

State estimation and Filtering

Project work 2024-2025

1 The SLAM problem

The Simultaneous Localization and Mapping (SLAM) problem is a popular challenge in mobile robotics. The SLAM problem has been intensively studied in the last 40 years. Although a number of different solutions have been devised for the wide variety of alternative settings proposed in the literature, such solutions are not yet a ready-to-use *technology*. This means that the available techniques and algorithms must be carefully tailored to the specific setup, which is defined by the sensors of the robot, its motion model, the type of map to be constructed and several other factors. For a comprehensive review of the many approaches to the SLAM problem, see [1].

In the SLAM problem, a mobile robot is placed at an unknown location in an unknown environment. The robot must incrementally build a consistent map of this environment while simultaneously determining its pose (position and orientation) within this map. In this project, we will consider the following setup:

- the environment is a 2D planar surface;
- the robot motion model is a unicycle, whose driving inputs are its forward and angular velocity;
- the robot is equipped with a Lidar (Laser imaging detection and ranging) scanning sensor, which measures distances with respect to objects or surfaces, on a set of predefined directions (angle range);
- the map consists of a set of landmarks, which are distinguished points in the environment, such as corners or poles.

Figure 1 shows a simulated experiment in which the robot travels through a corridor of the S.Niccolò building, while scanning the surrounding environment with the Lidar sensor (dashed rays).

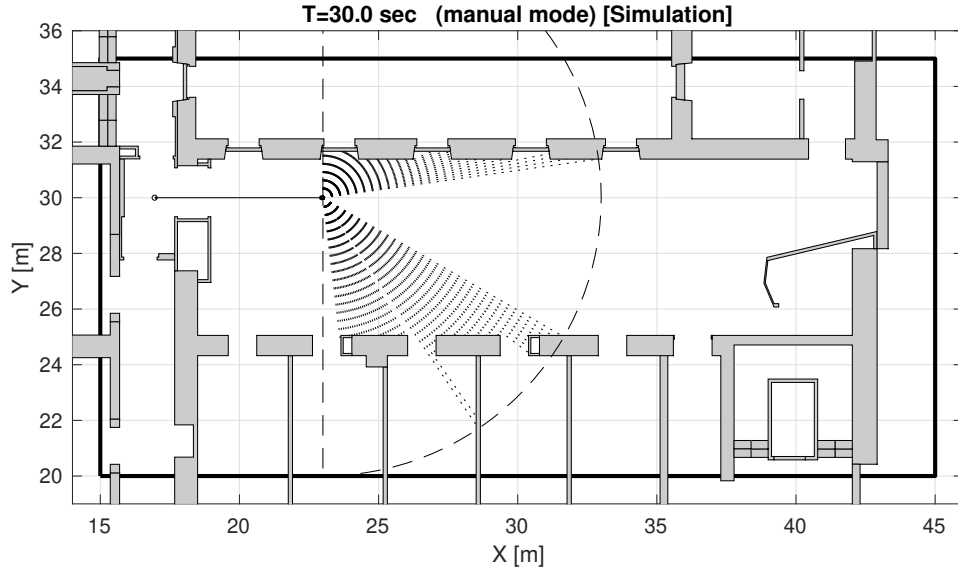


Figure 1: A mobile robot exploring a 2D environment

A single Lidar scan is reported in Figure 2. The maximum range is equal to 10 m , while the angle resolution is 1 deg , from -90° to 90° . For example, the detection of corners can be performed by extracting the local minima of the Lidar scan (with some special care: not all minima are equal...), which correspond to the red points in Figure 2. The detected landmarks are reported in red on the map, in Figure 3.

1.1 Robot motion model

The mobile robot moves in a 2D environment, described by a Cartesian (x, y) reference frame. The *robot pose* at time $t = 0, 1, \dots$ is defined by its coordinates $x(t)$, $y(t)$ (expressed in meters) and by its orientation $\theta(t)$ with respect to the x axis (expressed in radians). The unicycle motion model is described by the discrete-time equations

$$\begin{aligned} x(t+1) &= x(t) + T_s u_f(t) \cos \theta(t) \\ y(t+1) &= y(t) + T_s u_f(t) \sin \theta(t) \\ \theta(t+1) &= \theta(t) + T_s u_a(t) \end{aligned}$$

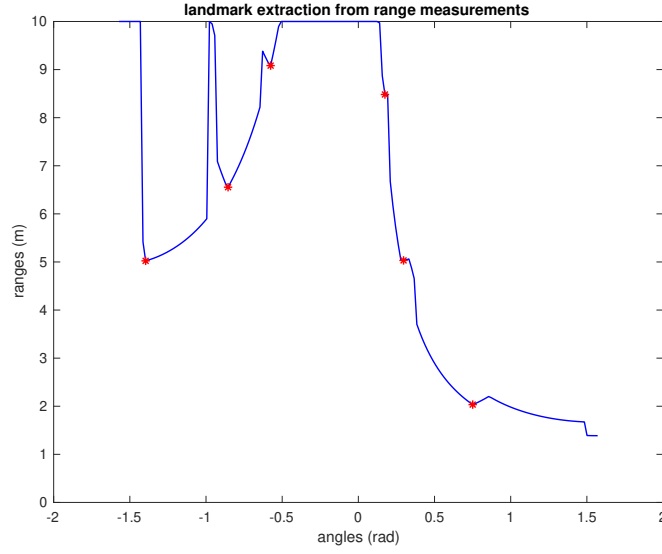


Figure 2: Range measurements with corner detection (red points).

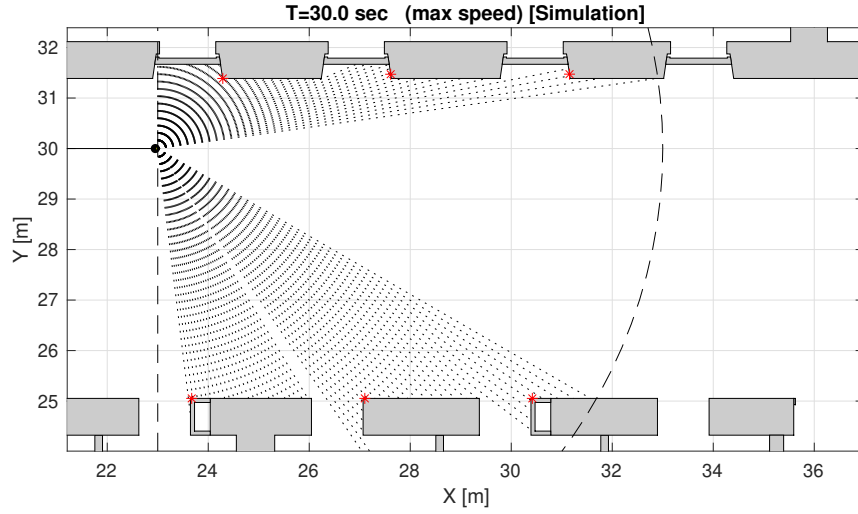


Figure 3: The detected landmarks (red points) extracted from the Lidar scan (dashed lines), assuming to know the robot pose (black dot).

where $u_f(t)$ is the forward velocity of the robot at time t , $u_a(t)$ is the angular velocity of the robot at time t , and T_s is the sampling time.

The odometry of the robot returns the measured values of the forward and angular velocity of the robot, hereafter denoted by $\hat{u}_f(t)$ and $\hat{u}_a(t)$. In practice, the odometric

measurements are corrupted by encoder errors, wheels slippage, terrain unevenness and other noise sources. In this work, it is assumed that such errors just add up to the nominal values, i.e.

$$\hat{u}_f(t) = u_f(t) + w_f(t)$$

$$\hat{u}_a(t) = u_a(t) + w_a(t)$$

where the vector $w(t) = [w_f(t) \ w_a(t)]'$ is modeled as a white process, with zero mean and (possibly time-varying) covariance matrix

$$Q(t) = \begin{pmatrix} \sigma_f^2(t) & 0 \\ 0 & \sigma_a^2(t) \end{pmatrix}.$$

1.2 Lidar sensor model

The Lidar sensor scans the environment, collecting range measurements at predefined angles in clockwise sense. The center of the angle span coincides with the robot orientation. Let S_α and r_α be the angle span and resolution, respectively. Then, the set of $N_r = \frac{S_\alpha}{r_\alpha} + 1$ angles at which measurements are taken (with respect to the positive x axis) is

$$\alpha_k(t) = \theta(t) + \frac{S_\alpha}{2} - (k-1)r_\alpha, k = 1, \dots, N_r.$$

The range distance corresponding to angle $\alpha_k(t)$ is denoted by $\rho_k(t)$. The maximum range detected by the Lidar is $\rho_{\max}$, so that $0 \leq \rho_k(t) \leq \rho_{\max}, \forall t, k$. Figure 4 shows a single range-angle pair $(\rho_k(t), \alpha_k(t))$ measured by the robot.

The Lidar measurements are corrupted by additive noise, which means that the available range measurements can be written as

$$m_{\rho_k}(t) = \rho_k(t) + v_{\rho_k}(t)$$

$$m_{\alpha_k}(t) = \alpha_k(t) + v_{\alpha_k}(t)$$

for $k = 1, \dots, N_r$, where the vector $v_k(t) = [v_{\rho_k}(t) \ v_{\alpha_k}(t)]'$ is modeled as a white process, with zero mean and covariance matrix

$$R = \begin{pmatrix} \sigma_\rho^2 & 0 \\ 0 & \sigma_\alpha^2 \end{pmatrix}.$$

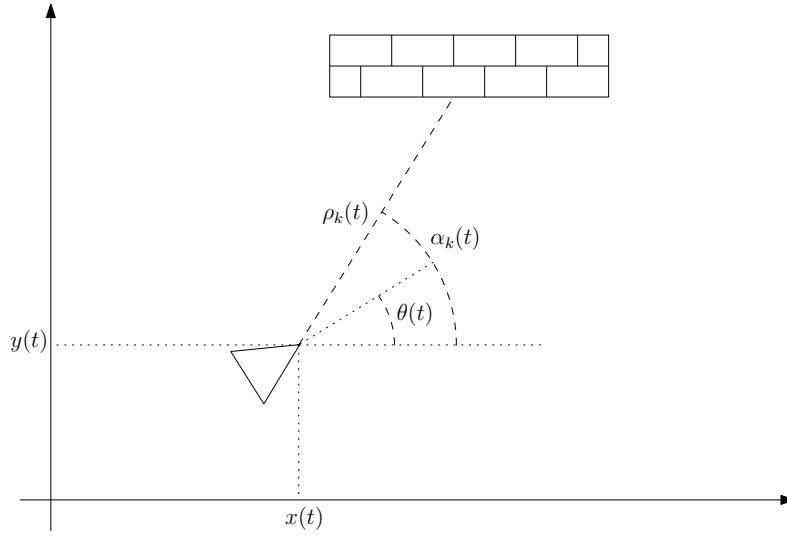


Figura 4: A single range-angle pair $(\rho_k(t), \alpha_k(t))$ measured by the robot.

2 Project organization

This project work is divided in four challenges, corresponding to formulations of the SLAM problems with increasing difficulty.

- (A) SLAM with simulated range measurements with respect to point landmarks and known data association.
- (B) SLAM with simulated range measurements with respect to point landmarks and unknown data association.
- (C) SLAM with simulated Lidar measurements on a real-world map.
- (D) SLAM with Lidar measurements taken from a real robot, in a real-world experiment.

The challenges will be assigned sequentially to each group, whenever the group has solved the previous one. At the end of the challenges, each group must deliver a unique report explaining the techniques employed and the results obtained in each challenge.

(A) SLAM with simulated range measurements with respect to point landmarks and known data association

In this first challenge, it is assumed that the environment is populated by point landmarks, whose number N_{land} is known a priori. The landmarks are *uniquely identified* when they are detected by the robot sensor. This means that whenever a landmark $L_j \in \mathbb{R}^2$, $j \in \{1, \dots, N_{land}\}$, is within distance $\rho_{\max}$ from the robot, the sensor returns the pair of measurements $m_{\rho_j}(t), m_{\alpha_j}(t)$, which is associated to the distance and angle bearing of landmark $L_j = [L_{j,x} \ L_{j,y}]'$ with respect to the current pose of the robot $z(t) = [x(t) \ y(t) \ \theta(t)]'$. The measurement model is given by

$$m_{\rho_j}(t) = \sqrt{(L_{j,x} - x(t))^2 + (L_{j,y} - y(t))^2} + v_{\rho_j}(t) \quad (1)$$

$$m_{\alpha_j}(t) = \text{atan2}\{L_{j,y} - y(t), L_{j,x} - x(t)\} - \theta(t) + v_{\alpha_j}(t) \quad (2)$$

Concerning the robot motion model, it is assumed that the uncertainty affecting the odometry is larger when the robot turns. In particular, the following model is adopted for the covariance of the process disturbance $w(t)$

$$Q(t) = \begin{cases} Q & \text{if } |\hat{u}_a(t)| \leq \omega_t \\ Q_{turn} & \text{if } |\hat{u}_a(t)| > \omega_t \end{cases}$$

where ω_t is a suitable threshold on the angular velocity and the variance σ_a^2 is typically much larger in Q_{turn} than in Q .

Objective

Design and implement an Extended Kalman Filter providing an estimate of the robot pose and landmark positions, at every time t , based on the range measurements $m_{\rho_j}(t), m_{\alpha_j}(t)$ and odometric data $\hat{u}_f(t), \hat{u}_a(t)$. Denote the state vector of the EKF as

$$\begin{aligned} z(t) &= [x(t) \ y(t) \ \theta(t) \ L_{1,x} \ L_{1,y} \ \dots \ L_{N_{land},x} \ L_{N_{land},y}]' \in \mathbb{R}^{3+2N_{land}} \\ &= [z^u(t) \mid z^l(t)]' \end{aligned}$$

where $z^u(t) \in \mathbb{R}^3$ contains the robot pose and $z^l(t) \in \mathbb{R}^{2N_{land}}$ the landmark coordinates. For a meaningful comparison with the ground truth, assume the initial condition of the

EKF equal to the true initial robot pose. Pick any reasonable initial value for the landmark positions (e.g., $L_j = [0 \ 0]'$, $\forall j$). This amounts to choose

$$\hat{z}(0|-1) = [x(0) \ y(0) \ \theta(0) \ | \ 0_{1 \times 2N_{land}}]'$$

For the initial covariance matrix, a reasonable choice is

$$P(0|-1) = \begin{bmatrix} P^u(0|-1) & 0 \\ 0 & P^l(0|-1) \end{bmatrix}$$

where $P^u(0|-1) = \text{diag}([\lambda_1 \ \lambda_2 \ \lambda_3])$, with λ_i , $i = 1, 2, 3$, sufficiently small (recall the initial pose is assumed to be known), and $P^l(0|-1) = \eta I_{2N_{land} \times 2N_{land}}$, with η sufficiently large to cover the entire environment. In order to avoid large linearization errors due to wrong landmark initialization, the first time a landmark is detected it is advisable to insert in the state vector the landmark coordinates inferred from the range measurement and the robot pose, before performing the correction step associated to the corresponding range measurement.

Data files

The files `data_point_land_1.npz` and `data_point_land_2.npz` contain data from two simulated experiments in the same environment. The only difference between the two experiments is the maximum range of the Lidar sensor, which is $\rho_{\max} = 10 \text{ m}$ in the first experiment and $\rho_{\max} = 5 \text{ m}$ in the second.

The files contain the following data (all arrays and matrices are in **numpy** format).

- The Python dictionary **Meas** contains the measurements collected by the Lidar. Specifically:
 - the array **Meas['range'] [t]** contains the range measurements $m_{\rho_j}(t)$;
 - the array **Meas['angle'] [t]** contains the angles $m_{\alpha_j}(t)$ at which range measurements are taken with respect to the robot pose;
 - the array **Meas['land'] [t]** contains the landmark indexes j corresponding to the range and angle measurements $m_{\rho_j}(t), m_{\alpha_j}(t)$, collected at time t (landmark indexes j range from 1 to N_{land}).

- The arrays **Uf** and **Ua** contain the sequences of measured robot velocities $\hat{u}_f(t)$, $\hat{u}_a(t)$ returned by the odometry of the robot.
- The scalars **Ts** and **Nland**, for the sampling time T_s and number of landmarks N_{land} .
- The matrices **Q**, **Qturn** and **R**, corresponding to the covariances Q , Q_{turn} and R , respectively. The scalar **wturn** contains the threshold ω_{turn} .
- The ground truth data, to be used only for comparison purposes. In particular:
 - the matrix **Pose**, whose columns contain the sequences $x(t)$, $y(t)$ and $\theta(t)$ of the robot pose;
 - the matrix **Landmarks**, in which each row contains the $x - y$ coordinates of a landmark (since in Python row numbers start from 0, the i -th row corresponds to the identification number $i + 1$ of the landmark, returned in `Meas['land']`).

Report

Compare the estimated robot trajectory and orientation with the ground truth (true robot pose) and with the corresponding estimates obtained by integrating the (noisy) robot odometry.

Verify the consistency of the estimates by checking if the pose estimation errors lie inside the corresponding 3σ confidence intervals. For the landmarks, check if at the final time instant the true landmark position belongs to the corresponding confidence ellipses, defined as

$$\mathcal{E}_j = \{L_j \in \mathbb{R}^2 : (L_j - \hat{L}_j)' P_j^{-1} (L_j - \hat{L}_j) < r_\chi\}$$

where $\hat{L}_j$ is the estimate of the j -th landmark location at the end of the simulation, $P_j \in \mathbb{R}^{2 \times 2}$ is the corresponding covariance matrix returned by the EKF, and r_χ is the value of the inverse χ^2 cumulative distribution function corresponding to the desired confidence level (e.g., for a 99% probability level, $r_\chi = 9.21$).

Compare the results obtained from the two data sets. In the second experiment, what happens when the robot returns to its initial position? How can it be explained?

Riferimenti bibliografici

- [1] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.