Statistical Techniques in Robotics (16-831, F11) Lecture#04 (Monday September 12)

Particle Filters: The Good, The Bad, The Ugly

Lecturer: Drew Bagnell *Scribe:Tommy Liu* [1]

This lecture is all about Particle Filters, the good, the bad, and the ugly. We talk about exactly why nobody implements particle filters in the vanilla form learned in last class and some of the tricks used to fix what's broken about the vanilla particle filters.

1 Particle Filters: The Good

1. Particle filters can answer most queries by looking at a modest number of samples of $Bel(x)$:
This is good because we do not need (and indeed, in most cases would not have) a complete solution of $Bel(x)$

2. The performance of particle filters scale with the amount of resource:
Greater computing power leads to more particles, and better results

3. The belief doesn't need any parametric form, as opposed to Kalman filters, which requires gaussian beliefs

4. (Normalized) Importance Sampling [lecture #3]:
Even if we cannot sample from

$$p(x_{t+1}|x_t, z_{t+1}, u_t) \propto p(z_{t+1}|x_{t+1}) \cdot p(x_{t+1}|x_t, u_t) \tag{1}$$
$$p(x_{t+1}|x_t, z_{t+1}, u_t) = \frac{obs.motion}{Z} \tag{2}$$

we can use some distribution q instead of p to sample from, and factor out importance weights [lecture #3]. However, unless q closely matches p, the weights would get increasingly smaller.

5. Works with a finite number of particles:
Particle filters (should) use resampling to do "survival of the fittest", thus sampling densely around where the best estimates lie.

6. Particle filters are "easy to implement", in the sense that the theory is easy to understand and put into code. In reality working particle filters are rarely implemented in vanilla form and require a few adjustments to produce any meaningful results, which we will discuss in "The Bad" section.

[1]Some content adapted from previous scribes: Joydeep Biswas

2 Particle Filters: The Bad

2.1 Loss of diversity

2.1.1 The Problem

Repeated resampling in the absence of any actual sensory observations could lead to loss of diversity. This problem is illustrated by the following example of a robot localizing itself in two dimensions on a map with two (identical) rooms.

Fig. 1 shows the state of a particle filter at a particular instant of time (say, $t = 0$). Suppose our particle filter has 8 particles, and at $t = 0$ we have 4 particles in each room, all with equal weights. This means the robot is 50/50 on which room it is in, but it know it's in the center of one of the two rooms. Now let's say the robot's motion model is "standing still", and the observation model is "see nothing". This means after updating based on motion model and re-weighting based on the observation model, neither the locations of the particles nor the weights are changed. This makes sense because we haven't moved or gained any new information. The bad part is in the resampling step. During resampling, it is unlikely that every particle will be picked exactly once. It is more likely instead that some of the particles will be picked twice, while others not at all. In the absence of any new observations, over time, the distribution of particles in each room will drift into one room or the other, and the particle filter would end up in a state like that in Fig. 2. Hence for no good reason, resampling made our robot more certain it is in room two, while in reality no actual information has been gained at all.

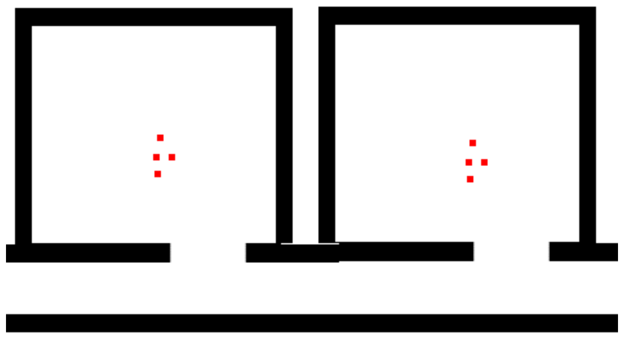

Figure 1: Loss of diversity example: State of particles at time step $t = 0$, with two equally weighted clusters and all particles with equal weights.

2.1.2 The Fix

1. Don't resample unless you have to!
   For example, you can measure the ratio of $\frac{min(w_i)}{max(w_i)}$ and only resample when it is greater than some threshold τ, or you can measure the $var(w_i)$ and resample when that grows to be greater than τ

2. Low Variance Resampling
   Before resampling, (conceptually) a list of the particles is created (Fig. 3), with the length

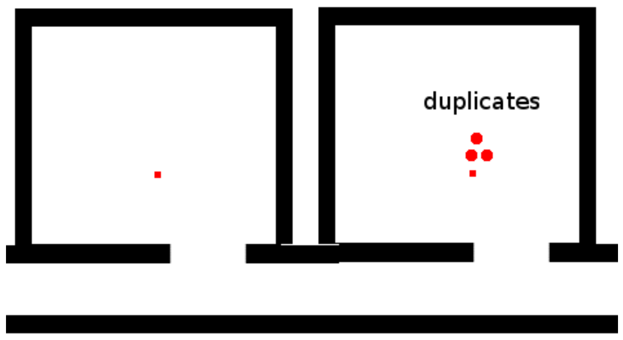

Figure 2: Loss of diversity example: State of particles at a later time step. The particles have essentially coalesced to a simgle hypotheses, located in room 2.

of the list equal to the sum of the weight of the particles, (say) W, such that each particle (conceptually) occupies a length in the list equal to its weight.
The naïve approach to resampling is to pick N random samples from the list, as depicted in Fig. 4. Intuitively, particles with weight greater than W/N should have been sampled at least once, particles with weight $2W/N$ at least twice, and so on. However, this is not ensured by the naïve algorithm.
An alternative algorithm is to find all particles with weights greater than W/N, draw one sample from them, decrease the length of their "segments" in the list by W/N, and draw the remaining samples from this new (shortened) list. An even better algorithm, called "low variance sampling" (similar in principle to "stochastic universal sampling"[2]) draws N samples using a single randomly generated number. Let the random number generated be r,such that $r \in (0, W)$. It is assumed that r is drown from a uniform distribution. Then, the first sample is drawn from location r in the list, the second from location $\left[\left(r + \frac{W}{N}\right) \bmod W\right]$, the i'th sample from location $\left[\left(r + \frac{i \cdot W}{N}\right) \bmod W\right]$ and so on, until N new samples have been generated, as illustrated by Fig. 5.

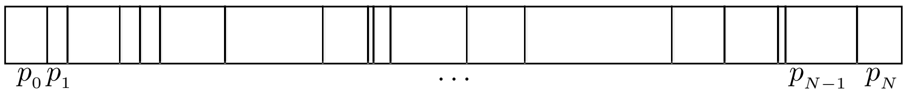

Figure 3: Resampling list

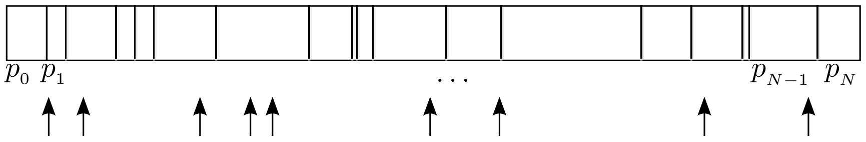

Figure 4: Naïve resampling: N samples are drawn from random locations, as indicated by arrows

[2]http://en.wikipedia.org/w/index.php?title=Stochastic_universal_sampling&oldid=356360569

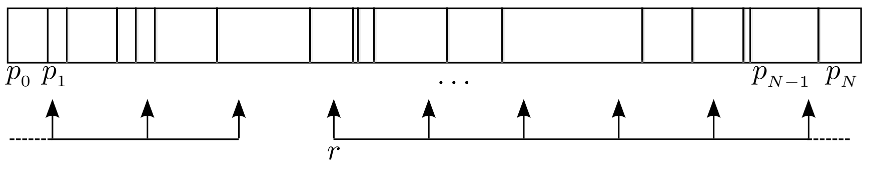

Figure 5: Low variance resampling: First sample drawn from random location r, rest picked at successive intevals of $\left[\frac{W}{N}\right]$

2.2 Really good observation models result in bad particle filters

2.2.1 The Problem

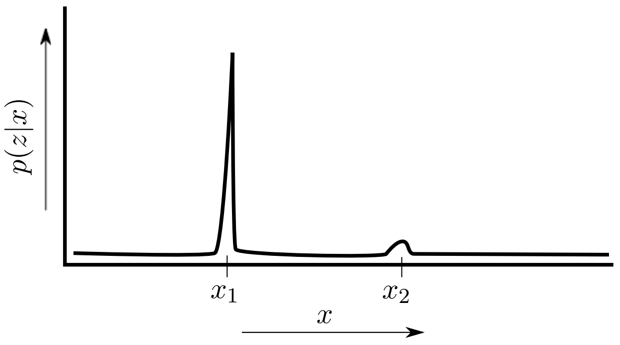

Figure 6: "Good" observation model: The observation likelihood has an accurate and confident (tall and narrow) peak around location x_1, and an erroneous low intensity peak around x_2.

If the observation model is really good (both accurate and confident), then the updates of the belief of the particle filter are likely to result in confident (closely clustered), yet incorrect estimates. This problem is best illustrated by the following example of a robot localizing itself in a single dimension x. Fig. 6 shows the observation likelihood $P(z|x)$ at an instant of time. Notably, the observation model has a very narrow but strong peak at location x_1, where the sensor believes that the robot is located. Also, there is a smaller peak due to some noisy readings (say) at location x_2. The observation likelihood is uniformly infinitesimal everywhere else. Given this observation likelihood, two problematic issues are likely to crop up:

1. Given the narrow width of the peak around x_1, it is likely that no particle would fall within the region covered by the peak. Thus, the weights of most particles would be driven to zero, and the accurate location estimate (as provided by the observation model) would be lost.
2. It is likely that a particle (or more) might lie under the region covered by the hump around x_2, and its (or their) weight(s) would increase with respect to the other particles, which (as described by the last point) would have infinitesimal weights. This would result in the particle filter providing a confident, albeit completetely wrong estimate of the robot's location.

2.2.2 The Fix

1. Low pass the observation model, or convolve the observation model with a Gaussian, thus smearing it out.
2. Throw more particles at it and hope we capture the observation model better
3. sample from a better ”q”, the sampling distribution we substituted for the desired ”p” distribution
4. Raise the likelihood values returned by the observation model to a power c where $c \in (0, 1)$. A commonly chosen value of c is $1/M$, where M is the dimensionality of the observation, thus taking the geometric mean of the observation values. Intuitively, this means we're undercounting the observation.
5. Rao-Blackwellization Compute some dimensions of the distribution analytically, and sample the rest, this partial sampling method basically says: if we can do any part of the sampling analytically instead, do it and get better accuracy for at least that part.

2.3 Non-deterministic behaviour

Given the random sampling necessary for a particle filter, it is unlikely that two particle filters, even with identical initial states and observations of time, would follow the exact same history of estimates. For some applications and for some users, it is important to be in complete control of the behaviour of the software, and non-deterministic behaviour is unnerving. Missile guidance and security applications come to mind here

2.4 Mismatch between p and q

The farther q is to p, the lesser it will be sampling the modes of p. Eventually, the particle filter could entirely loose track of the modes of p, and degenerate into overconfident, yet incorrect estimates.

2.5 Unable to measure performance

We can't tell how well a particle filter is doing at any point during the operation unless we have the groud truth data. The filter itself doesn't provide any ”confidence” measures.

3 Particle Filters: The Ugly (Misconceptions)

1. “*Particle filters sample from a motion model, weight by an observation model*”
   Particle filters do not *always* sample from a motion model and then weight by an observation model. They *could* sample from other distributions, if they were available, easy to sample from, and/or more accurate.

2. "*Particle filters are for localization*"
   Particle filters are not used *exclusively* for localization