EL2320 - Particle Filter Proposal Distribution

John Folkesson

Particle Filters
In Chap 8.3.6 and (Monte Carlo Localization With Mixture Proposal Distribution,
Thrun (thrun.mclmix.pdf))

Sampling Variance

The two sampling steps in the particle filter result in the particle distribution being close but not identical to the posteori. The difference is called Sampling Variance. By simple using more particles we can reduce this type of error.

Resampling

Although sample variance can be made small repeated resampling will eventually cause this error to grow to significant levels.

The book presents the paradox of the robot not moving and using no sensor eventually having M copies of a single particle.

Another name for this problem is *Particle Deprivation*. This means that there are no particles in places where the state has some probability of being.
A method of limiting this is adding random particles. This is not a great solution but can help.

Here is a demo of particle deprivation along with some solutions.
`http://www.csc.kth.se/~kootstra/applets/premature_convergence/`

Delayed Resampling, Variance Reduction

Instead of doing:
Phase II: resample or importance sampling

1. draw from the set of $\hat{x}_t^{(m)}$ with probability $w_t^{(m)} \rightarrow x_t^{(m)}$

Every iteration, it can be better to keep the weights as part of the discription for the next iteration, skiping the resample. So:
Phase I: propagate to new time using the model then becomes,

1. sample $p(x_t|x_{t-1}^{(m)}, u_t) \rightarrow \hat{x}_t^{(m)}$
2. compute $w_{t+1}^{(m)} = p(z_{t+1}|\hat{x}_t^{(m)})w_t^{(m)}$ (importance factor)

Low Variance Sampling

1. select a random number r on $[0, 1/M]$.
2. pick M particles with each particle index j corresponing to max j such that $r + k/M < \sum_{i=0}^{j} w_i$ for $k = 0...M-1$

This is both more efficient and maintains the same particle set for the case of all weights being equal.

Stratified Sampling

If the distribution is multimodal, (two or more peaks), then it is often better to do a seperate resample of particles of each peak. This is called Stratified sampling.

One first groups the particles into classes.

Then using the total wieght in each class one sets the number of resample points for each class.

One then does some sampling scheme to select that number of points from each set.

Particle Filter

In computer vision particle filter is called condensation algorithm and is used for tracking.

The particle filter is also called the bootstrap filter.

The particle filter is also called the Monte-Carlo filter.

Good explaination of the Particle Filter:
Monte Carlo Localization forMobile Robots, Dellaert et al.
(sampling-icra99.pdf)

Monte Carlo Localization, MCL

MCL is a specific Particle filter application. For localization of a robot.

It allows multimodal distributions to be represented.

It allows Global Localization.

Kidnapped Robot

If there are no particles near the correct position the filter fails.

This can be recovered from by injecting random particles.

The criteria can be the average of the measurement probabaility $p(\mathbf{z}_t|\mathbf{z}_{1..t-1}, u_{1..t-1})$

When we start having trouble explaining the measurements we may be lost.

The random particles can be selected based on the latest measurements for better efficiency.

Curse of Dimensionality, Number of Particles

There is this one choice to make: how many particles M to use. As one increases the dimension of the state one generally needs an exponentially increasing number of particles. This makes particle filters as described here impractical for more than a few dimensions.

A way to chose the number of particls adaptively is presented in: Adapting the Sample Size in Particle Filters Through KLD-Sampling, D. Fox, (adaptive-ijrr-2003.pdf)

KLD Sampling

Kullback-Lieber Divergence is a measure of how different two distributions are.

An approximation of this quantity as a function of our particle filter solution is used to determine when to stop generating new particles.

In practice what happens is spread out distributions get more particles than tight ones.

The algorithm requires a grid of binary bins over the state space.

KLD Sampling

First set all bins to empty (false), $k = 0$, $M = \infty$, and $i = 1$.

1. draw a particle $\mathbf{x}_{t-1}^j$ from particle set with probability w_{t-1}^j
2. sample the motion model $p(\mathbf{x}_t^i | \mathbf{x}_{t-1}^j, u_t)$
3. $w_t^i = p(z_t | \mathbf{x}_t^i)$
4. if bin$(\mathbf{x}_t^i)$ =empty (false).
   - $k = k + 1$
   - bin$(\mathbf{x}_t^i)$ =occupied (true).
   - if $(k > 1)$ $M = \frac{k-1}{2\epsilon}[1 - \frac{2}{9(k-1)} + \sqrt{\frac{2}{9(k-1)}}\gamma_\delta]^3$
5. i=i+1;
6. if $(i \geq M)$ stop

γ_δ is the upper $1 - \delta$ quantile of the standard normal dist.
ϵ is the 'error' as measured by the KLD.

Particle Filter

Input is a set of 'particles' $x_{t-1}^{(m)}$ drawn from the $p(x_{t-1}|z_1.., z_{t-1}, u_1..u_{t-1})$.
m runs from $1...M$

Phase I: propagate to new time using the model.

1. sample $p(x_t|x_{t-1}^{(m)}, u_t) \rightarrow \hat{x}_t^{(m)}$
2. compute $w_t^{(m)} = p(z_t|\hat{x}_t^{(m)})$ (importance factor)

Phase II: resample or importance sampling

1. draw from the set of $\hat{x}_t^{(m)}$ with probability $w_t^{(m)} \rightarrow x_t^{(m)}$

Particle Filter

Compare to Bayes Filter:

$$p(\mathbf{x}_t|\{\mathbf{z}_t\},\{\mathbf{u}_t\}) \propto p(\mathbf{z}_t|\mathbf{x}_t,\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})p(\mathbf{x}_t|\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})$$

$$\propto p(\mathbf{z}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{u}_t,\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\{\mathbf{z}_{t-1}\},\{\mathbf{u}_{t-1}\})(d\mathbf{x}_{t-1})^n$$

A more general is the concept of *proposal distribution* $g(x)$ and *target distribution* $f(x)$ related by importance weights $w = f/g$ on some set of particles.
$f(x) > 0 \Rightarrow g(x) > 0$

Then integrals over $f(x)$ get replaced by sums of $w^{(m)}$ for particles within the region of integration. (convergance as $M \to \infty$)

Other choices of proposal distribution lead to other weights and a different algorthim:
Monte Carlo Localization With Mixture Proposal Distribution, Thrun (thrun.mclmix.pdf)

Proposal Distribution

$$p(\mathbf{x}_t|\{\mathbf{z}_t\},\{\mathbf{u}_t\}) \propto p(\mathbf{z}_t|\mathbf{x}_t,\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})p(\mathbf{x}_t|\{\mathbf{u}_t\},\{\mathbf{z}_{t-1}\})$$
$$\propto (\frac{f(x)}{g(x)})g(x)$$

Problems arise with very accurate sensor measurements **x**. These can cause all the importance weights to be about 0.

A possible 'solution' is to change the proposal distribution to:

$g(x_t) = \frac{p(z_t|x_t)}{\pi(z_t)} = \frac{p(z_t|x_t)}{\int p(z_t|x_t)dx_t}$

Thus one samples points that could generate the observation.

But what about $w_i = \frac{f(x)}{g(x)}$?

Importance Weight, Method I

$$g(x_t) = \frac{p(z_t|x_t)}{\pi(z_t)} = \frac{p(z_t|x_t)}{\int p(z_t|x_t)dx_t}$$

Draw a second sample from $Bel(x_{t-1}) = p(x_{t-1}|z_{1:t-1}, u_{1:t-1})$
Then the pair x_t^i, x_{t-1}^i is distributed:
$\frac{p(z_t|x_t)}{\pi(z_t)} Bel(x_{t-1})$

$$w_i = [\frac{p(z_t|x_t)}{\pi(z_t)} Bel(x_{t-1})]^{-1} \frac{p(z_t|x_t, \{u_t\}, \{z_{t-1}\}) p(x_t|x_{t-1}, u_t) Bel(x_{t-1})}{p(z_t|u_{t-1}...z_1)}$$

$$\propto p(x_t|x_{t-1}, u_t)$$

(note that we can ignore anything without an x)
The problem remains however since we will chose point x_{t-1} that doesn't explain the measurements

Importance Weight, Method II

Do two sampling steps but now generate a set of x_t^i from each.
The first set is as before sampled from the points implied by z_t.
The second samples from the $Bel(x_{t-1})$.
One then generates the second set of x_t^i by sampling the forward $p(x_t|x_{t-1}, u_t)$ as in standard MCL.
Now use this second sample set to generate a density function approximation: $p(x_t|u_{1:t}, z_{1:t-1}) \approx p(x_t|u_t, d)$, where d is just to indicate this approximation.
This function then can generate out weight in places were we have no samples.

$$
\begin{aligned}
w_i &= [\frac{p(z_t|x_t)}{\pi(z_t)}]^{-1} \frac{p(z_t|x_t)p(x_t|u_t, d)}{p(z_t|u_t, d)} \\
&= p(x_t|u_t, d)
\end{aligned}
$$

Importance Weight, Method III

Sample as before to get x_t^i. Then sample backwards to get x_{t-1}^i.

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

$$\pi(x_t^i|u_t) = \int p(x_t^i|u_t, x_t - 1)dx_{t-1}$$

This leads to

$$w_i = \pi(x_t^i|u_t)Bel(x_{t-1})$$

And then one argues that for robot localization $\pi(x_t^i|u_t)$ is constant. One must then estimate $Bel(x_{t-1})$ by generating a density from the prior particle set.

Proposal Distributions

Point is there can be problems and there are some things one can do vis-a-vis the proposal distribution to address these.
The problems are that the samples may be thin or non-existant where the target distribution is largest.