Lecture 3 – Particle filters

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

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

$$= \eta\, \underbrace{p(z_t \mid x_t)}_{\text{Sensor model}}\ \underbrace{\overline{Bel}(x_t)}_{\text{After motion update}}$$

Belief represented as particles:

$$\{x_t^1, x_t^2, \ldots x_t^M\} : M \text{ particles}$$

Expected state (pose, e.g.) of the robot –

$$\underset{Bel(x_t)}{E}(X_t) = \int Bel(x_t)\, x_t\, dx_t \rightarrow \text{Sampling form } Bel(x_t) \text{ is hard!}$$

(product of complex distributions)

How to sample?

1. Exhaustive Discretization: [Measure values for all x, y, θ]

   for x in $\{X\}$

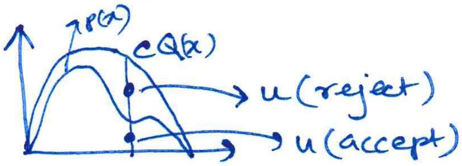
   $A[x] = Bel(x)$ → Very expensive!

   → sample form this array.

2. Uniform Sampling: Fixed set of 'M' particles

   ↳ Need a lot of particles for accurate representation.

3. Rejection Sampling:

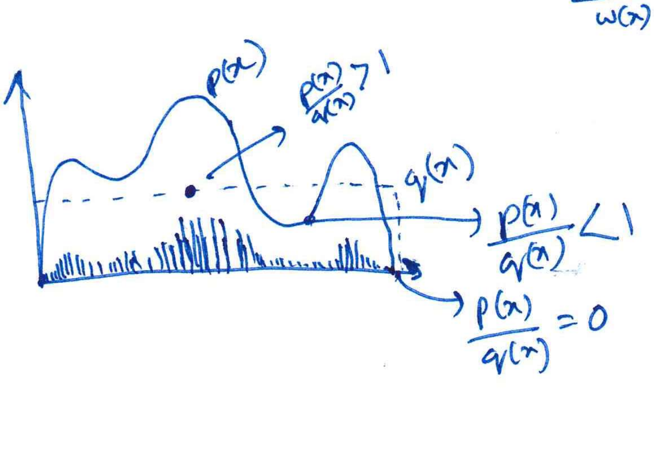


Again, a lot of particles (unwanted)

4. Importance Sampling:

$$\underset{p(x)}{E}(y) = \int y\, p(x)\, dy = \int y \underbrace{\frac{p(x)}{q(x)}}_{w(x) \rightarrow \text{Importance weight}} q(x)\, dy = \int y\, w(x)\, q(x)\, dy$$

$$= \underset{q(x)}{E}[y\, w(x)]$$



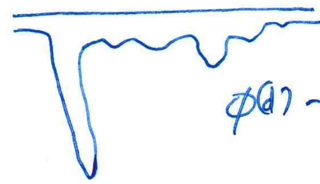
$$w(x) = \frac{p(x)}{q(x)} = \frac{\text{Target}}{\text{Proposal}}$$

Target: Hard to sample form

Proposal: Easier to sample form

To sample, you have to enumerate all possibilities →
so, sometimes even if sampling is hard, evaluating may be easy.

e.g.



$\phi(d) \sim$ depth

$p(z_t|x_t) \rightarrow$ difficult to sample from, easier to evaluate.

Let $g(x_t)$ = proposal distribution.

Let $g(x_t) = \overline{Bel}(x_t) \rightarrow$ Easy to sample from! (Proposal distribution)

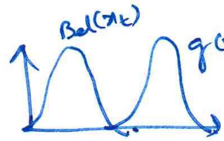
$\therefore \; w(x_t) = \dfrac{Bel(x_t)}{\overline{Bel}(x_t)} = \eta\, p(z_t|x_t) \rightarrow$ Easier to <u>evaluate</u>!

$Bel(x_t) \rightarrow$ Target distribution (difficult to sample)

↳ Importance weight = $\dfrac{\text{Target}}{\text{Proposal}}$

<u>Caveat</u>:

$Bel(x_t) > 0$; $g(x_t) > 0$ Otherwise



→ FAIL

<u>Algorithm</u>:

Proposal distribution → from motion model (sampling is easy)
Target distribution → $Bel(x_t)$

$PF\left(Bel(x_{t-1}), z_t, u_t\right)$

$\approx \left(\{x^1_{t-1}, x^2_{t-1}, \ldots x^M_{t-1}\}, z_t, u_t\right)$

<u>Sequential Importance Sampling</u>

$Bel(x_{t-1}) \Rightarrow \langle x^1_{t-1}, \ldots x^M_{t-1} \rangle$

for i = 1 to M

Sample ← [
$\bar{x}^i_t \sim \overline{Bel}(x_t)$

$\int p(x_t|x_{t-1}, u_t)\, \overline{Bel}(x_{t-1})\, dx_{t-1}$ (only one past x_{t-1} for each particle)

$p(x_t|x^i_{t-1}, u_t) \rightarrow$ sampling directly from motion model (for PF) because its just one particle, and one past x_{t-1}.
]

Importance weighting [
for i = 1 to M
$w^i_t = \eta\, p(z_t|\bar{x}^i_t)\, w^i_{t-1}$
]

Normalizing [
for i = 1 to M
$w^i_t = \dfrac{w^i_t}{\sum_i w^i_t}$
]

return (x^i_t, w^i_t)

Why is sequential importance sampling not good?

Infinite particles → no problem

finite particles → over time, sample with very low probability will keep increasing!

What <u>change</u>? → <u>Res</u>ampling (selecting <u>likely</u> samples)

After <u>Importance weighting</u> and Normalizing step; res<u>ample</u> (sample with replacement)

for $i = 1$ to M

$\bar{x}_t^i \sim$ sample proportional to weights

$w_t^i = 1/M$ (no need double weighting)

⇓

Sequential <u>Importance</u> Resampling Algorithm

So, Importance weighting step ⇒ for $i = 1$ to M
$w_t^i = \eta \, p(z_t | x_t^i)$

So, PF $(Bel(x_{t-1}), z_t, u_t)$

Sample
for $i = 1$ to M
$\bar{x}_t^i \sim \overline{Bel}(x_t)$
$\sim p(x_t | x_{t-1}^i, u_t)$

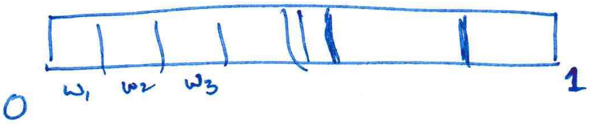
Importance
for $i = 1$ to M
$w_t^i = \eta \, p(z_t | \bar{x}_t^i)$

Normalize
for $i = 1$ to M
$$w_t = \frac{w_t^i}{\sum_i w_t^i}$$

Resample
Resample according to w_t
$\langle x_t^1, x_t^2, \ldots x_t^M \rangle$ (selecting <u>likely</u> samples)

return $\langle x_t^1, x_t^2, \ldots x_t^M \rangle$

So, how to resample?

1. 

cdf ← Naive algorithm

(Random Resampling)

for i = 1 to M
  $r = \text{rand}(0,1)$
  choose particle whose bin 'r' falls into

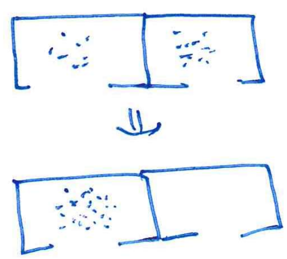
Variability in samples, ✗ but since iid, no control over total ensemble variance ✗

PF with Random resampling

Advantages

1. Parallelizable
2. Any motion/sensor model (sample/evaluate) or vice-versa - dual PF

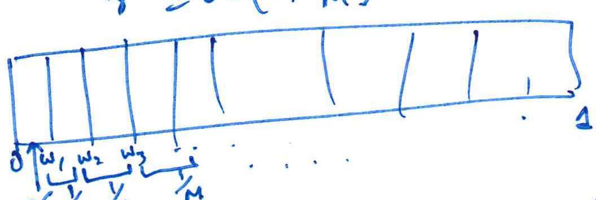
Disadvantages

1. 

2. low-variance resampler (stochastic)

1 random sample 'r' (not all)

→ $r = U(0, \frac{1}{M})$



→ Then, increment it by '$\frac{1}{M}$'.

So, not iid.

→ guaranteed to sample a particle with weight $\geq \frac{1}{M}$.

→ can show ensemble variance is lower than naive random sampling.

P.T.O

Some Issues with PF

1) lack of Diversity / Particle starvation

Random sampling

Solutions :- a) low-variance resampler ✓

b) Add random particles (when to add?)

c) if $\dfrac{\min \{w^i\}}{\max \{w^i\}} > \tau$ (threshold)

↳ resample!

2) Perfect (really good) sensor model causes PF to fail!

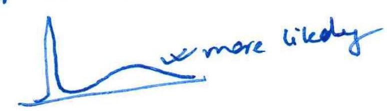
⇒ extreme case example

$p(z|x) = \begin{cases} 1 & \text{if at } \hat{x} \\ 0 & \text{otherwise} \end{cases}$

So, most particles weightless → lost track!

e.g → bumper sensor (0 or 1 contact)

or LIDAR :-

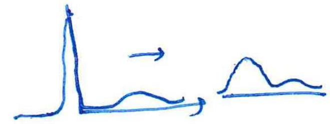


Solutions :-

a) sample directly from sensor model (but hard!)
or make "proposal" closer to sensor model

add more particles

b) Make sensor model noisy
i → gaussian around it
ii → $p(z_t|x_t) = q \rightarrow q^{1/p}$ (squash it)
$0 < p < 1$



P.T.O

GLOBAL LOCALIZATION (Active research area)

You (Robot) don't know where you start (can't use GPS indoor)

Solution :-

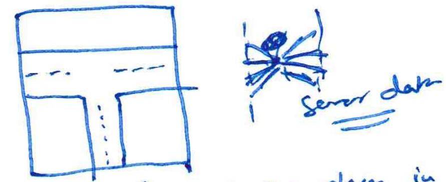
a) Make an inverse sensor model $p(x|z)$ → sample from (Belief – difficult)

b) Uniform discretization (sampling)

for $i = 1$ to M

$w^i = p(z|x^i)$

choose max i

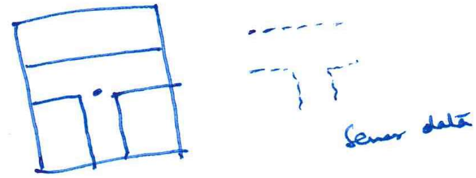


(any of the places in the three corridors)

or, ↳

for $i = 1$ to M_s $(M_s < M)$

Choose $x^i \propto$ location's probability



RELOCALIZATION (KIDNAPPED ROBOT) PROBLEM (Active research area)

You don't know that you don't know (you have been kidnapped)

↳ wrong belief

1) How to know? → don't care
2) Solution? → Add random particles } Naive

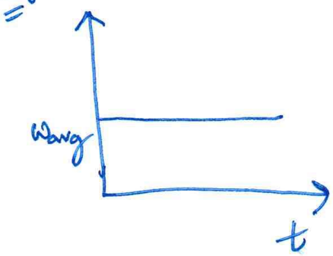
a) Add random particles at each step {Naive} → Disadvantage → Two floors example. → Two hallways example.

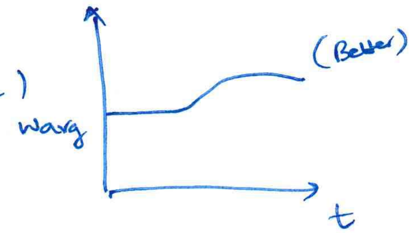
b) If for all particles $P(z_t|x_t)$ have low observation probability, then I am lost. → Key Idea

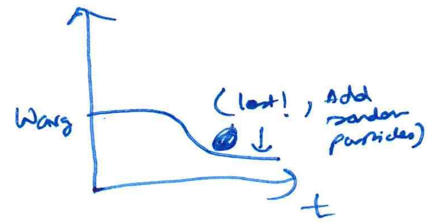
So, $w_{avg} = \frac{1}{M} \sum_{i=1}^{M} P(z_t|x_t^i)$

P.T.O

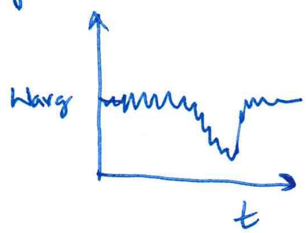
e.g.



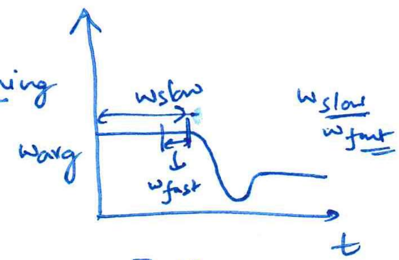




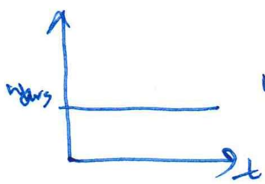
In practice:-



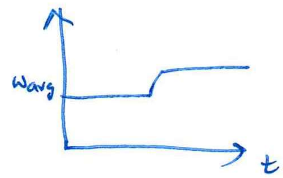
(Noise) → Need smoothing



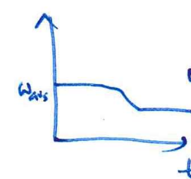
Ratio = $\left[\dfrac{w_{fast}}{w_{slow}} \right.$



$w_{fast} \approx w_{slow}$ (OK)



$w_{fast} > w_{slow}$ (good)



(bad) $w_{fast} < w_{slow}$ ⇓ Add random particles

Algorithm :

Repeat

→ Compute w_i for all particles 1 to M

→ $w_{avg} = \frac{1}{M} \sum_{i=1}^{M} w^i$

→ $w_{fast} = (1-\alpha_1) w_{fast} + \alpha_1 w_{avg}$

→ $w_{slow} = (1-\alpha_2) w_{slow} + \alpha_2 w_{avg}$ $\quad 0 < \alpha_2 \ll \alpha_1 < 1$

→ If $\left(\max\left(0, 1 - \left(\frac{w_{fast}}{w_{slow}}\right)\right)\right) > 0$

add random particles or else nothing.

(*) Another variation:- Adaptively sampling particle set.

(Random particle Monte-Carlo localization)

Random Particle MCL. xxx