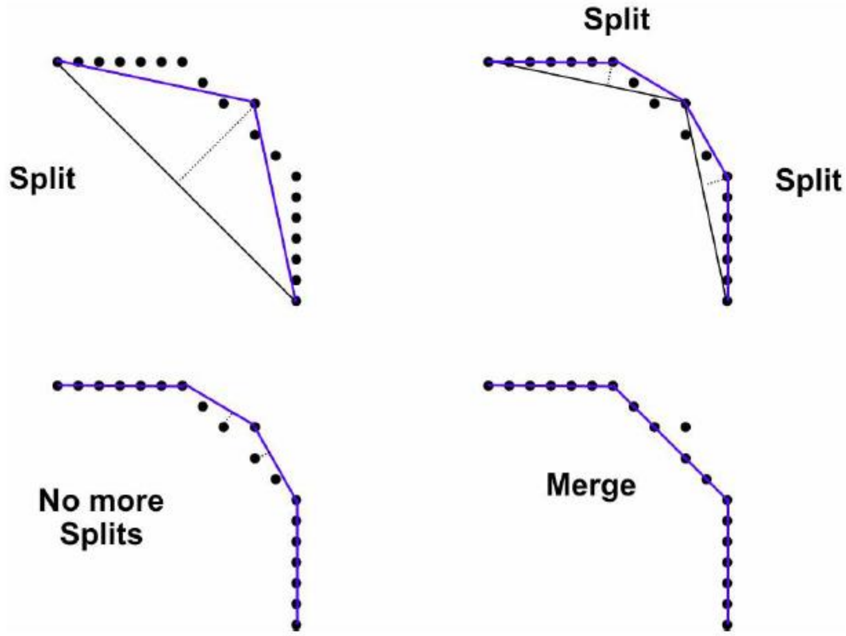
$$r = u(y_1 - y_2) + v(x_2 - x_1) + y_2 x_1 - y_1 x_2$$

$$d = \frac{r}{D}$$



SPLIT AND MERGE

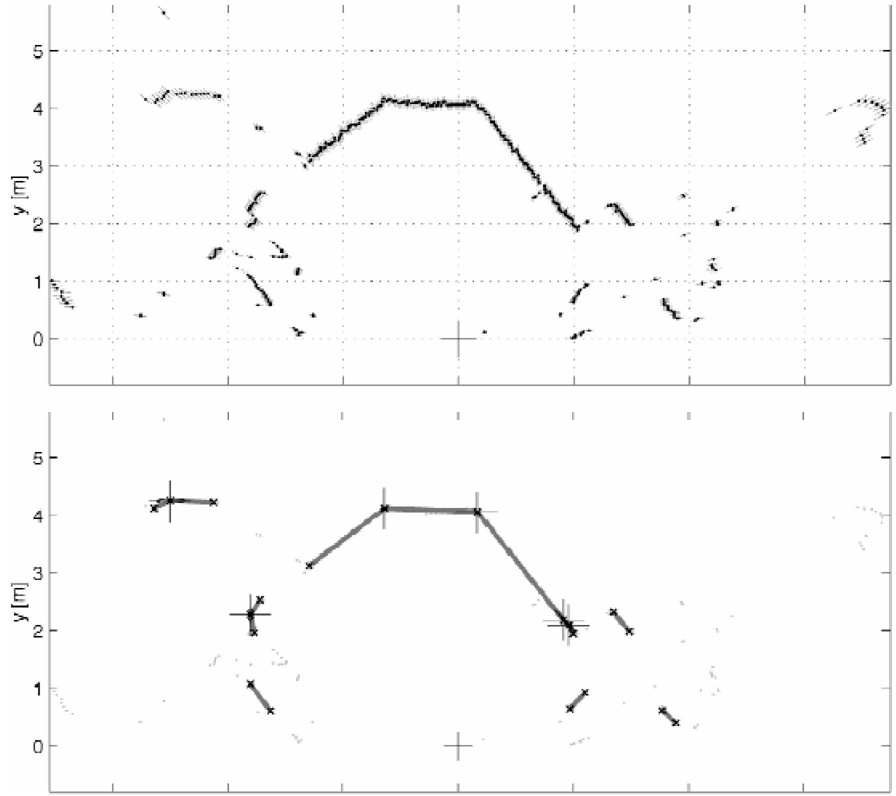
- If two consecutive segments are close/collinear enough, obtain the common line and find the most distant point
- If distance <= threshold, merge both segments



LIDAR DATA WITH LINE FITTING

Raw
range data

Line
segments



RANSAC

RANSAC = RANdomSAmpleConsensus

- A generic and robust fitting algorithm of models in the presence of outliers (i.e. points which do not satisfy a model)

- Generally applicable algorithm to any problem where the goal is to identify the inliers which satisfy a predefined model

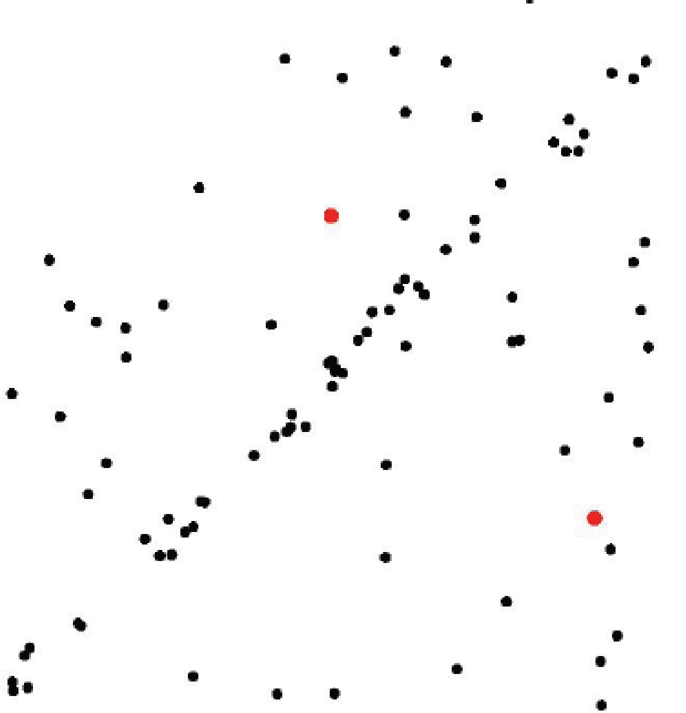
- Typical applications in robotics are: line extraction from 2D range data, plane extraction from 3D range data, feature matching…

RANSAC

- RANSAC Example

RANSAC

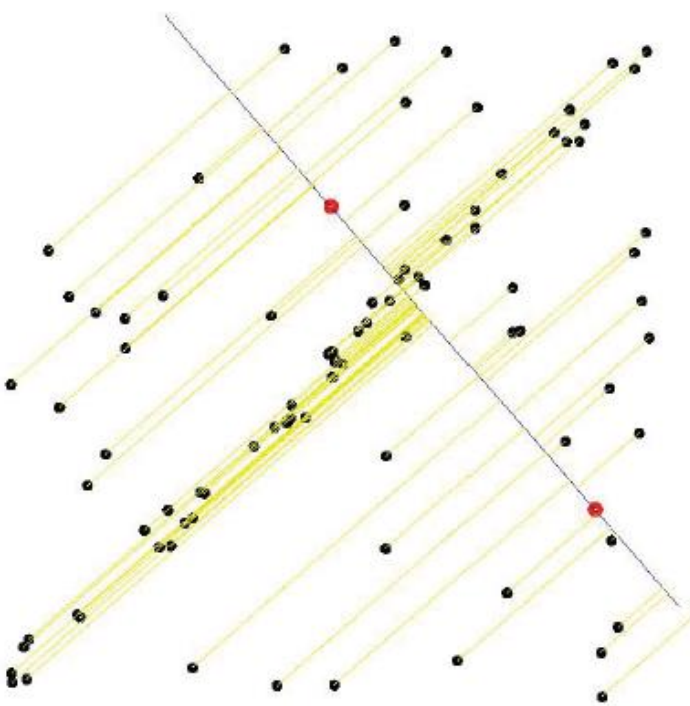
- RANSAC Example



- **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

RANSAC

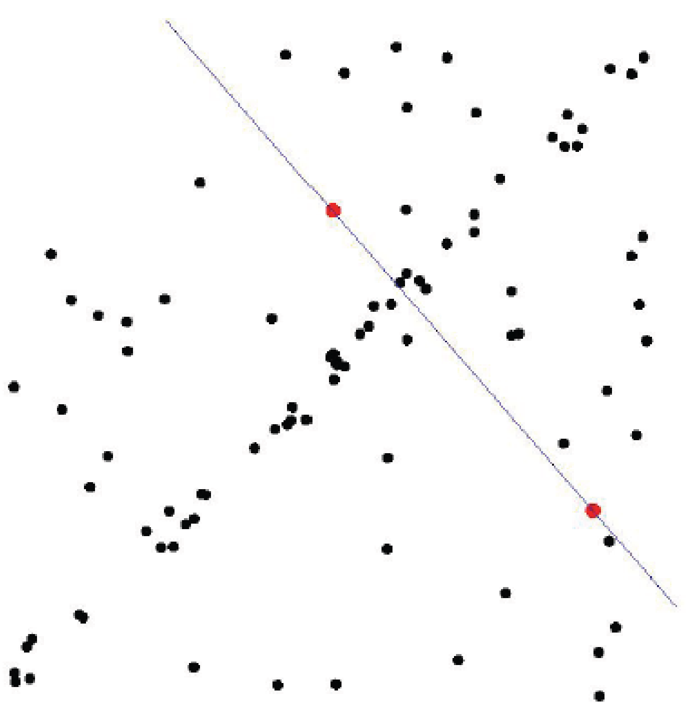
- RANSAC Example



- 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

RANSAC

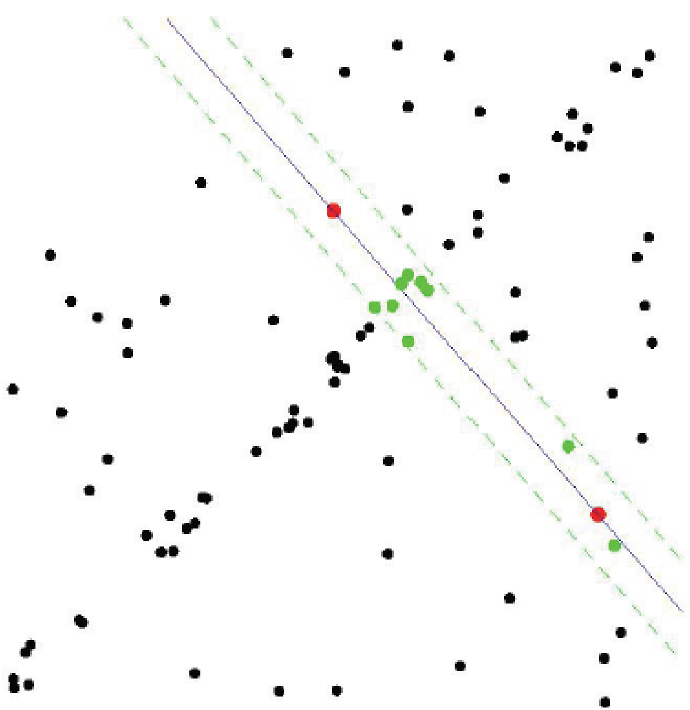
- RANSAC Example



- 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

RANSAC

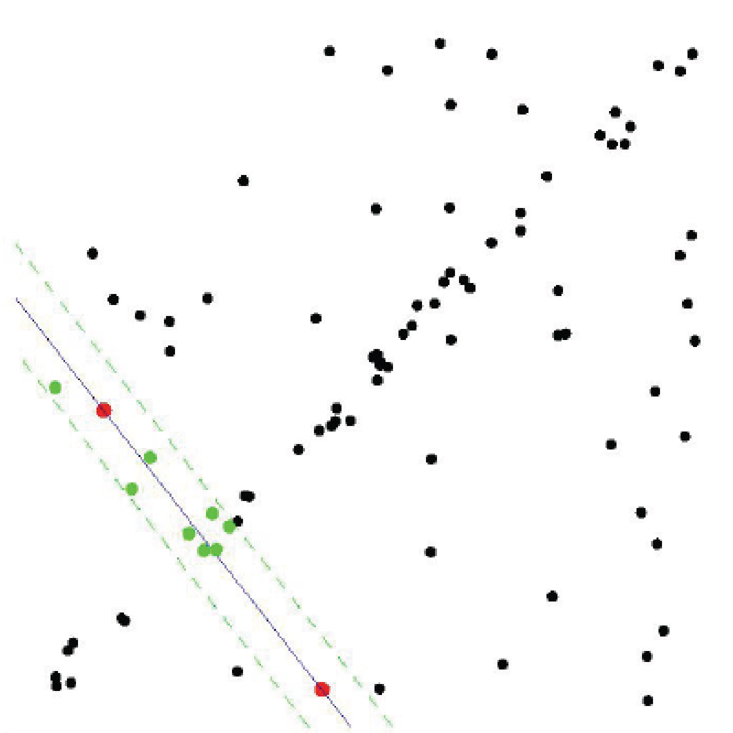
- RANSAC Example



- 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

RANSAC

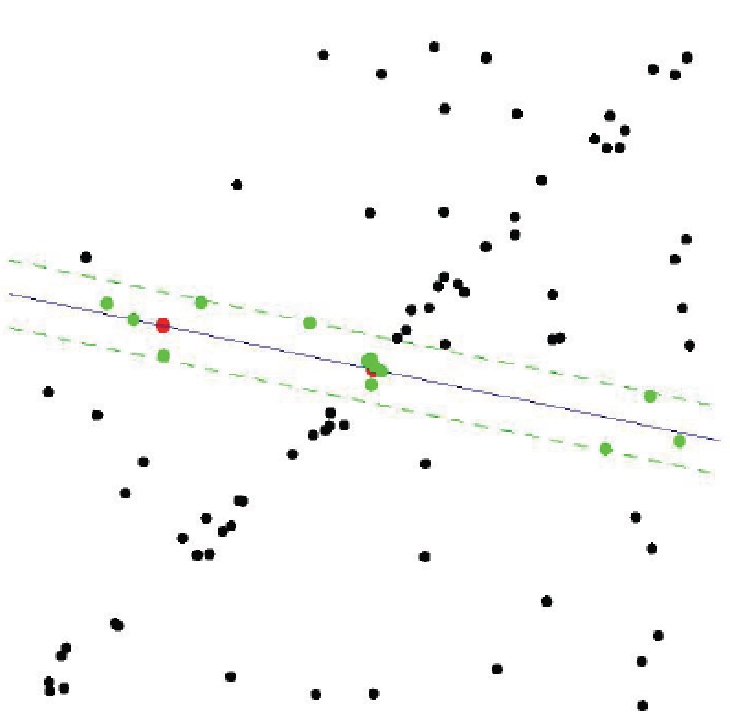
- RANSAC Example



- 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**

RANSAC

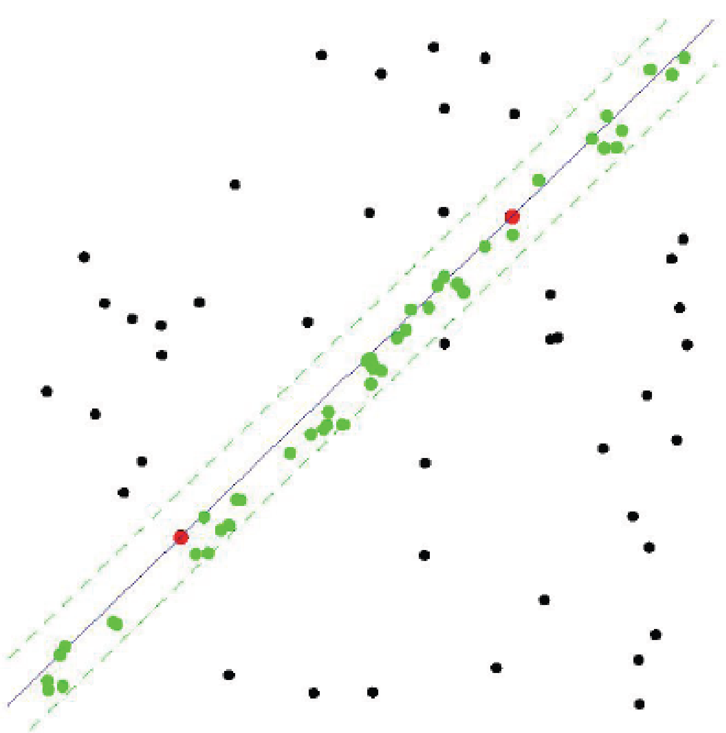
- RANSAC Example



- 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**

RANSAC

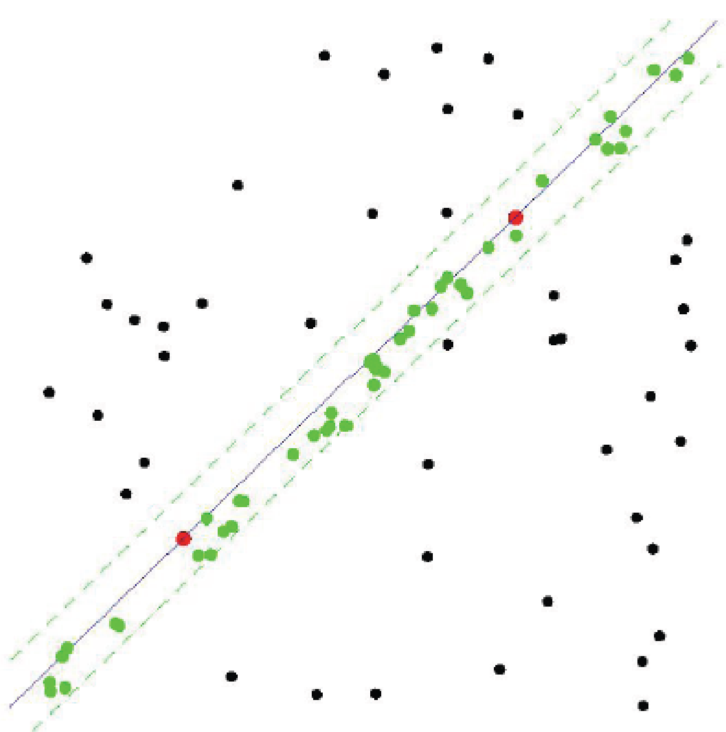
- RANSAC Example



- 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**

RANSAC

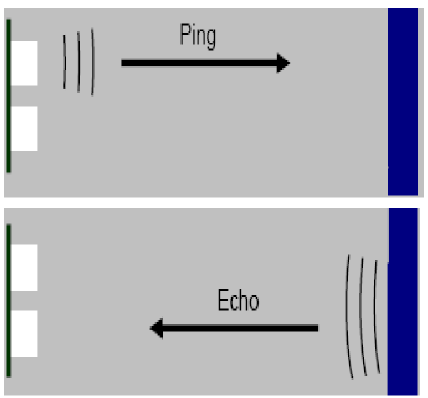
- RANSAC Example

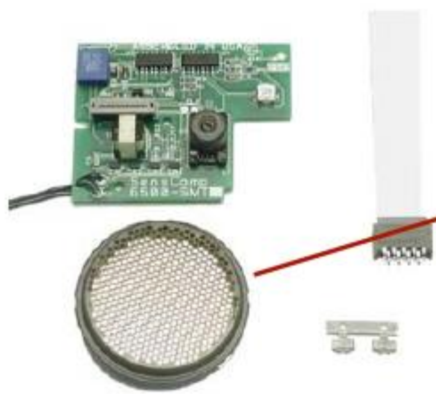


- Stop after k iterations and select model with the max number of inliers.

ULTRASOUND SENSOR - SONARS

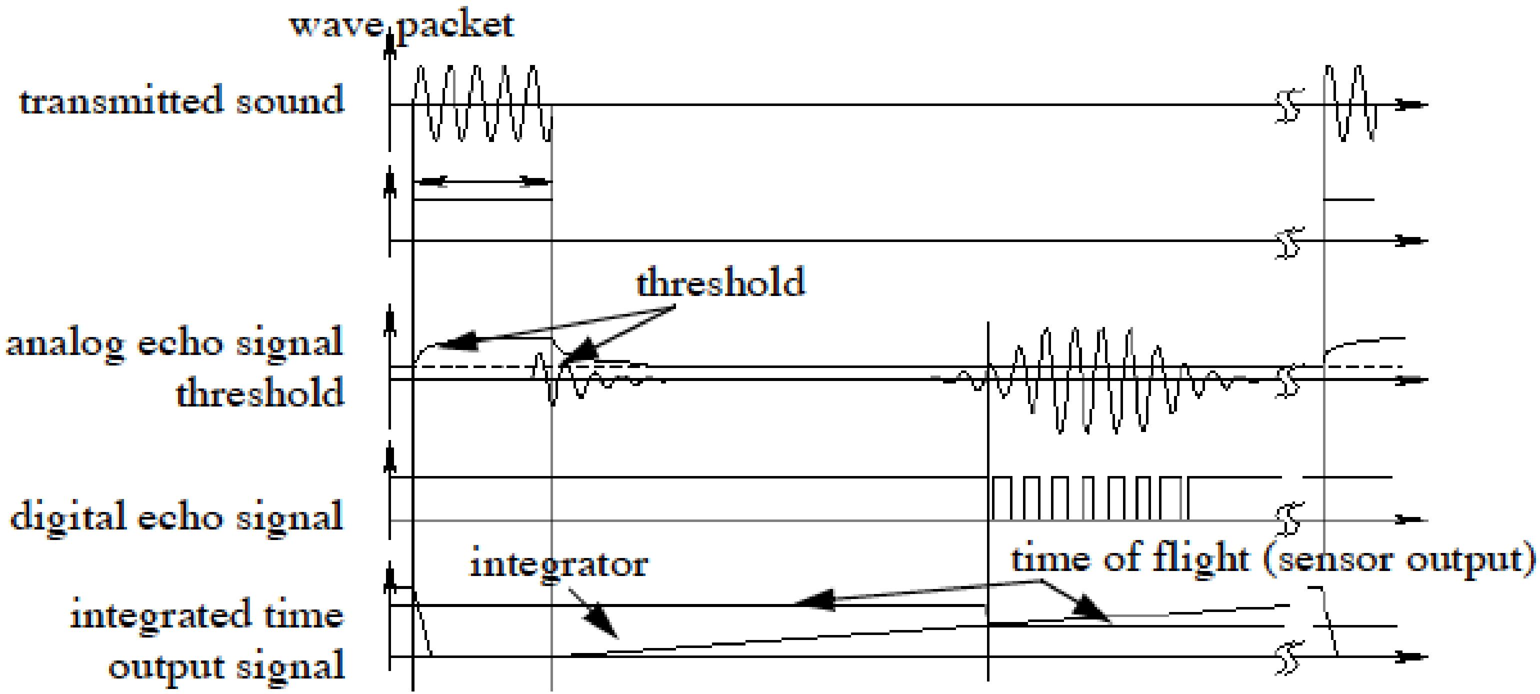
- Emits an ultrasound signal
- Waits until they receive the echo
- Time of flight
- Active





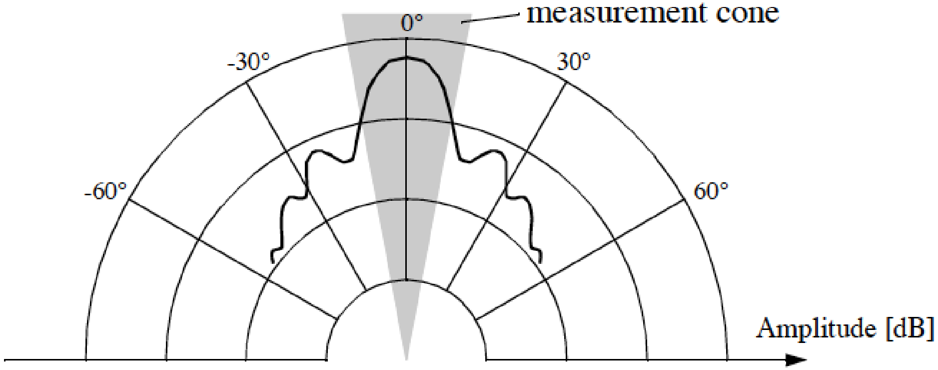
Polaroyd 6500

SONARS

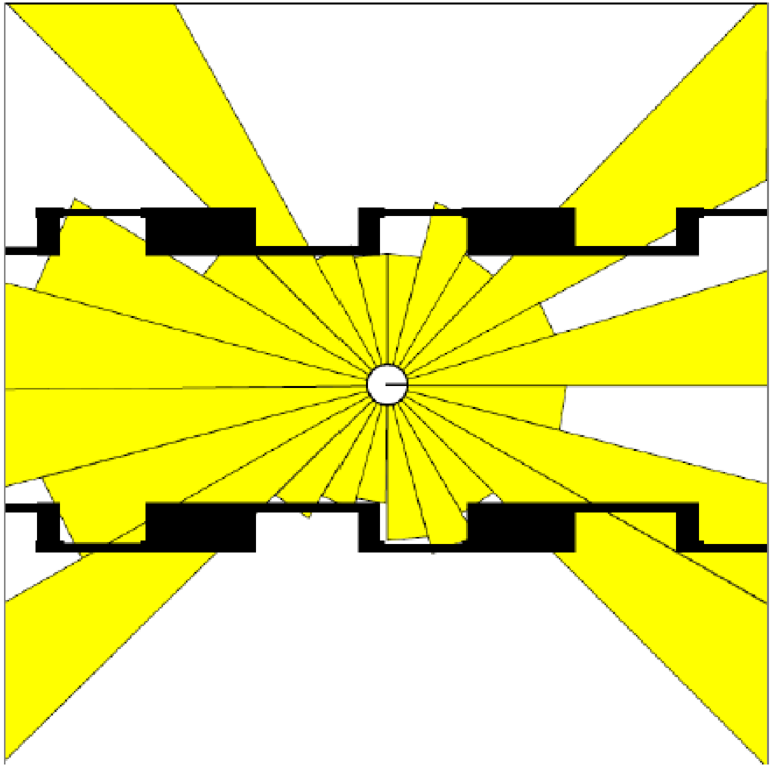


Signals of an ultrasonic sensor

SONARS



Typical intensity distribution of a ultrasonic sensor



signal profile

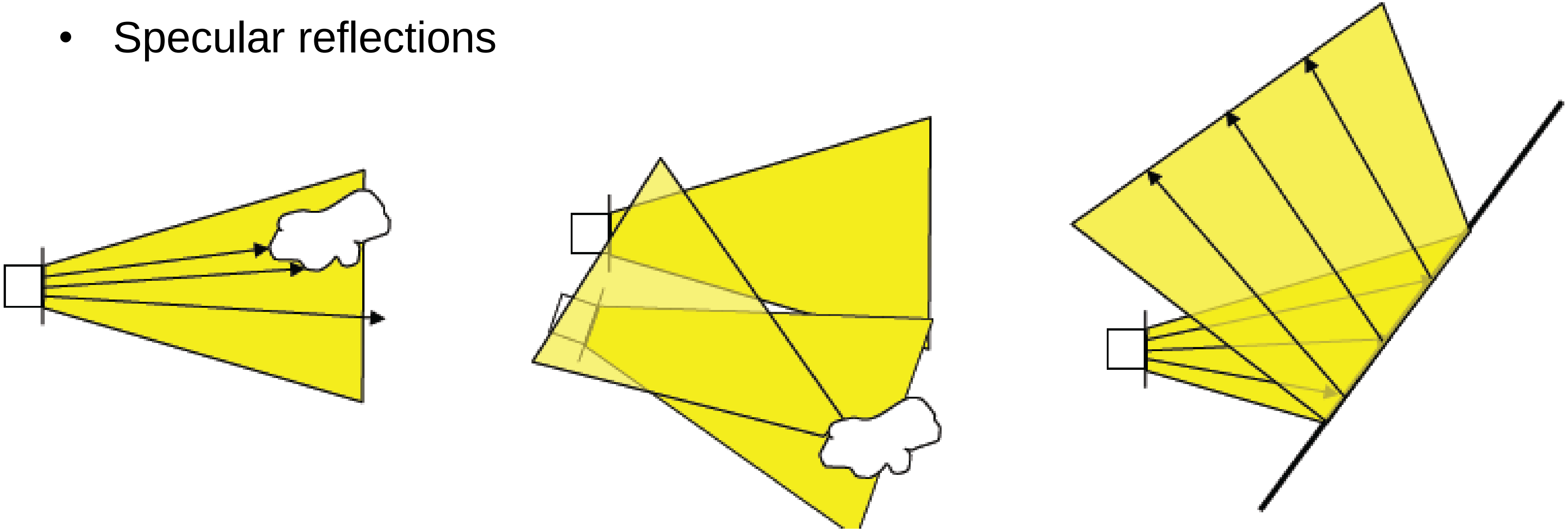
ultrasound scan

SONARS

Source of Error

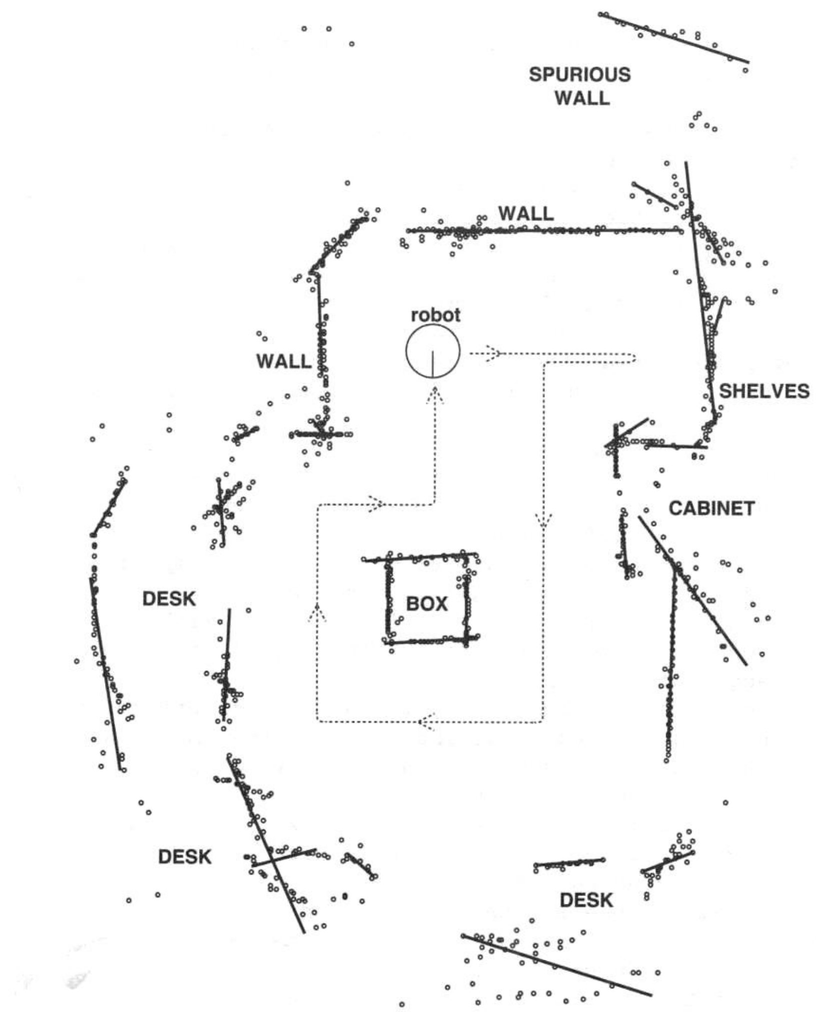
- Opening angle
- Interference
- Specular reflections

- Accurate range
- Inaccurate bearing



SONARS

- Accumulated measurements



SONARS PARALLEL OPERATION

Given a 15 degrees opening angle, 24 sensors are needed to cover the whole 360 degrees area around the robot

Let the maximum range we are interested in be 10m

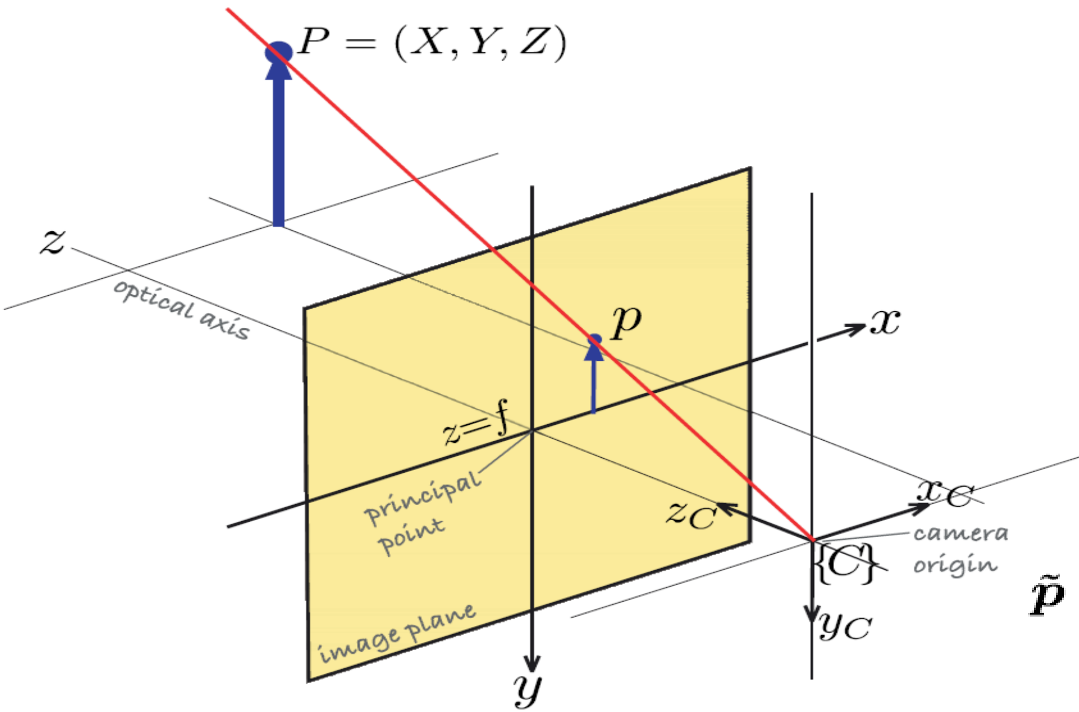
- The time of flight then is 2*10/330 s=0.06 s
- A complete scan requires 1.45 s
- To allow frequent updates (necessary for high speed) the sensors have to be fired in parallel.
- This increases the risk of interference

CAMERAS

Types of cameras

- Monocular
- Monochrome / RGB
- Stereo
- Thermal
- Depth

PINHOLE CAMERA MODEL



$$ {}^{C}\tilde{\boldsymbol{P}} = (X, Y, Z, 1)^T $$

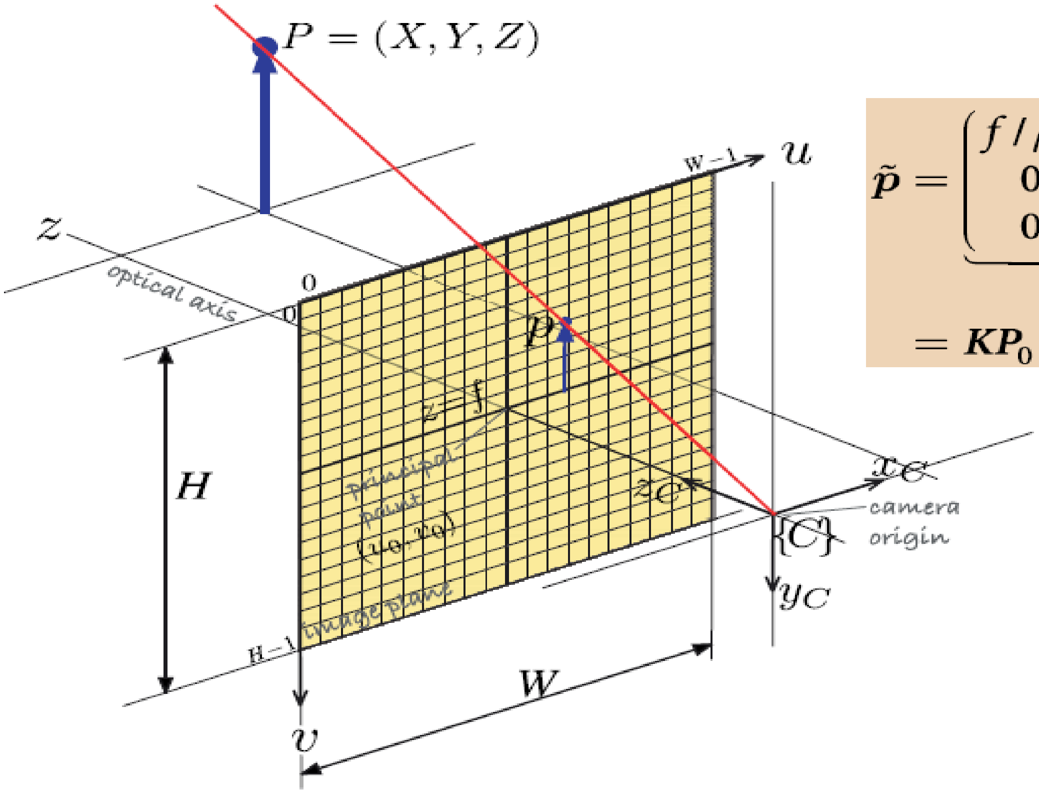
$$ \boldsymbol{x} = f\frac{X}{Z},\ \boldsymbol{y} = f\frac{Y}{Z} $$

$$ \tilde{\boldsymbol{p}} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} $$

$$ \tilde{\boldsymbol{p}} = \underbrace{\begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\boldsymbol{K}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} {}^{C}\tilde{\boldsymbol{P}} $$

$\boldsymbol{K}$: Camera Calibration Matrix

PINHOLE CAMERA MODEL



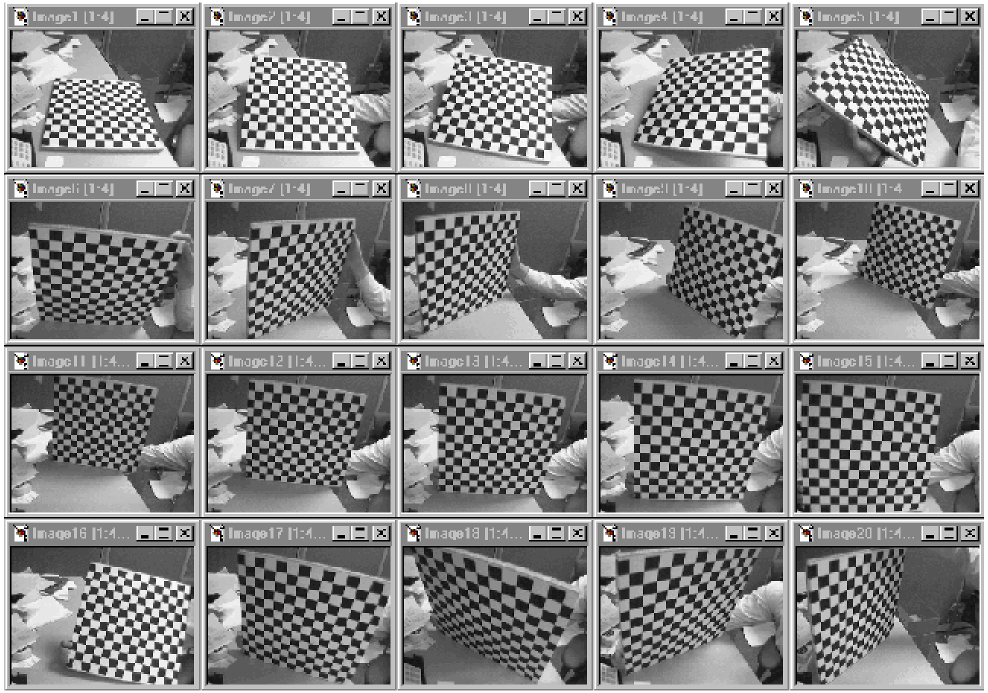
$$\tilde{\boldsymbol{p}} = \underbrace{\begin{pmatrix} f/\rho_w & 0 & u_0 \\ 0 & f/\rho_h & v_0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\text{intrinsic}} \underbrace{\left({}^{0}\mathbf{T}_C\right)^{-1}}_{\text{extrinsic}} \tilde{\boldsymbol{P}}$$

$$= \boldsymbol{K}\boldsymbol{P}_0 \, {}^{0}\boldsymbol{T}_C^{-1} \tilde{\boldsymbol{P}}$$

Camera Calibration Matrix accounting for pixels and principal point (u_0,v_0)

CAMERA CALIBRATION

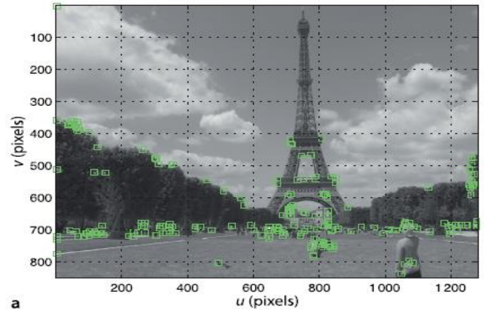
- Focal Length f
- Pixel size ρ_w, ρ_h
- Distortion coefficients k_1, k_2.
- Image center u_0, v_0



VISUAL FEATURES

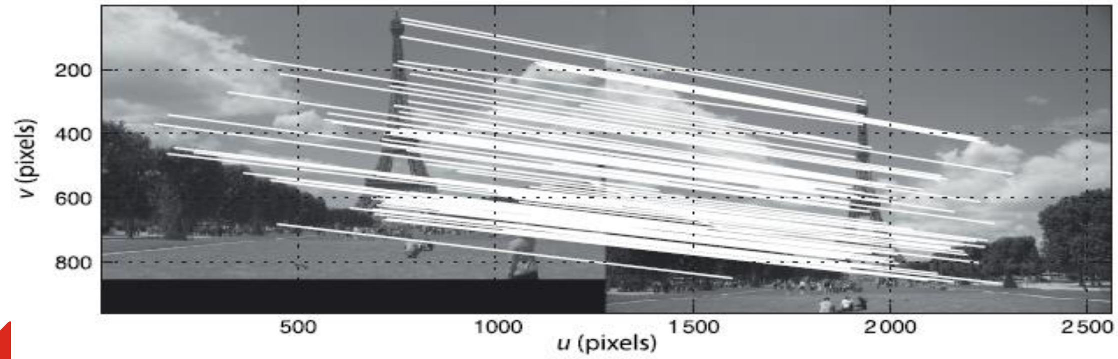
Requires image processing

Extract relevant information from the image



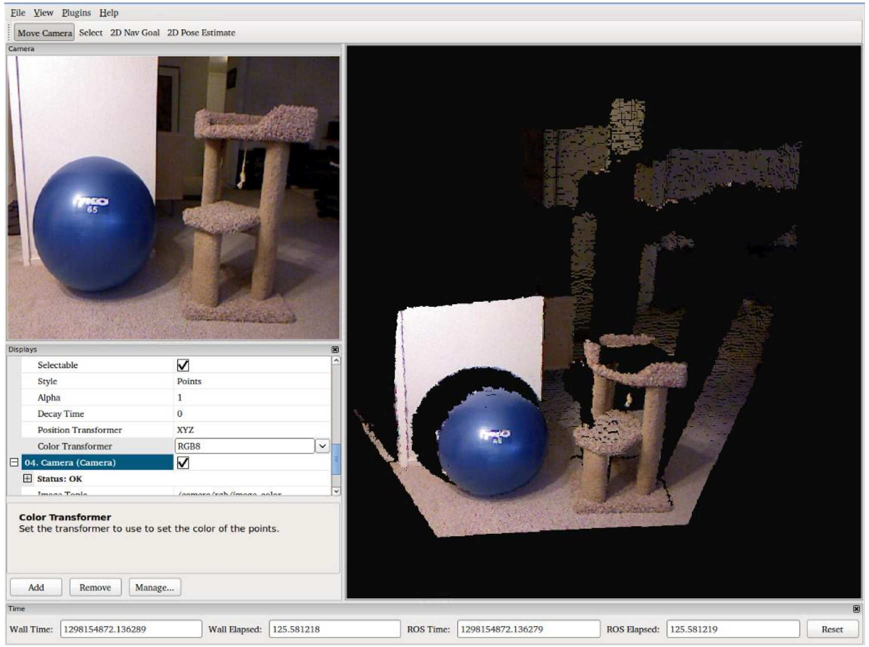
a

Track over consecutive frames

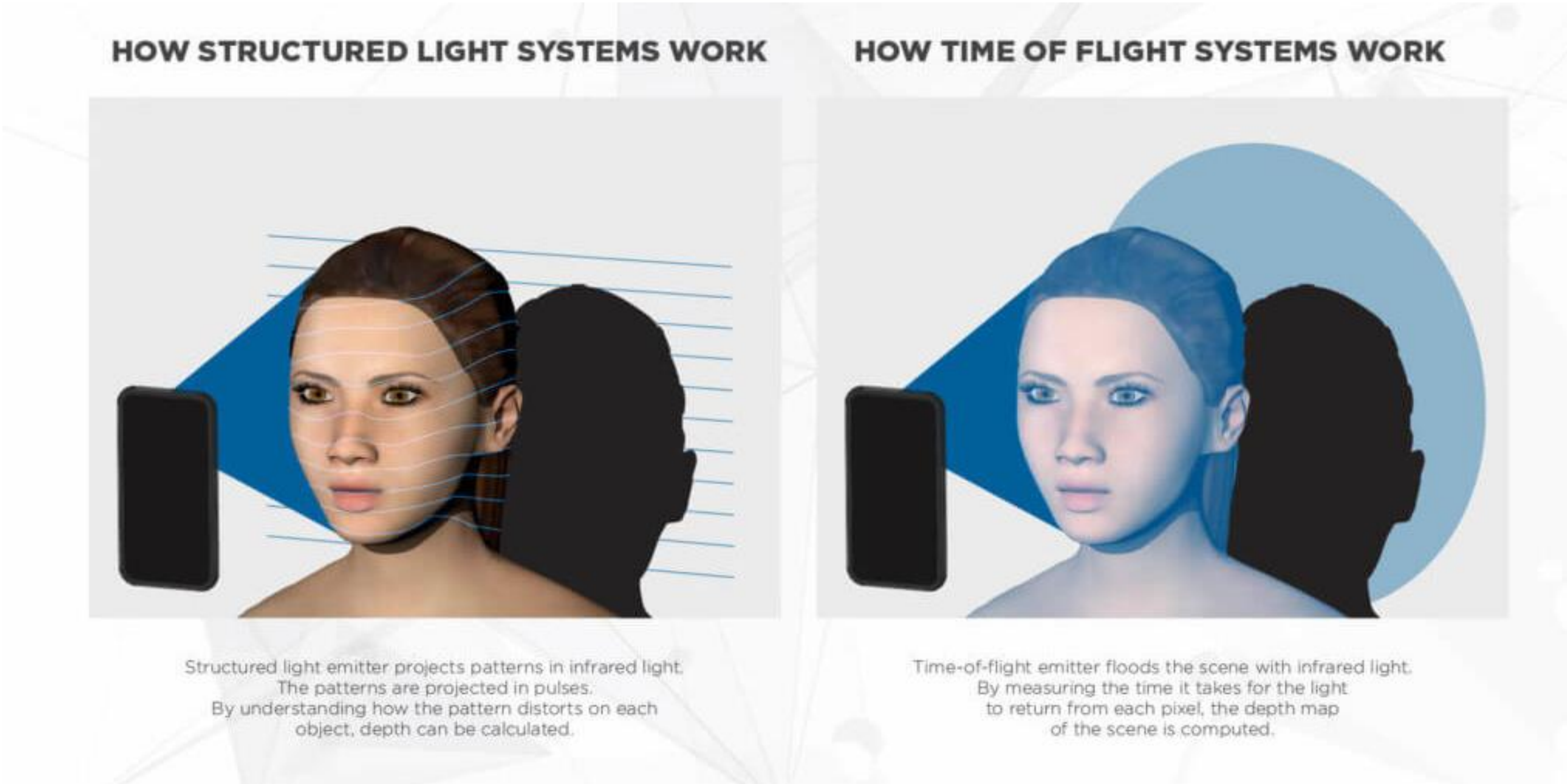


DEPTH CAMERAS

- RGD + Depth
- 3D information
- Noisy in depth (0.6 – 5m)
- Similar to Stereo/ Can be combined with Stereo
- Structure Light or Time of Flight
- Ability to combine image processing and point cloud processing



DEPTH CAMERAS



SOFTWARE TOOLS

- Point Cloud Library (PCL)
- OpenCV
- Camera Calibration Toolbox (matlab)
- ROS packages