Statistical Techniques in Robotics (16-831, F09) Lecture #4 (Thursday Sept. 4)

Sample-Based Filters and Normalized Importance Sampling

Lecturer: Drew Bagnell *Scribe: Elliot Cuzzillo*

1 Filtering

We have seen Bayes filters applied to Markov localization, and applied to occupancy or density mapping. We encounter problems with Bayes filters when either the Markov or the Bayes assumptions are violated; that is, when there are correlations in time, or correlations between sensors. Also, we sometimes encounter computational complexity issues with Bayes filters.

So, we explore a new kind of filtering which may help with these problems: Monte Carlo sample-based filters.

2 Background

Note that when we take the expectation of an expression $\mathbf{E}[f(X)]$, we actually are giving two arguments to the expectation function: a probability distribution or density function p, and a function f. Then expectation is always an integration.

Monte Carlo is a method by which we might do this integration. We have

$$\{X\}_{1...a}$$

a vector random variable. If all the X_i are independent, then we have $P(\vec{X}) = \prod_{i=1}^{R} P_i(X_i)$. If all the distributions are the same, it's independently identically distributed, or "i.i.d."

Also, the Strong Law of Large Numbers says that $Y_R = 1/R \sum f(X_i)$ converges to $\mathbf{E}_P[f(X_i)]$ as R grows large.

3 Monte Carlo

Now suppose we have a filter, and a belief distribution $\text{bel}_t(x)$. Then from the Strong Law of Large Numbers, we can answer any question we want about the distribution. For instance, if we want the variance, all we need is $\mathbf{Var}(X) = \mathbf{E}[X^2] - \mathbf{E}[X]^2 = Y_R = (1/R \sum X_i^2) - (1/R \sum X_i)^2$. Note that the computational complexity of finding the variance is independent of the number of dimensions of $\vec{X}$; it's only dependent on the number of samples. This is a nice property of Monte Carlo integration.

4 Application to Robot Filtering

Suppose our robot is in a given state x_0, and we want to sample from bel(x_1). Which is to say, we want to sample from $\frac{p(z_1|x_1)p(x_1|x_0)}{p(z_1)}$ where z_i is our observation at time i. It's most often not difficult to sample from the motion model from a particular state, since the motion model is usually a well-behaved parametric distribution. But sampling from all states that are possible given one observation is difficult.

This leads to the formulation of the problem of importance sampling.

5 Normalized Importance Sampling

The problem is, we have a distribution q(x) from which we can easily take samples, and another distribution p(x) we can't sample directly, but which we would like to sample from. Then,

$$\mathbf{E}_p[f] = \sum p(x)f(x) = \sum p(x)f(x)\frac{q(x)}{q(x)} = \sum q(x)\left(\frac{f(x)p(x)}{q(x)}\right) = \mathbf{E}_q[w(x)f(x)]$$

where $w(x) = \frac{p(x)}{q(x)}$. In our robot-filtering application, $q(x)$ is the distribution of current states given only past states (i.e. the motion model), and $p(x)$ is the full belief distribution, $\text{bel}_t(x)$.

For convenience, we then say that $\mathbf{E}_p[x_1]$ (the goal of our computation) is proportional to $1/R\sum_{i=1}^{r} p(z_1|x_1)f(x)$ (where we say proportional because the distribution is not normalized to 1). So, after observing this, we normalize:

$\tilde{w_i} = p(z_1|x_1)$, and $w_i = \frac{\tilde{w_i}}{\sum_i \tilde{w_i}}$.

So, as an algorithm:

Algorithm 1 Basic Particle Filter Algorithm

```
for all i do
  sample x_0^i from p(x_0)
end for
for all t do
  for all i do
    sample x_t^i from p(x_t|x_{t-1}^i)
    w_i  *= p(z_t|x_t^i)
  end for
  ŵ_i = w_i / Σ w_i
end for
E[f(x)] = Σ_i ŵ_i f(x_i)
```

Note: If the sensor model is too accurate, it is likely that all the w_i will be zero, and the algorithm will fail.

5.1 Resampling

This way, we have an accurate model if we have infinitely many samples, but in practice, the motion model will spread samples out and leave highly likely areas underresolved, and highly unlikely areas with many unnecessary particles with low weight.

To fix this problem, we periodically resample our particles according to their weight, sampling from the distribution $P(x = x_i) = w_i$, where w_i is the normalized weight of x_i. (There will be duplicates.) We then reset all the w_i to be equal. Our algorithm is now known in various fields as Normalized Importance Sampling with Resampling, Sampling Importance Resampling, Particle Filter, or Condensation Filter.

6 Extensions

However, this is still problematic. To begin with, if we always resample with the same frequency, but nothing has changed in the world, we will lose variance in our particles through random attrition. Some particles will go unsampled in a random resampling, despite being not any less likely. One idea is to do Low-Variance Resampling, where you sample particle x_i at least n times if $w_i \cdot R$ is at least n, and then randomly sample from the fractional remainder.

Another idea is to adaptively sample more particles when the sum of the un-normalized weights is too small, because small un-normalized weights indicate highly unlikely particles.

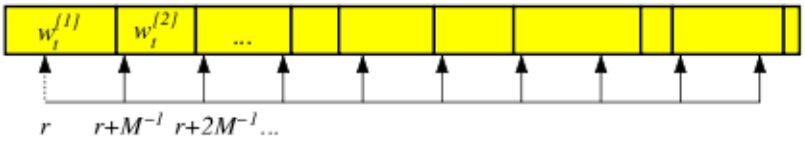

Figure 4.7 Principle of the low variance resampling procedure. We choose a random number r and then select those particles that correspond to $u = r + (m-1) \cdot M^{-1}$ where $m = 1, \ldots, M$.

7 Further Extensions

In some cases where the motion model is highly inaccurate but the observation model gives a very accurate estimate of the state is to sample directly from the observation model and then weight by the likelihood under the motion model.

Almost all particle filters in practice sample from something smarter than the naive motion model; for instance, some update a Kalman filter on each timestep, and then sample from the resulting Kalman filter distribution. Alternately, one can use a lower-dimensional particle filter, capturing less of the state, and sample from that. In general, real particle filters make an attempt to use some of the observation information in the sampling step, and then correct the sampling with the true observation distribution. These methods are related to Rao-Blackwellization, which will be discussed later.