Statistical Techniques in Robotics (16-831, F12) Lecture#03 (Monday September 5)

Sampling Based Filters

Lecturer: Drew Bagnell *Scribes: M. Taylor, M. Shomin* [1]

1 Beam-based Sensor Model

Many common sensors in robotics are beam-based: Sonar, Radar, LIDAR/LADAR, etc. These sensors work interpreting a reflected signal to indicate distance measurements from the sensor to the nearest solid object along a particular vector.

Our observation of one such sensor scan z consists of k measurements.

$$z = \{z_1, z_2, ..., z_k\} \tag{1}$$

We assume that individual measurements are indepentent of one another given the robot's pose.

$$P(z|x, m) = \prod_{k=1}^{k} P(z_k|x, m) \tag{2}$$

In reality, this assumption does not always hold. As an example, if the angle between beams is sufficiently small and detected objects are suitably large, it is very likely that one beam will return a distance measurement similar to its immediate neighbor. This assumption is still adequate enough to form a basic functional model.

Typically, we imagine that a beam-based sensor will follow a gaussian probability distribution with the expected value centered on the actual distance from the sensor to the target as seen in figure 1.

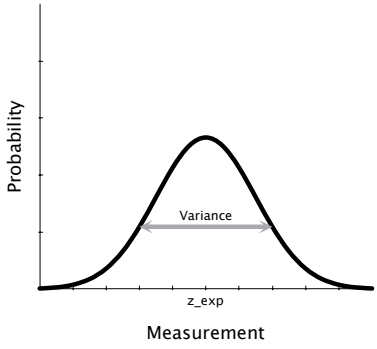

Figure 1: Sensor with a particularly poor (large variance) Gaussian distribution

Typical sources of error in range measurements can include:

[1] Slightly modified from previous scribe: Nate Wood

- Beams reflected by small obstacles
- Beams reflected by people or moving objects
- Sensor crosstalk and multipath interference
- Maximum range measurements
- Random noise

One potential beam-based sensor model consists of four components, as shown in figures 2, 3, 4, and 5.

We can linearly combine these distributions to create our sensor model, as seen in figure 6.

We are left with the question ofhow to choose our model parameters. We have four mixture variables representing the influence of each of the sensor modes on the resuling mixture density. These variables ideally represent the probability of each mode being responsible for a single measurement.

Because we cannot simply extract these values, common practice is to empirically take many measurements with the sensor across a fixed distance and compare them with samples drawn from the model, and then hand-tune the variables until the observations match the samples.

2 Monte Carlo Method

As discussed in previous lectures, the belief of a robot, for example, is defined as:

$$Bel(x_t) := p(x_t|u_{1:t}, z_{1:t}) \tag{3}$$

If we wish to use the belief of x_t in any useful manner we are going to use the expected value. The expected value of any function f is given by:

$$\mathrm{E}[f] = \sum_x f(x)p(x), \tag{4}$$

where $p(x)$ is the distribution of x. Rarely, if ever, is the distribution known in closed form. However, if you can generate samples of x_i from the distribution $p(x)$ the expectation can be estimated as:

$$\mathrm{E}[f] \approx \frac{1}{N} \sum_{i=1}^{N} f(x_i). \tag{5}$$

This sampling based approach falls into a general class of algorithms called Monte Carlo Methods. This method of using random sampling to compute results have been reinvented/rediscovered many times, with the name being given by physicists at Los Alamos Scientifc Laboratory in reference to the Monte Carlo Casino in Monaco.

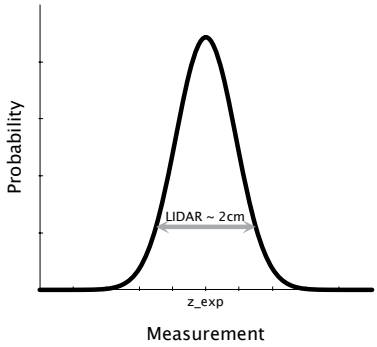

Figure 2: Probability distribution from a good sensor value

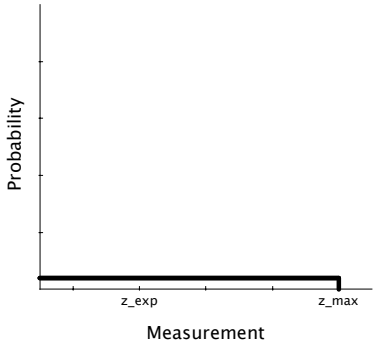

Figure 3: Probability distribution of receiving a random sensor value

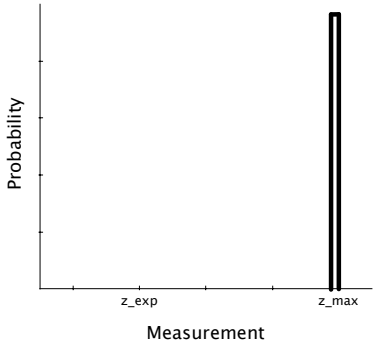

Figure 4: Probability distribution of receiving a value indicative of the sensor's maximum range (no return echo)

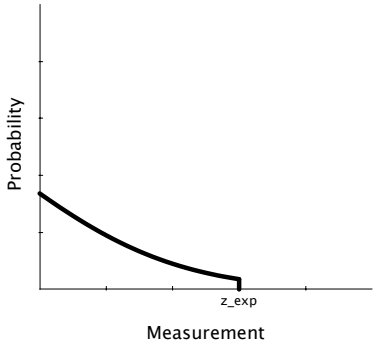

Figure 5: Probability distribution of an unexpected or moving object close to the sensor

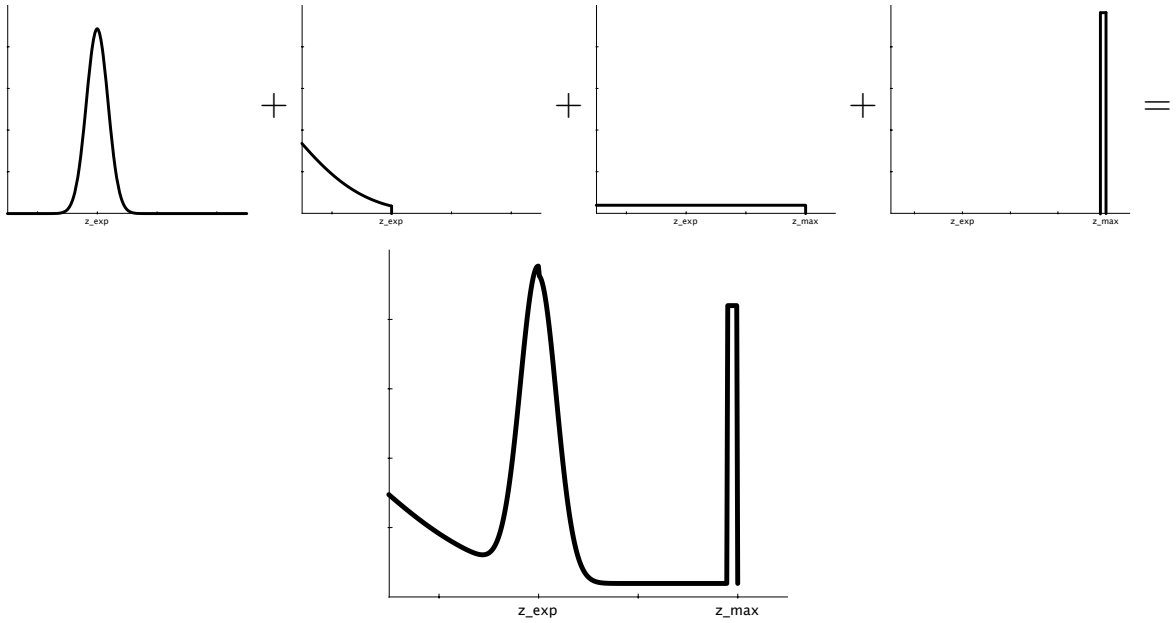

Figure 6: Combined beam-based sensor model

2.1 Example 1: Area Estimation

Suppose we have a shape we wish to calculate the area of, as depicted by the amoeba shape in Fig. 7. Conceptually we can find the desired area by randomly sampling the area and taking the proportion of samples that fell inside the area to the total number of samples and multiplying by the area of the box.

$$A_{amoeba} \approx A_{total} \frac{S_{inside}}{S_{total}} \tag{6}$$

More formally:

- Let $X = [x\ y]^T$ be our uncertain quantity (Random Variable).
- Assume we have an indicator function $Y = I(X)$ which takes the value 1 if X lies in the area and 0 otherwise.
- We can sample uniformly from X
- The percentage of the total area the desired area covers is the expected value of the indicator function.

$$\mathrm{E}[Y] \approx \frac{1}{N} \sum_{i=1}^{N} I(X_i) \tag{7}$$

- The area of the shape then

$$A_{amoeba} \approx \mathrm{E}[Y] A_{total} \tag{8}$$

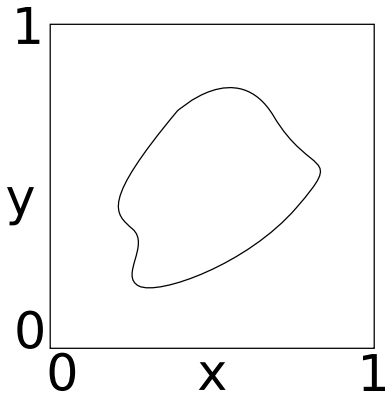

Figure 7: Area we wish to calculate

One of the advantages of Monte Carlo methods is that they are not significantly affected by dimensionality. For example, if we are instead trying to estimate the volume of a three-dimensional object, we do not need significantly more samples, so long as those samples are randomly chosen across all three dimensions. The same is true of other n-dimension objects.

Additionally, while they are not explored here, adaptive Monte Carlo methods exist which can dynamically change the sample distribution such that the samples have a higher probability of being informative.

2.2 An aside on π

The classical example is a Monte Carlo estimate of π. To do this, we make the room a circle, estimate random X, and count the samples of X inside the circle.

2.3 Example 2: Which room am I in?

We have a map with rooms numbered 1, 2, 3 as shown in Fig. 8.

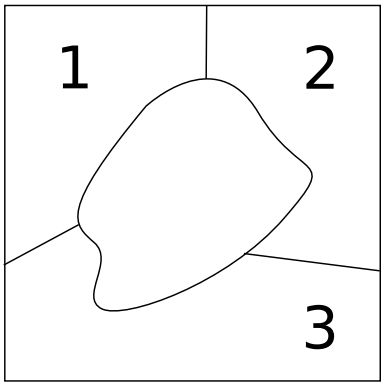

Figure 8: Rooms in which robot may be located

We want probability that robot is in room $\{1, 2, 3\}$. But how would we write this as an expectation?

We define rooms as subsets of continuous space, and invent a function $f()$ which is 1 if $x_t \in room(1)$, 0 otherwise. This is essentially the indicator function of the room: 1_{room}

Then the probability the robot is in the room is the also the expectation of the indicator function: $Pr(robot\ is\ in\ room(1)) = E[1_{room(x)}]$. In our case, this is the sum of the probabilities of being in a particular room. In general, the expectation of an indicator function is the probability of the probability of the indicated expression.

In many cases, we don't know $p(x)$ in closed form, but we can generate samples (X) with $pr(X = x)$. Supposing this is true, we can do

$$\frac{1}{n}\sum x_i \rightarrow_{n\rightarrow\infty} \mathrm{E}[x] \tag{9}$$

or more strongly

$$\frac{1}{n}\sum f(x_i) \rightarrow_{n\rightarrow\infty} \mathrm{E}[f(x)] \tag{10}$$

2.4 Return to Robots

In the case of robots we would like to compute the expected value of a function f knowing both actions, u, and measurements, z.

$$\mathop{\mathrm{E}}_{p(x_t|z_t,x_{t-1},u_{t-1})}[f(x_t)] \tag{11}$$

Knowing that we will never know a closed form solution for this distribution leaves the previously described sample based method. In order to use the Monte Carlo Method would require samples drawn from the distribution

$$p(x_t|z_t, x_{t-1}, u_{t-1}), \tag{12}$$

which can be rewritten as

$$\eta\ p(z_t|x_t)p(x_t|x_{t-1}, u_{t-1}). \tag{13}$$

The distribution is the product of the measurment model, $p(z_t|x_t)$, and the robot motion model, $p(x_t|x_{t-1}, u_{t-1})$. It is not obvious how to sample from the product of these distributions. It is, however, straightforward to sample from the robot motion model, a classic example of which is shown in Fig. 9.

3 Importance sampling (IS)

Although we are unable to sample from the required distribution, $p(x)$, we can use a trick that will allow us to sample from the known distribution, $q(x)$. The trick: Sample from the available

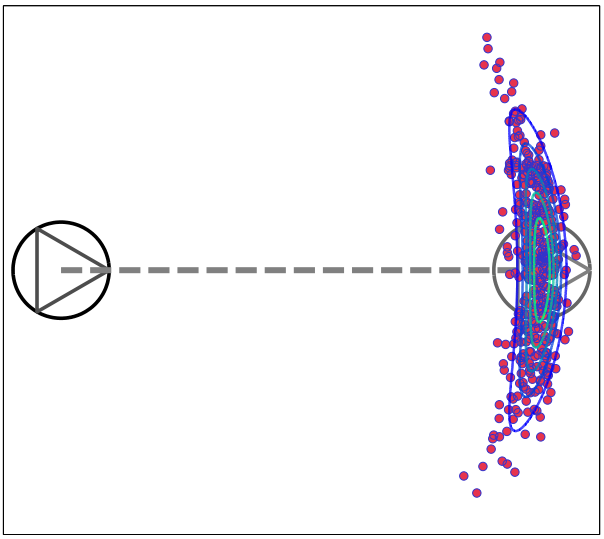

Figure 9: The motion model: Posterior distributions of the robot's pose upon executing the motion command illustrated by the dashed line. Isocontours are shown for the probabilty of the particle. This plot has been projected into 2-D. The original density is three-dimensional, taking the robot's heading direction θ into account.

distribution and reweight samples to fix it. Manipulating the expectation over $p(x)$ yields the following approximiation:

$$\begin{aligned}
\underset{p(x)}{\mathrm{E}}[f(x)] &= \sum p(x)f(x) && \text{(14a)}\\
&= \sum p(x)f(x)\frac{q(x)}{q(x)} && \text{(14b)}\\
&= \sum q(x)\frac{p(x)}{q(x)}f(x) && \text{(14c)}\\
&= \underset{q(x)}{\mathrm{E}}[\frac{p(x)}{q(x)}f(x)] && \text{(14d)}\\
&\approx \frac{1}{N}\sum_{i=1}^{N}\frac{p(x_i)}{q(x_i)}f(x_i). && \text{(14e)}
\end{aligned}$$

$\frac{p(x)}{q(x)}$ is called the importance weight. When we are undersampling an area, it weights it stronger; likewise, oversampled areas are weighted more weakly.

Consider the following (somewhat contrived) pair of functions:

Notice that there are places where p is nonzero and q is zero, which means we have regions to sample that we aren't sampling at all.

Our estimate of the expectation of p becomes:

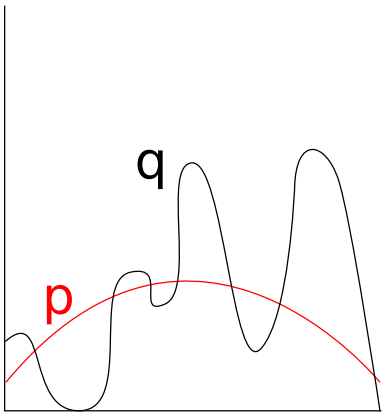

Figure 10: Comparison of distribution we wish to sample from, p, and distribution we can sample from, q.

$$\frac{1}{N}\sum_{i=1}^{N}\frac{p(x_i)}{q(x_i)}f(x_i) \rightarrow_{N\rightarrow\infty} E_p[f(x)] \tag{15}$$

3.1 Example

Set q as the motion model. Note that in reality we never use this – we use a better approximation of $p(x_t|x_{t-1})$ because the motion model takes too many samples to be reliable.

$$p = \eta\ p(z_t|x_t)p(x_t|x_{t-1}) \tag{16}$$

$$X^i \sim p(x_t|x_{t-1}) \tag{17}$$

$$w^i = \frac{p}{q} = p(z_t|x_t) \tag{18}$$

$$\bar{X} = \mathrm{E}[X] = \frac{1}{N}\sum_{i=1}^{N} w^i X^i \tag{19}$$

But notice that we dropped the normalizer from bayes rule! (if we don't know Z)

4 Normalized IS

We can normalize estimate using:

$$\frac{1}{N}\sum w^i \rightarrow_{N\rightarrow\infty} \sum q(x) = \sum p(x_t|x_{t-1})p(z_t|x_t) = Z \tag{20}$$

$$\tilde{w}^i = \frac{w^i}{\sum w^i} \tag{21}$$

$$\bar{X} = \sum \tilde{w}^i X^i \tag{22}$$

Algorithm 1 NIS 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 $p(x_t|x_{t-1}^i)$
    $w_i \quad * = p(z_t|x_t^i)$
  end for
  $\tilde{w}_i = \frac{w_i}{\sum w_i}$
end for
$E[f(x)] = \sum_i \tilde{w}_i f(x_i)$

5 SIR Filter (Particle filter, Condensation)

There is a bug in the NIS filter. The true path will be a sample with vanishing probability, so we have to throw out low probability samples and resample in high probability regions. This is a Particle filter.