Monte Carlo Localization using Particle Filter

Lab Exercise 03

The goal

The goal of this exercise is to implement a Monte-Carlo Localization (MCL) algorithm to localize an one-dimensional Mobile robot being moved, with constant velocity, in a hallway. You will program all the code in Matlab.

The MCL Problem

The MCL is an implementation of the Markovian Localization problem where the involved pdfs (probability density functions) are represented through samples (particles) and the Bayes Filter is implemented through the Particle Filter. Markov Localization addresses the problem of state estimation from sensor data. MCL is a probabilistic algorithm and instead of maintaining a single hypothesis as to where in the world a robot might be, MCL maintains a probability distribution over the space of all such hypotheses. The probabilistic representation allows it to weight these different hypotheses in a mathematically sound way.

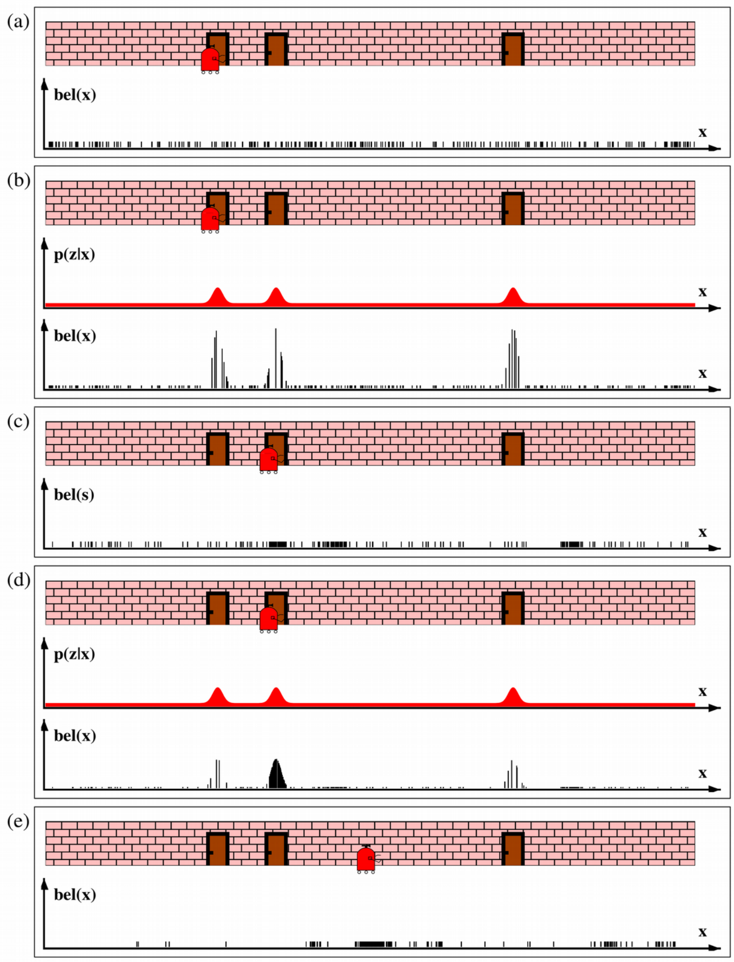

Figure 1: MCL of a Mobile Robot Moving in a Hallway.

In Figure 1, the initial global uncertainty is achieved through a set of pose particles drawn at random and uniformly over the entire pose space, as show in Fig-1a. As the robot senses the door, line 5 (see Figure 2) of in the algorithm MCL assigns importance factors to each particle. The resulting particle set is shown in Figure 1b. The height of each particle in this figure shows its importance weight. It is important to notice that this set of particles is identical to the one in Figure

1a—the only thing modified by the measurement update are the importance weights.

Then Figure 1c shows the particle set after re-sampling (line 8-11 in the algorithm MCL) and after incorporating the robot motion (line 4). This leads to a new particle set with uniform importance weights, but with an increased number of particles near the three likely places. The new measurement assigns non-uniform importance weights to the particle set, as shown in Figure 1d. At this point, most of the cumulative probability mass is centered on the second door, which is also the most likely location. Further motion leads to another re-sampling step, and a step in which a new particle set is generated according to the motion model (Figure 1e). The particle sets approximate the correct posterior, as would be calculated by an exact Bayes filter.

The Algorithm

Figure 2 shows a general Particle Filter and Figure 3 the pseudocode of the MCL Algorithm as an instance of the Particle Filter.

1: **Algorithm Particle_filter**($\mathcal{X}_{t-1}, u_t, z_t$):
2: $\quad \bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
3: $\quad$ for $m = 1$ to M do
4: $\quad\quad$ sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$
5: $\quad\quad w_t^{[m]} = p(z_t \mid x_t^{[m]})$
6: $\quad\quad \bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7: $\quad$ endfor
8: $\quad$ for $m = 1$ to M do
9: $\quad\quad$ draw i with probability $\propto w_t^{[i]}$
10: $\quad\quad$ add $x_t^{[i]}$ to $\mathcal{X}_t$
11: $\quad$ endfor
12: $\quad$ return $\mathcal{X}_t$

Figure 2: Particle filter algorithms.

1: **Algorithm MCL**($\mathcal{X}_{t-1}, u_t, z_t, m$):
2: $\quad \bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
3: $\quad$ for $m = 1$ to M do
4: $\quad\quad x_t^{[m]} = \textbf{sample\_motion\_model}(u_t, x_{t-1}^{[m]})$
5: $\quad\quad w_t^{[m]} = \textbf{measurement\_model}(z_t, x_t^{[m]}, m)$
6: $\quad\quad \bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
7: $\quad$ endfor
8: $\quad$ for $m = 1$ to M do
9: $\quad\quad$ draw i with probability $\propto w_t^{[i]}$
10: $\quad\quad$ add $x_t^{[i]}$ to $\mathcal{X}_t$
11: $\quad$ endfor
12: $\quad$ return $\mathcal{X}_t$

Figure 3: MCL Algorithm.

Work to do

You must complete the m-file containing the MCL algorithm. These are the steps you must follow:

1. **Read the m-file** and try to understand how it works. Even though it is incomplete, it is possible to run it. It is necessary to have the `animation.mat` file provided for the animation.
2. Read the help of the **random** and **pdf** MATLAB functions. They will be useful for this lab.
3. Program the **sample_motion_model** (prediction). Tip: since it is a simulation it is perfect motion, include the noise in your model as well.
4. Program the **measurement_model** (update).
5. Program the **resampling** step (either *Resampling Wheel* or *Low Variance Sampling*).
6. After finishing the lab exercise, upload **ONLY ONE** file (no .zip or multiple versions), and please name the file as: `MCL_Localization_lab.m`.

Note 1: Report is NOT required! If the code is running it should be enough. Only write a report if you believe your solution is not working properly, explaining the details. In case you decide to write a report, it shouldn't be more than **4 pages.**

Note 2: Please do not modify the code before the comment

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%   COMPLETE THESE FUNCTIONS   %%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

You only need to concentrate in these three functions: `sample_motion_model()`, `measurement_model()`, and `resampling()`.

Note 3: you ***cannot*** use the built-in Matlab function `randsample(),datasample()` or similar.

Learning material

The lab can be solved with few lines of code, however understanding is key. Since no report is required, most of the time will be expending on learning. In order to complement the lecture (which doesn't actually include MCL), it is highly recommended to see some of the great videos from Sebastian Thrun in the Udacity curse: "Artificial Intelligence for Robotics".

Videos

Link of the Udacity course can be found next. In particular for the Lesson 1, it is also recommended to complete the Quiz questions (which are in Python), it will give you some insight in how to complete the lab, the code is quite similar.

- Lesson 1 [Localization]
- Lesson 8 [Particle Filters] (and some of them Quiz questions)

Note: Lesson 4 (Kalman Filters) is **not required** for this exercise and can be skipped.

Book

This lab exercise is based on the **Probabilistic Robotics** book [ETH library]. Available digital version [pdf]. **If you want to complement the learning even further, the recommended chapter are**: Chapter 1., Chapter 2. (Bayes filter included), Chapter 8. (until 8.3.3).

Note: you could also see Chapter 4.2 in the book for **re-sampling techniques**, like *Low Variance Resampling* (Table 4.4 in the book).