Map Learning And Localization Using Rao-Blackwellized Particle Filters

Abhisek Panda, Prof. Indranil Saha
Indian Institute of Technology, Kanpur

Introduction

SLAM or Simultaneous Localization And Mapping is the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it. This project mainly focuses on implementing a SLAM algorithm for a single bot in such a way that it can be extended to multiple bots(swarm) navigating the same area with only a few minor changes in the algorithm. It is assumed that suitable path planning techniques are available and hence the algorithm does not discuss about path planning.

Challenges

- Sensor and Odometry Noise
- Divergence of filters due to wrong data associations
- Limited computational power

Mathematical Modelling

A probability distribution can be used to represent the belief of the robot. Belief is defined as the probability of the robot being at any point x_k at time instant $t = k$ given all the odometry readings $u_{1:k}$ and sensor measurements $z_{1:k}$.

$$bel(x_k) = p(x_k|\, u_{1:k}, z_{1:k}) \tag{1}$$

So the localization can be redefined as the problem to maintain the belief for each possible x_i. The main goal of our algorithm is to make the belief distribution as close as possible to the real distribution.

Motion Model: $p(x_k \mid x_{k-1}, u_t)$

Sensor Model: $p(z_t|m_{t-1}, x_t)$

Mapping With Known Poses

Assumptions

- All the cells are either completely filled or completely empty.
- The world is static.
- Any 2 cells in the grid are independent of each other.

The Mapping with Known Poses algorithm estimates the following distribution:

$$l(m_i|x_{1:t}, z_{1:t}) = l(m_i|z_t, x_t) + l(m_i|z_{1:t-1}, x_{1:t-1}) - l(m_i) \tag{2}$$

where l denotes the log odds notation:

$$l(x) = \log \frac{p(x)}{1 - p(x)}$$

Rao-Blackwellized Particle Filters

The Rao-Blackwellized particle filter uses the following factorization:

$$p(x_{1:t}, m \mid z_{1:t}, u_{1:t}) = p(m_i \mid x_{1:t}, z_{1:t})\; p(x_{1:t} \mid z_{1:t}, u_{1:t}) \tag{3}$$

The term $p(m_i \mid x_{1:t}, z_{1:t})$ can be computed efficiently by the mapping algorithm. To determine $p(x_{1:t} \mid z_{1:t}, u1:t)$, we apply a particle filter where each particle represents a potential representation of the robot trajectory. Each particle also carries its own version of the map. These maps are created on basis of sensor measurements and the trajectory of corresponding particle

Sub-Routines involved in the Particle Filter Algorithm

Sampling: Obtain a new generation of particles $\{x_t^i\}$ from the previous sample $\{x_{t-1}^i\}$. These samples are drawn from a proposal distribution π.
Here, we use:

$$\pi \sim p(x_t|m_{t-1}^i, x_{t-1}, z_t, u_t)$$

That is, along with the odometry measurements, the latest sensor measurement is also incorporated to the proposal. This has an obvious advantage over simply using the motion model, $\pi \sim p(x_t \mid x_{t-1}, u_t)$ as the proposal distribution, since the sensors are often more accurate than the cumulative odometry readings.

Importance Weighing: Each particle is assigned an importance weight according to the Importance Sampling Principle.

$$w_t^{(i)} = \frac{target}{proposal} \tag{4}$$

$$= \frac{p(x_{1:t}^{(i)} \mid z_{1:t}, u_{1:t})}{\pi(x_{1:t}^{(i)} \mid z_{1:t}, u_{1:t})} \tag{5}$$

$$\propto \frac{p(z_t \mid m_{t-1}^{(i)}, x_t^{(i)})\; p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_t)}{\pi(x_t \mid x_{1:t-1}^{(i)}, z_{1:t}, u_{1:t})}\, w_{t-1}^{(i)} \tag{6}$$

On replacing π with the new proposal model, we get:

$$w_t^{(i)} = w_{t-1}^{(i)} \frac{\eta\; p(z_t \mid m_{t-1}^{(i)}, x_t^{(i)})\; p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_t)}{p(x_t \mid m_{t-1}^{(i)}, x_{t-1}^{(i)}, z_t, u_t)} \tag{7}$$

$$\propto w_{t-1}^{(i)} \frac{p(z_t \mid m_{t-1}^{(i)}, x_t^{(i)})\; p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_t)}{\frac{p(z_t \mid m_{t-1}^{(i)}, x_t)\; p(x_t^{(i)} \mid x_{t-1}^{(i)}, u_t)}{p(z_t \mid m_{t-1}^{(i)}, x_{t-1}^{(i)}, u_t)}} \tag{8}$$

$$= w_{t-1}^{(i)} . p(z_t \mid m_{t-1}^{(i)}, x_{t-1}^{(i)}, u_t) \tag{9}$$

$$= w_{t-1}^{(i)} \int p(z_t \mid x') p(x' \mid x_{t-1}^{(i)}, u_t) dx' \tag{10}$$

Re-sampling: Particles are drawn with replacement with their likelihood of being drawn proportional to their importance weights. Re-sampling should be done only when the Effective Sample Size(N_{eff} is less than $N/2$ so as to avoid loss of good samples which leads to *particle impoverishment*.

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} (w^{(i)})^2} \tag{11}$$

Note that the weights used to evaluate N_{eff} should be normalized.

Mapping: Each particle maintains an individual map. The map estimate $p(m_t^{(i)} \mid x_{1:t}^{(i)}, z_{1:t})$ is computed as explained Chapter 4. Note that the map for each particle is updated at each timestep.

Results

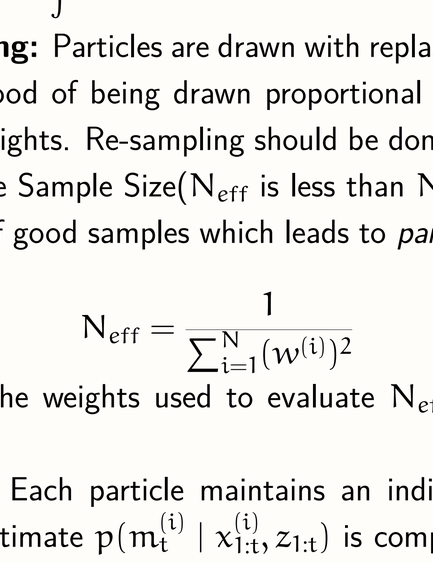

Figure 1: The map obtained when the robot poses are accurately known.

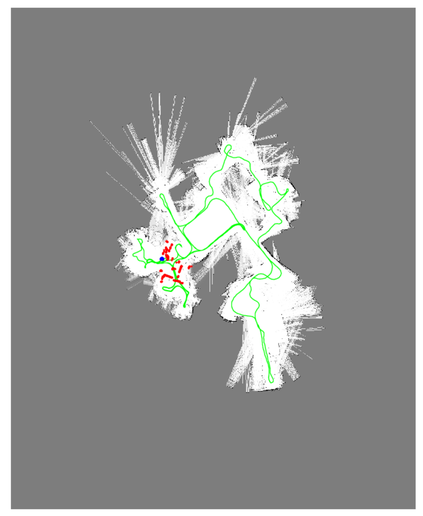

Figure 2: The map obtained by using 15 particles to represent the robot

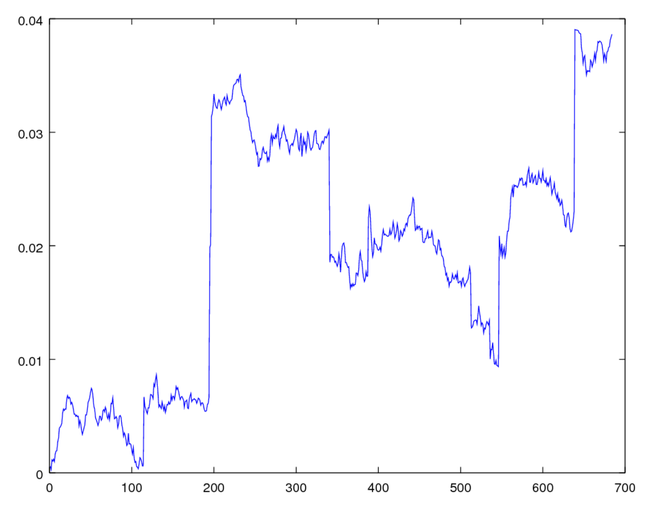

Figure 3: The distance between real and predicted poses at each timestep(in meters)

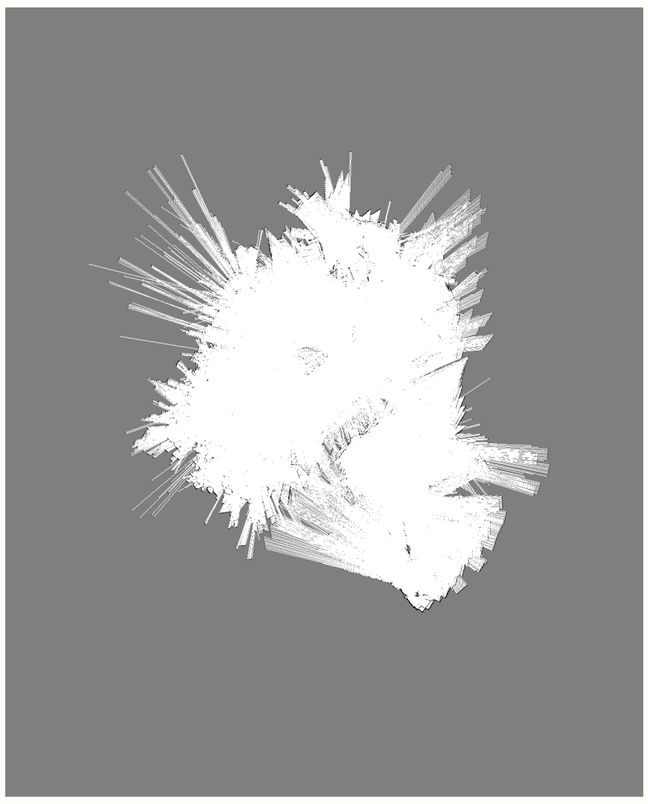

Figure 4: The map obtained by using only motion model as basis of localization

References

[1] SLAM course offered by Professor Cyrill Stachniss, University of Freiburg,

[2] C. Stachniss and G. Grisetti. *Mapping results obtained with Rao-Blackwellized particle filters*.

[3] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. *Monte carlo localization for mobile robots.* In Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA), Leuven, Belgium, 1998.

[4] M. Montemerlo and S. Thrun. *Simultaneous localization and mapping with unknown data association using FastSLAM.* In Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA) , pages 1985-1991, Taipei, Taiwan, 2003.

[5] Source Code for the entire project `https://github.com/Abhipanda4/SLAM-Algorithm`