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).

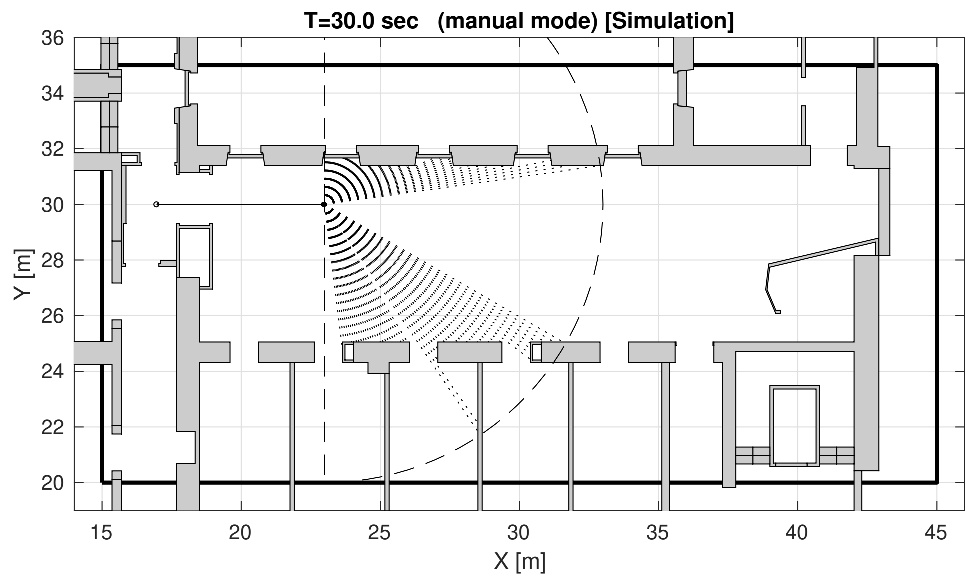

Figura 1: A mobile robot exploring a 2D environment

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^o to 90^o. 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 $t = 0, 1, \ldots$ 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

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

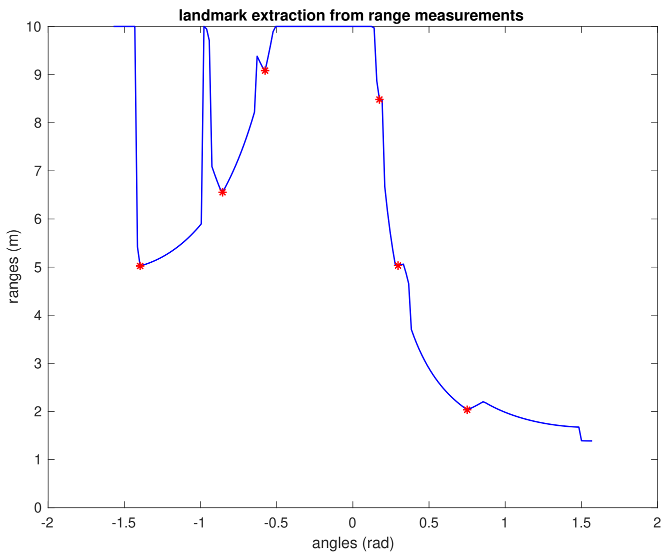

Figura 2: Range measurements with corner detection (red points).

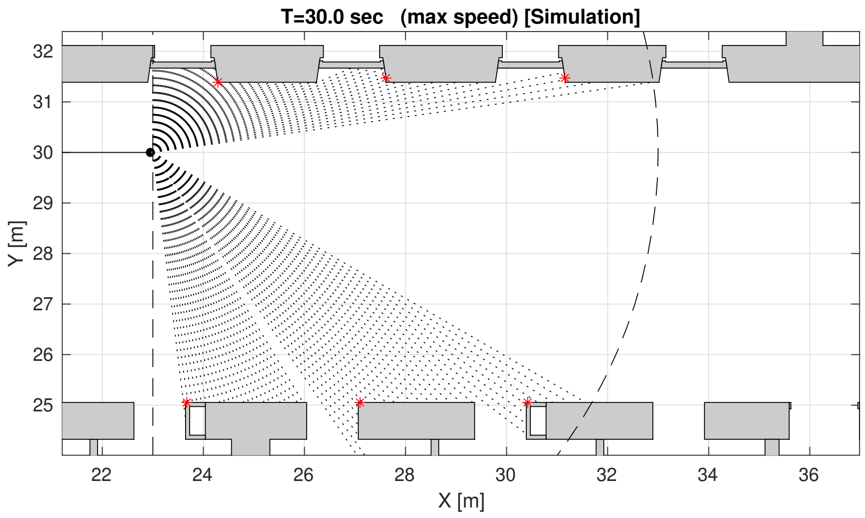

Figura 3: The detected landmarks (red points) extracted from the Lidar scan (dashed lines), assuming to know the robot pose (black dot).

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

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

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 $k = 1, \ldots, N_r$, 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

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

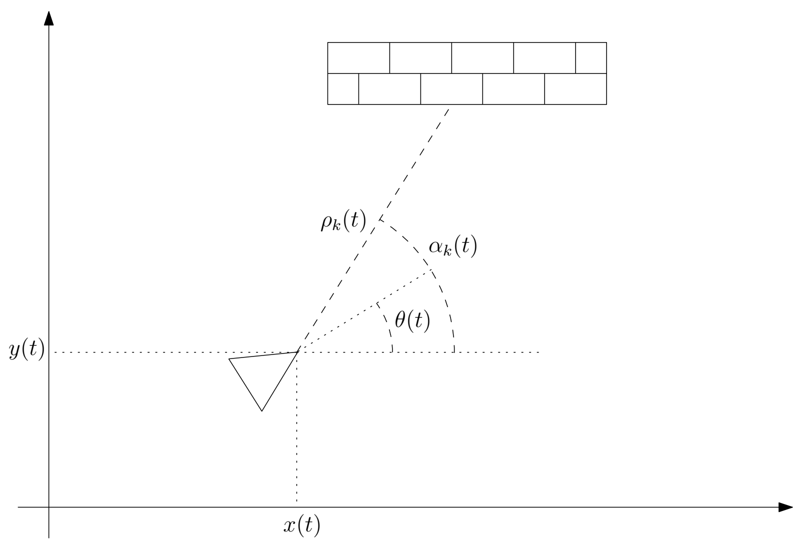

Figura 4: A single range-angle pair $(\rho_k(t), \alpha_k(t))$ measured by the robot.

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

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

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

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 $w(t)$

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

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

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

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

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

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 $\rho_{\max} = 10\,m$ in the first experiment and $\rho_{\max} = 5\,m$ 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 ladmarks 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?

(B) SLAM with simulated range measurements with respect to point landmarks and unknown data association

A key problem in robot navigation based on landmarks is to uniquely identify a perceived landmark. Artificial landmarks can be designed in order to make this identification easy (like QR codes, AprilTags, etc.). However, in natural environment commonly employed landmarks, like corners or furnitures, are often indistinguishable. Therefore, it is important to: (i) establish if a detected landmark is a new element to be added to the map or it is among those already present in the map; (ii) in the latter case, correctly associate the detected landmark to the existing one. Making errors in this case can dramatically deteriorate the performance of the SLAM algorithm. If associations are not detected, the map will be populated of many spurious landmarks and the robot will eventually be unable to correct the odometry errors using the environmental measurements. On the other hand, wrong associations will generate incorrect map estimates, with respect to which it will be impossible to localize the robot. Hence, data association is a crucial task that must be performed carefully.

Data association with Mahalanobis distance

A common method for performing data association is based on maximum likelihood. The idea is to compute, for each pair of range-angle measurements at time t, the likelihood that such measurements are associated to one of the previously detected landmarks. If the resulting maximum likelihood is above a certain threshold, it will be associated to the corresponding landmark. Conversely, if all likelihoods are below another threshold, the measurement is used to insert a new landmark in the map. Notice that when the maximum likelihood lies between the two thresholds, the measurement is simply discarded. In fact, it is better to avoid processing a measurement when its data association has a significant probability to be incorrect.

Consider a vector $x \in \mathbb{R}^n$ and a Gaussian distribution in $\mathbb{R}^n$ with mean μ and covariance matrix P. Then, by observing that $P^{-1/2}(x - \mu)$ is a standard Gaussian vector (with zero mean and unitary covariance), one has that $d = (x - \mu)' P^{-1}(x - \mu)$ is a χ^2 random variable with n degrees of freedom. The variable d is also called the *Mahalanobis distance* of vector x from the distribution $\mathcal{N}(\mu, P)$.

By using the above idea, we can compute the Mahalanobis distance of the measurements $m_{\rho_j}(t), m_{\alpha_j}(t)$ from its predicted distribution at time t provided by the EKF, assuming the measurement is associated to landmark L_k. This amounts to

$$d_{j,k}(t) = \delta_{j,k}(t)'[H_k(t)P(t|t-1)H'_k(t) + R]^{-1}\delta_{j,k}(t)$$

where

$$\delta_{j,k}(t) = \begin{bmatrix} m_{\rho_j}(t) - \sqrt{(\hat{L}_{k,x}(t|t-1) - \hat{x}(t|t-1))^2 + (\hat{L}_{k,y}(t|t-1) - \hat{y}(t|t-1))^2} \\ m_{\alpha_j}(t) - \operatorname{atan2}\{\hat{L}_{k,y}(t|t-1) - \hat{y}(t|t-1), \hat{L}_{k,x}(t|t-1) - \hat{x}(t|t-1)\} + \hat{\theta}(t|t-1) \end{bmatrix},$$

$P(t|t-1)$ is the predicted covariance matrix of the EKF, and $H_k(t)$ is the Jacobian matrix of the nonlinear measurement model (1)-(2), computed in the predicted state values $\hat{x}(t|t-1)$, $\hat{y}(t|t-1)$, $\hat{\theta}(t|t-1)$, $\hat{L}_{k,x}(t|t-1)$, $\hat{L}_{k,y}(t|t-1)$.

Now, let

$$d_j^* = \min_k d_{j,k}$$

where the minimum is taken with respect to all landmark indexes k that are present in the map at time $t-1$. Then, by choosing two suitable thresholds $\tau_1 < \tau_2$, corresponding to the probability levels for association to an existing landmark and generation of a new landmark, respectively, one has that

- if $d_j^* < \tau_1$, the measurement $m_{\rho_j}(t), m_{\alpha_j}(t)$ is associated to landmark $L_{k_j^*}$, where

$$k_j^* = \arg\min_k d_{j,k}$$

- if $d_j^* > \tau_2$, the measurement $m_{\rho_j}(t), m_{\alpha_j}(t)$ is associated to a new landmark to be inserted in the map. The predicted state vector is augmented by inserting the new landmark position

$$\hat{L}_x = \hat{x}(t|t-1) + m_{\rho_j}(t)\cos\left(m_{\alpha_j}(t) + \hat{\theta}(t|t-1)\right)$$
$$\hat{L}_y = \hat{y}(t|t-1) + m_{\rho_j}(t)\sin\left(m_{\alpha_j}(t) + \hat{\theta}(t|t-1)\right)$$

while the covariance matrix $P(t|t-1)$ is augmented by inserting the covariance matrix of the new landmark (usually chosen as $\lambda I_{2\times 2}$, with λ a sufficiently large value).

Finally, in both cases the measurement $m_{\rho_j}(t), m_{\alpha_j}(t)$ is processed in the correction step of the EKF, with the established landmark association. The procedure is repeated for every measurement pair $m_{\rho_j}(t), m_{\alpha_j}(t)$, $j = 1, \dots, \ell_t$, where ℓ_t is the number of landmarks detected by the Lidar at time t.

As an example, the values of the thresholds can be chosen as $\tau_1 = 5.9915$ and $\tau_2 = 13.8155$, corresponding respectively to the 95% and the 99.9% confidence levels of the χ^2 distribution with 2 degrees of freedom. This means that when a new landmark is inserted in the map, there is a probability of about 0.1% that the considered measurement refers to an existing landmark. Different thresholds can be chosen by using the inverse χ^2 distribution (in Python, use function `chi2.ppf` from `scipy.stats.distributions`).

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 $\hat{u}_f(t), \hat{u}_a(t)$. Insert the landmarks progressively into the state vector, when they are detected for the first time. For comparison with the ground truth, assume the initial condition of the EKF equal to the true initial robot pose.

Data files

Use the same data files as in challenge (A).

Report

Report the same comparisons as in challenge (A). Annotate if there are significant differences between the results obtained in challenges (A) and (B).

(C) SLAM with simulated Lidar measurements on a real-world map

In the third challenge of this project work, measurements are provided by a Lidar sensor. The Lidar model is the one described in Section 1.2. At each time t, a set of range-angle pairs $[m_{\rho_k}(t)\ m_{\alpha_k}(t)]'$, with $k = 1, \dots, N_r$, is available. In order to apply the EKF-based SLAM algorithm developed for challenge (B), it is necessary to extract the range-angle measurements that correspond to a potential landmark in the map. This, in turn, requires to define the types of landmarks that it is useful to place in the map. In particular, one should prefer landmarks that are clearly identifiable, so that they can be easily recognized and distinguished from the other landmarks, every time they are detected.

Landmark extraction

In this project, we aim at extracting range measurements corresponding to sharp corners in the map. By looking at Figures 2 and 3, it is apparent that one way to detect corners is to look for local minima in the range measurements plot. This can be done, for example, by using the Python function `scipy.signal.find_peaks`. The relevance of a local minimum can be evaluated in terms of the "prominence" parameter. The command

```
scipy.signal.find_peaks(-ranges, prominence=prom)
```

finds all the local minima in the function taking the values `ranges`, whose prominence is larger than `prom`. The minus sign is due to the fact that `scipy.signal.find_peaks` returns local maxima. The prominence of a minimum measures how much the minimum stands out from the surrounding signal and is defined as the vertical distance between the minimum and its highest contour line. Figure 5 shows an example of prominence for a local minimum.

Care must be taken to consider only local minima corresponding to corners in the map. In fact, there may be local minima in the range plot which come from the scan of a straight wall (in this case, the local minimum does not correspond to a corner but to to the point

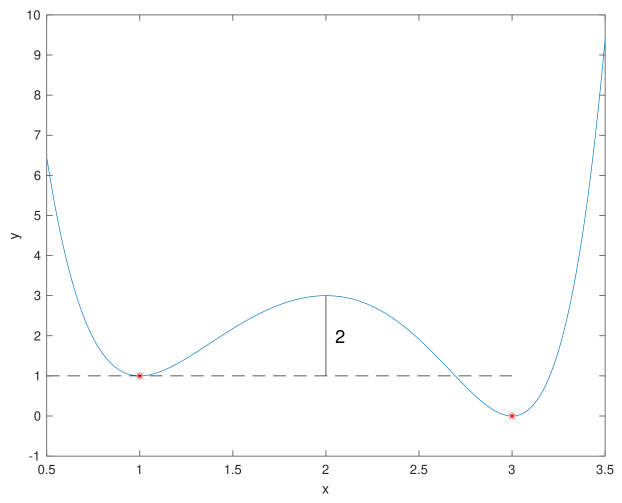

Figura 5: A function with two minima. The local minimum in $x = 1$ has prominence 2.

in the wall which is closest to the robot). One aim of this challenge is to find a meaningful way to discard such spurious local minima.

Objective

First, write a function taking as inputs the range measurements of a single Lidar scan and producing as outputs the measurements corresponding to a landmark (a corner in the map). Suitably tune the prominence parameter to avoid the detection of spurious landmarks. Find a meaningful way to skip local minima not corresponding to corners.

Then, exploiting the code developed for challenge (B), design and implement a EKF providing an estimate of the robot pose and landmark positions, at every time t, based on the Lidar measurements and odometric data. Initialize the EKF as in the previous challenges (in particular, set the initial robot pose equal to the true one, for the sake of comparison with the ground truth). If necessary, tune the data association thresholds τ_1, τ_2 in order to promote a correct landmark matching.

Data files

The files `data_sim_lidar_1.npz` and `data_sim_lidar_2.npz` contain data from two simulated experiments within a portion of the San Niccolò building. The two experiments differ for the level of noise in the Lidar measurements.

The files contain the following data.

- The matrix `noisyRangeData` contains the range measurements collected by the Lidar. The entry `noisyRangeData(j,t)` contains the range measurement $m_{\rho_j}(t)$.

Ranges exceeding the maximum distance $\rho_{\max}$ detectable by the Lidar are set to -1.

- The vector `angles` contains the angles spanned by the Lidar. The entry `angles(j)` contains the angle $m_{\alpha_j}(t)$ (it is assumed that angles are the same at every time t).

- The vectors `Uf`, `Ua`, the matrices `Q`, `Qturn` and `R`, the scalars `Ts`, `wturn`, and the matrix `Pose`, contain the same quantities as in the data files for challenges (A) and (B).

- In order to compare the estimated map with a ground truth description of the environment, the array `Obstacles` contains a description of the San Niccolò map which can be plotted by the command `PlotMapSN(Obstacles)` (function `PlotMapSN` is provided along with the data files). Moreover, matrix `trueMap` contains an ideal representation of the obstacles in the map extracted from a noiseless version of the Lidar measurements and robot trajectory. It can be plotted by using the command `ply.plot(trueMap[:,0],trueMap[:,1],'.r')`.

Report

Report the same comparisons as in challenges (A) and (B). Moreover, compare the ground truth maps of the environment described above with those obtained by plotting all the noisy Lidar scans from: (a) the estimated robot pose; (b) the robot pose obtained by integrating the robot odometry.

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.