Gaussian Filters

- The Kalman filter and its variants can only model **Gaussian distributions**

\[ p(x) = \frac{\text{det}(2\pi\Sigma)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)}{2} \]

Motivation

- Goal: approach for dealing with **arbitrary distributions**

Key Idea: Samples

- Use **multiple samples** to represent arbitrary distributions
**Particle Set**
- Set of weighted samples
  \[ \mathcal{X} = \{\langle x[i], w[i]\rangle \}_{i=1,...,N} \]
- The samples represent the posterior
  \[ p(x) = \sum_{i=1}^{N} w[i] \delta_{x[i]}(x) \]

**Particles for Approximation**
- Particles for function approximation
  - The more particles fall into an interval, the higher its probability density
  - How to obtain such samples?

**Importance Sampling Principle**
- We can use a different distribution \( g \) to generate samples from \( f \)
- Account for the “differences between \( g \) and \( f \)” using a weight \( w = f/g \)
- target \( f \)
- proposal \( g \)
- Pre-condition: \( f(x)>0 \Rightarrow g(x)>0 \)
Particle Filter

- Recursive Bayes filter
- Non-parametric approach
- Models the distribution by samples
- Prediction: draw from the proposal
- Correction: weighting by the ratio of target and proposal

The more samples we use, the better is the estimate!

Particle Filter Algorithm

1. Sample the particles using the proposal distribution
\[ x_t^{[i]} \sim \pi(x_t | \ldots) \]
2. Compute the importance weights
\[ w_t^{[i]} = \frac{\text{target}(x_t^{[i]})}{\text{proposal}(x_t^{[i]})} \]
3. Resampling: “Replace unlikely samples by more likely ones”

```
Particle_filter(\mathcal{X}_{t-1}, u_t, z_t):
1: \mathcal{X}_t = \mathcal{X}_t = \emptyset
2: \text{for } m = 1 \text{ to } M \text{ do}
3: \quad \text{sample } x_t^{[m]} \sim \pi(x_t)
4: \quad w_t^{[m]} = \frac{p(z_t | x_t^{[m]})}{\pi(x_t^{[m]})}
5: \quad \mathcal{X}_t = \mathcal{X}_t + \{x_t^{[m]}, w_t^{[m]}\}
6: \text{endfor}
7: \text{for } m = 1 \text{ to } M \text{ do}
8: \quad \text{draw } i \text{ with probability } \propto w_t^{[i]}
9: \quad \text{add } x_t^{[i]} \text{ to } \mathcal{X}_t
10: \text{endfor}
11: \text{return } \mathcal{X}_t
```

Monte Carlo Localization

- Each particle is a pose hypothesis
- Proposal is the motion model
\[ x_t^{[i]} \sim p(x_t | x_{t-1}, u_t) \]
- Correction via the observation model
\[ w_t^{[i]} = \frac{\text{target}(x_t^{[i]})}{\text{proposal}(x_t^{[i]})} \propto p(z_t | x_t^{[i]}, m) \]
Particle Filter for Localization

Particle filter($X_{t-1}, u_t, z_t$):
1: $\tilde{X}_t = X_t = \emptyset$
2: for $m = 1$ to $M$ do
3: sample $x_t^{[m]} \sim p(x_t | u_t, x_t^{[m]}_{t-1})$
4: $w_t^{[m]} = p(z_t | x_t^{[m]})$
5: $\tilde{X}_t = \tilde{X}_t + (x_t^{[m]}, w_t^{[m]})$
6: endfor
7: for $m = 1$ to $M$ do
8: draw $i$ with probability $\propto w_t^{[i]}$
9: add $x_t^{[i]}$ to $X_t$
10: endfor
11: return $X_t$

Application: Particle Filter for Localization (Known Map)

Resampling

- Survival of the fittest: “Replace unlikely samples by more likely ones”
- “Trick” to avoid that many samples cover unlikely states
- Needed as we have a limited number of samples

Resampling

- Roulette wheel
- Binary search
- $O(n \log n)$
- Stochastic universal sampling
- Low variance
- $O(n)$
Low Variance Resampling

```
low_variance_resampling(X_t, W_t):
1:   \tilde{X}_t = \emptyset
2:   r = \text{rand}(0, M^{-1})
3:   c = w_t^{[1]}
4:   i = 1
5:   for m = 1 to M do
6:     U = r + (m - 1) \cdot M^{-1}
7:       while U > c
8:         i = i + 1
9:       c = c + w_t^{[i]}
10:      endwhile
11:     add x_t^{[i]} to \tilde{X}_t
12:   endfor
13:   return \tilde{X}_t
```
motion update

measurement

weight update

resampling
motion update

measurement

weight update

resampling
motion update
measurement
weight update
resampling
Summary – Particle Filters

- Particle filters are non-parametric, recursive Bayes filters
- Posterior is represented by a set of weighted samples
- Not limited to Gaussians
- Proposal to draw new samples
- Weight to account for the differences between the proposal and the target
- Work well in low-dimensional spaces

Summary – PF Localization

- Particles are propagated according to the motion model
- They are weighted according to the likelihood of the observation
- Called: Monte-Carlo localization (MCL)
- MCL is the gold standard for mobile robot localization today