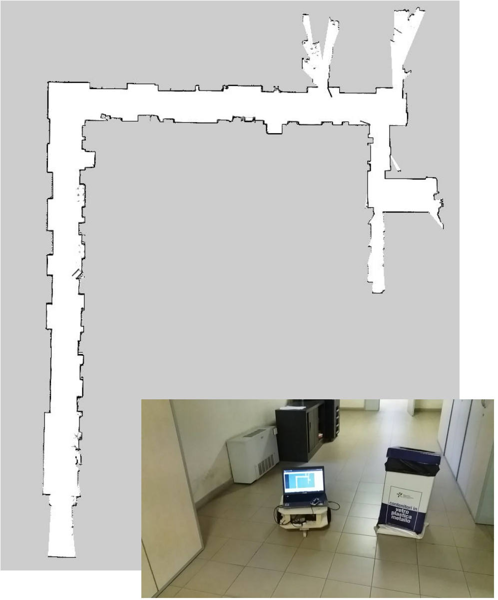
Particle filter-based localization with ROS

María T. Lázaro
mtlazaro@diag.uniroma1.it
Office: B114

Scenario

- Robot equipped with a laser moving around the DIAG basement
- Provides:
  - Odometry measurements
  - Laser scans (range and bearings)
  - Extrinsic laser calibration (Bartolo)
- Data collected as a ROS bagfile

```
mayte@zenbook:~/Documentos/probabilistic_robotics_2017_18/applications/cpp/16_thin_localizer/test_real$ rosbag info dis-underground.bag
path:        dis-underground.bag
version:     2.0
duration:    1:50s (110s)
start:       Nov 15 2017 17:35:07.06 (1510763707.06)
end:         Nov 15 2017 17:36:57.53 (1510763817.53)
size:        3.7 MB
messages:    15909
compression:  bz2 [24/24 chunks; 20.37%]
uncompressed: 17.4 MB @ 161.4 KB/s
compressed:    3.5 MB @  32.9 KB/s (20.37%)
types:        nav_msgs/Odometry      [cd5e73d190d741a2f92e81eda573aca7]
              sensor_msgs/LaserScan [90c7ef2dc6895d81024acba2ac42f369]
              tf2_msgs/TFMessage    [94810edda583a504dfda3829e70d7eec]
topics:       /odom    5484 msgs    : nav_msgs/Odometry
              /scan    4427 msgs    : sensor_msgs/LaserScan
              /tf      5998 msgs    : tf2_msgs/TFMessage
```



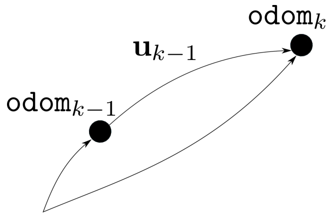
Problem Overview

- Robot state

$$\mathbf{x}_k = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix}$$

- Belief space: set of pose samples $\mathbf{x}^{[i]}$ with weights $w^{[i]}$
- Control input: given by the odometry measurements

$$\mathbf{u}_{k-1} = \begin{pmatrix} u_x \\ u_y \\ u_\theta \end{pmatrix}$$



- Measurements: laser points with respect to the laser frame

$$\mathbf{z}_j = \begin{pmatrix} z_x \\ z_y \end{pmatrix}$$

Prediction

- Transition function

$$\mathbf{x}_{k|k-1}^{[i]} = \mathbf{f}(\mathbf{x}_{k-1|k-1}^{[i]}, \mathbf{u}_{k-1} + \mathbf{n}_{k-1}^{[i]})$$
$$= \mathbf{x}_{k-1|k-1}^{[i]} \oplus (\mathbf{u}_{k-1} + \mathbf{n}_{k-1}^{[i]})$$

Composition of 2D transformations

$$\mathbf{x}_B^A = \begin{pmatrix} x_1 \\ y_1 \\ \theta_1 \end{pmatrix} \quad \mathbf{x}_C^B = \begin{pmatrix} x_2 \\ y_2 \\ \theta_2 \end{pmatrix}$$

$$\mathbf{x}_B^A \oplus \mathbf{x}_C^B = \begin{pmatrix} x_1 + x_2 \cos\theta_1 - y_2 \sin\theta_1 \\ y_1 + x_2 \sin\theta_1 + y_2 \cos\theta_1 \\ \theta_1 + \theta_2 \end{pmatrix}$$

- Control noise
  - Sampled from Gaussian distribution

$$\mathbf{n}_{k-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{k-1})$$

  - Proportional to robot motion

$$\mathbf{\Sigma}_{k-1} = \begin{pmatrix} \sigma_x^2 & & \\ & \sigma_y^2 & \\ & & \sigma_\theta^2 \end{pmatrix} \qquad \begin{pmatrix} \sigma_x^2 \\ \sigma_y^2 \\ \sigma_\theta^2 \end{pmatrix} = \mathbf{A}\mathbf{u}_{k-1}$$

Update

$$\mathbf{x}_{k|k}^{[i]} = \mathbf{x}_{k|k-1}^{[i]}$$

$$w_{k|k}^{[i]} = w_{k|k-1}^{[i]} \boxed{p(\mathbf{z}_k | \mathbf{x}_{k|k-1}^{[i]})}$$

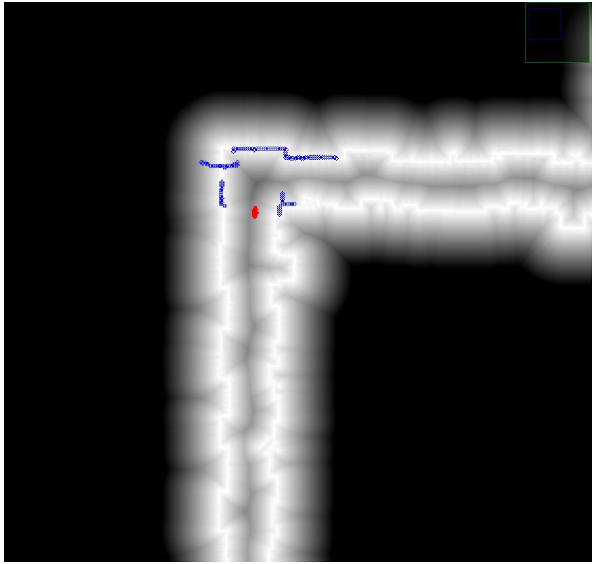
It's just a function!

```
value = likelihood(pose,measurements)
```

- For each laser point we check how close it is to our map, given the current pose $\hat{\mathbf{z}}_j^{[i]} = \mathbf{x}_{k|k-1}^{[i]} \oplus \mathbf{z}_j$
  - Small distance → High likelihood

$$p(\mathbf{z}_k | \mathbf{x}_{k|k-1}^{[i]}) \sim \mathbf{exp}(-\sum d(\hat{\mathbf{z}}_j^{[i]}))$$

Distance map



Update

- Resampling

```
function sampled_indices = uniformSample(weights,
    num_desired_samples)

dim_weights = size(weights,1);
%normalize the weights (if not normalized)
normalizer = 1./sum(weights);
%resize the indices
step = 1./num_desired_samples;

y0 = rand()*step;   %sample between 0 and 1/num_desired_sample;
yi = y0;     %value of the sample in the y space
cumulative = 0;   %this is our running cumulative distribution

for weight_index=1:dim_weights
  cumulative += normalizer*weights(weight_index); %update cumulative
  % fill with current_weight_index
  % until the cumulative does not become larger than yi
  while(cumulative > yi)
    sampled_indices(end+1,1) = weight_index;
    yi += step;
  endwhile

endfor

endfunction
```