Particle Filter

- Recall: Discrete filter
    - Discretize the continuous state space
    - High memory complexity
    - Fixed resolution (does not adapt to the belief)

- Particle filters are a way to **efficiently** represent **non-Gaussian distributions**

- Basic principle
    - Set of state hypotheses (“particles”)
    - Survival-of-the-fittest

Mathematical Description

- Set (actually a multi-set) of weighted samples

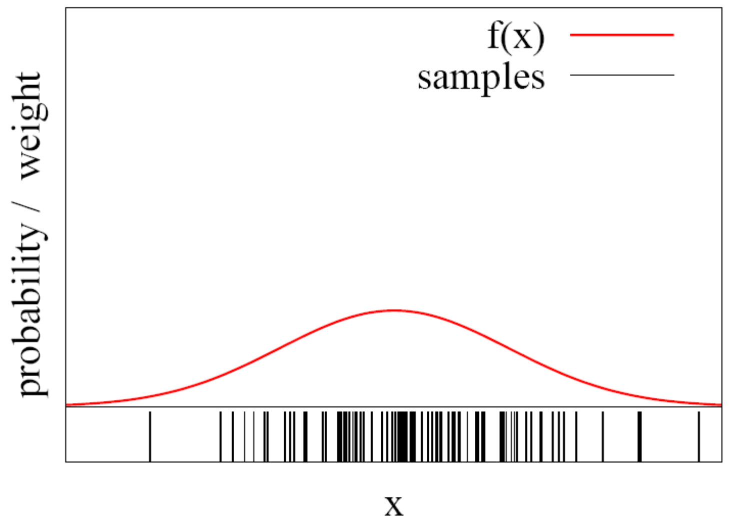
$$S = \left\{ \left\langle \underset{\substack{\uparrow \\ \text{state hypothesis}}}{s^{[i]}}, \underset{\substack{\uparrow \\ \text{importance weight}}}{w^{[i]}} \right\rangle \mid i = 1, \ldots, N \right\}$$

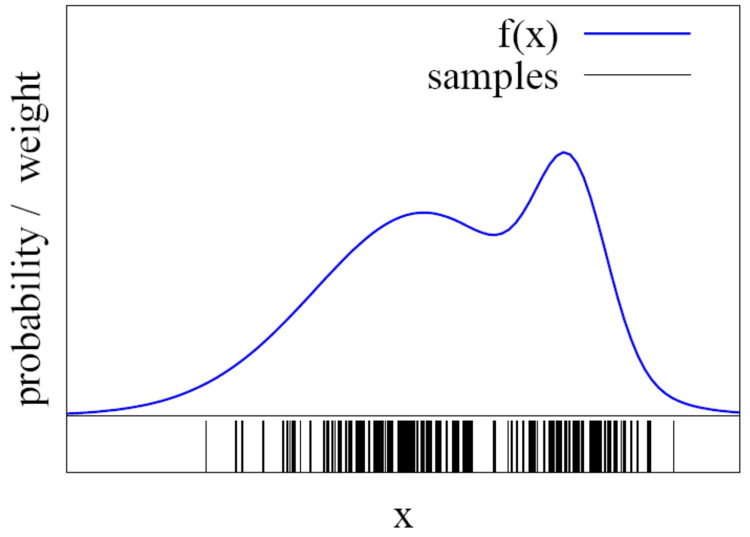
- The samples represent the posterior

$$p(x) = \sum_{i=1}^{N} w_i \cdot \overset{\begin{cases} 1 & \text{if } x = s^{[i]} \\ 0 & \text{otherwise} \end{cases}}{\swarrow}\!\!\delta_{s^{[i]}}(x)$$

Function Approximation

- Particle sets can be used to approximate functions





- The more particles fall into an interval, the higher the probability of that interval

Bayes Filter with Particle Sets

- Measurement update

$$Bel(x) \leftarrow p(z|x)\overline{Bel}(x)$$

$$= p(z|x) \sum_i w_i \, \delta_{s^{[i]}}(x) = \sum_i p\left(z \middle| s^{[i]}\right) w_i \, \delta_{s^{[i]}}(x)$$

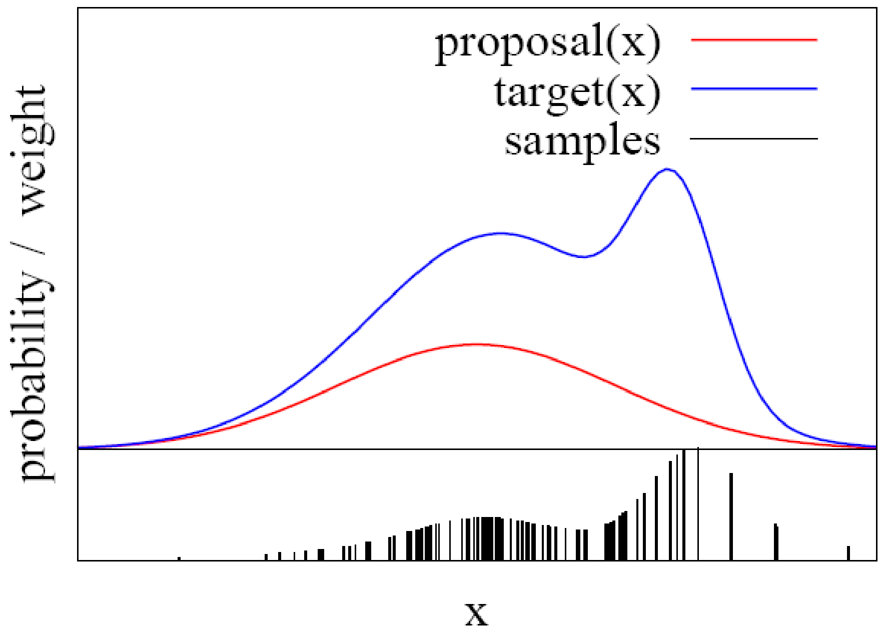
- Motion update

$$\overline{Bel}(x) \leftarrow \int p(x \mid u, x') \, Bel\,(x') dx'$$

$$= \int p(x \mid u, x') \sum_i w_i \, \delta_{s^{[i]}}(x') dx' = \sum_i p(x \mid u, s^{[i]}) \;\; w_i$$

Importance Sampling Principle

- We can even use a different distribution g to generate samples from f
- By introducing an importance weight w, we can account for the "differences between g and f"
- $w = f / g$
- f is called target
- g is called proposal
- Pre-condition:
  $f(x)>0 \rightarrow g(x)>0$



Particle Filter Algorithm

- Sample the next generation of particles using the proposal distribution

- Compute the importance weights:

  $$weight = target\ distribution\ /\ proposal\ distribution$$

- Resampling: “Replace unlikely samples by more likely ones”

Particle Filter Algorithm

```
1.  Algorithm **particle_filter**($S_{t-1}, u_t, z_t$) returns $S_t$:
2.  $S_t = \emptyset, \quad \eta = 0$
3.  **For** $i = 1, \dots, n$                                              *Generate new samples*
4.          Sample index $j(i)$ from the discrete distribution given by $w_{t-1}$
5.          Sample $x_t^i$ from $p(x_t \mid x_{t-1}, u_t)$ using $x_{t-1}^{j(i)}$ and $u_t$
6.          $w_t^i = p(z_t \mid x_t^i)$                                    *Compute importance weight*
7.          $h = h + w_t^i$                                                *Update normalization factor*
8.          $S_t = S_t$ È $\{< x_t^i, w_t^i >\}$                           *Add to new particle set*
9.  **For** $i = 1, \dots, n$
10.         $w_t^i = w_t^i / h$                                            *Normalize weights*
```

Particle Filter Algorithm

$$\boxed{Bel(x_t) = \eta\, p(z_t \mid x_t) \int p(x_t \mid x_{t-1}, u_t)\, Bel(x_{t-1})\, dx_{t-1}}$$

target distribution

draw x^i_{t-1} from $Bel(x_{t-1})$

draw x^i_t from $p(x_t \mid x^i_{t-1}, u_t)$

proposal distribution

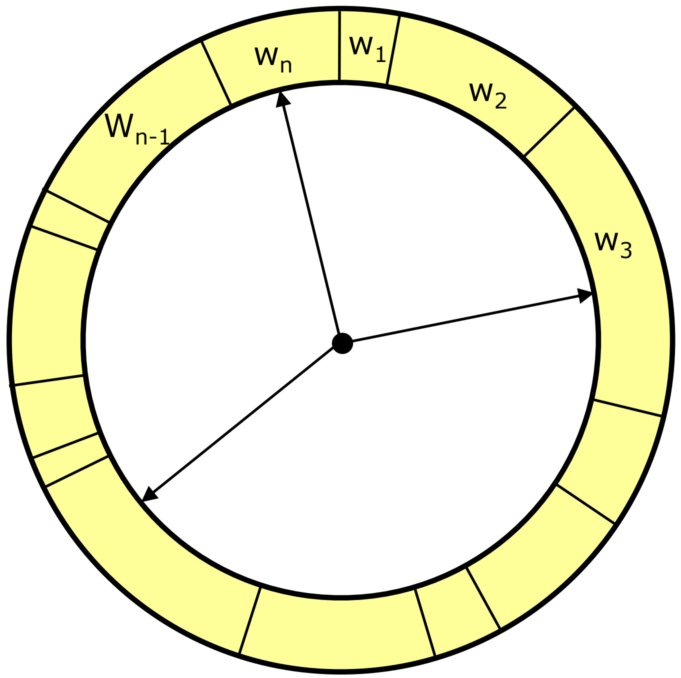
Importance factor for x^i_t:

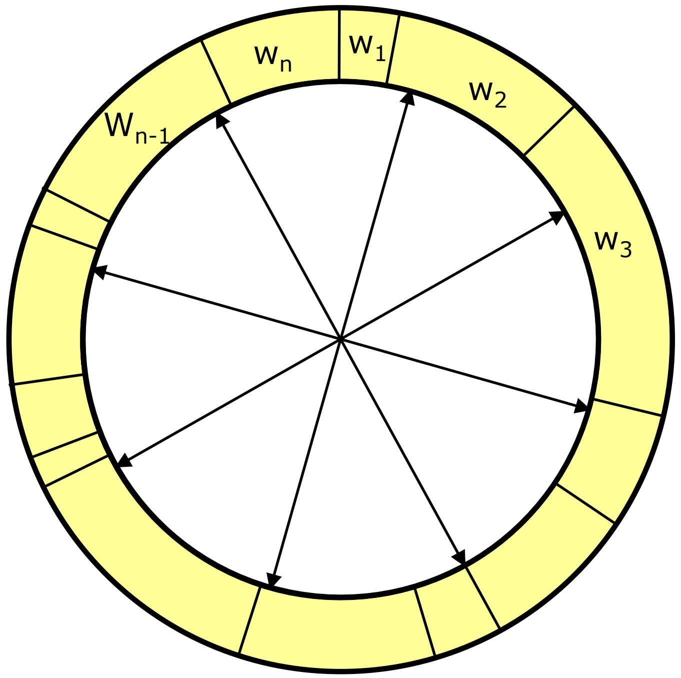
$$\begin{aligned} w^i_t &= \frac{\text{target distribution}}{\text{proposal distribution}} \\ &= \frac{\eta\, p(z_t \mid x_t)\, p(x_t \mid x_{t-1}, u_t)\, Bel\ (x_{t-1})}{p(x_t \mid x_{t-1}, u_t)\, Bel\ (x_{t-1})} \\ &\propto p(z_t \mid x_t) \end{aligned}$$

Resampling

- **Given**: Set S of weighted samples.
- **Wanted**: Random sample, where the probability of drawing x_i is given by w_i.
- Typically done n times with replacement to generate new sample set S'.

Resampling





- Roulette wheel
- Binary search, O(n log(n))

- Stochastic universal sampling
- Systematic resampling
- Linear time complexity O(n)
- Easy to implement, low variance

Resampling Algorithm

```
1.  Algorithm **systematic_resampling**(S,n):

2.  $S' = \emptyset, c_1 = w^1$
3.  For $i = 2 \ldots n$                          Generate cdf
4.      $c_i = c_{i-1} + w^i$
5.  $u_1 \sim U]0, n^{-1}], i = 1$                Initialize threshold

6.  For $j = 1 \ldots n$                          Draw samples …
7.      While ( $u_j > c_i$ )                     Skip until next threshold reached
8.          $i = i + 1$
9.      $S' = S' \cup \{< x^i, n^{-1} >\}$        Insert
10.     $u_{j+1} = u_j + n^{-1}$                  Increment threshold

11. Return $S'$
```

sample from the uniform distribution in $(0, n^{-1}]$

Also called **stochastic universal sampling**

Summary – Particle Filters

- Particle filters are an implementation of recursive Bayesian filtering
- They represent the posterior by a set of weighted samples
- They can model arbitrary and thus also non-Gaussian distributions
- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter