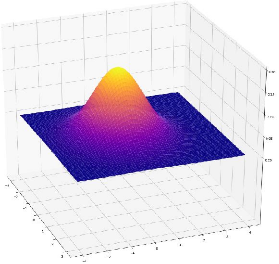
Introduction to Robotics

Week-2

Prof. Balaraman Ravindran

IIT-Madras

- Gaussian Filters are a family of recursive state estimators for continuous spaces, that all share the idea that beliefs are represented by multivariate normal distributions.



$$p(x; \mu, \Sigma) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

- This choice to represent beliefs by a Gaussian distribution creates a model where there is a small margin of uncertainty around a single true state. Gaussian distributions are a poor choice for estimation problems where many distinct hypothesis exist.

The Kalman Filter

Besides the Markov assumption, the belief distribution estimated by the Kalman algorithm will be a Gaussian distribution if the following three properties hold:

1. The system must follow linear dynamics, with randomness in state transition modelled as Gaussian noise.

   $x_t = A_t x_{t-1} + B_t u_t + \epsilon_t.$

   Here, $\epsilon_t \sim \mathcal{N}(0, R_t)$

   The state transition probability $p(x_t|u_t,\ x_{t-1})$ is thus given by:

$$\frac{1}{(2\pi)^{n/2}|R_t|^{1/2}} exp(-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t))$$

   i.e $\mathcal{N}(A_t x_{t-1} + B_t u_t,\ R_t)$

2. The measurement probability must also be linear in it's arguments, with added Gaussian noise:

   $z_t = C_t x_t + \delta_t$

   Here, $\delta_t \sim \mathcal{N}(0, Q_t)$

   The measurement probability $p(z_t|x_t)$ is thus given by:

   $\frac{1}{(2\pi)^{n/2}|Q_t|^{1/2}} exp(-\frac{1}{2}(x_t - C_t x_t)^T Q_t^{-1}(x_t - C_t x_t))$

   i.e $\mathcal{N}(C_t x_t,\ Q_t)$

3. The initial belief $bel(x_0)$ must be normal distributed with some mean μ_0 and covariance Σ_0. i.e $\mathcal{N}(\mu_0,\ \Sigma_0)$

   $bel(x_0) = \frac{1}{(2\pi)^{n/2}|\Sigma_0|^{1/2}} exp(-\frac{1}{2}(x_t - \mu_0)^T \Sigma_0^{-1}(x_t - \mu_0)$

Gaussian Filters

- Similar to the **Bayes Filter** algorithm, the **Kalman Filter** algorithm recursively estimates $bel(x_t)$ from $\{bel(x_{t-1}),\ u_t,\ z_t\}$
- $bel(x_t)$ is parameterized by $\mu_t,\ \Sigma_t$

Algorithm Kalman_filter($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)**:**

$\bar{\mu}_t = A_t\ \mu_{t-1} + B_t\ u_t$

$\bar{\Sigma}_t = A_t\ \Sigma_{t-1}\ A_t^T + R_t$

$K_t = \bar{\Sigma}_t\ C_t^T (C_t\ \bar{\Sigma}_t\ C_t^T + Q_t)^{-1}$

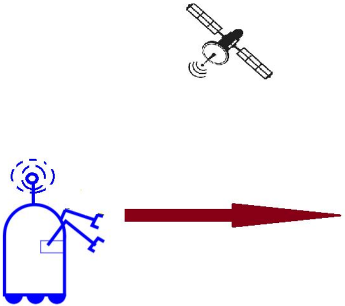
$\mu_t = \bar{\mu}_t + K_t(z_t - C_t\ \bar{\mu}_t)$

$\Sigma_t = (I - K_t\ C_t)\ \bar{\Sigma}_t$

return μ_t, Σ_t

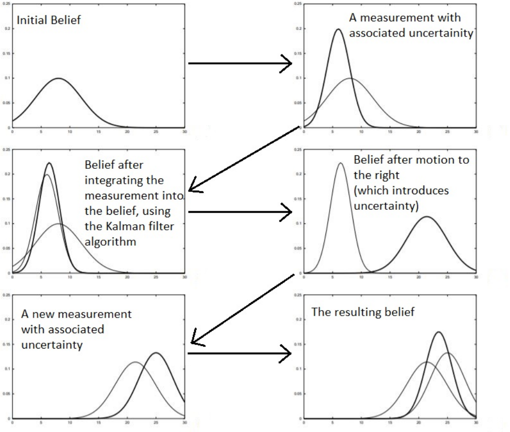
- **Note:** Similar to the Bayes filter, we have a **prediction** step *(Lines 2, 3)* - which modifies belief in accordance to an action, and a **measurement update** step *(Lines 4,5,6)* - in which sensor data is integrated into the present belief.
- The variable $\boldsymbol{K_t}$, computed in Line 4 is called *Kalman gain*. It specifies the degree to which the measurement is incorporated into the new state estimate.

A simplistic one-dimensional localization scenario



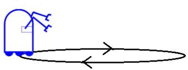
- The robot moves along the horizontal axis.
- The robot queries it's GPS sensors on it's location

An Illustration of the Kalman Filter Algorithm



The Extended Kalman Filter

- The assumptions of linear state transitions and linear measurements with added Gaussian noise are rarely satisfied in practice. For example, a robot moving in a circular trajectory with constant velocity cannot be described by linear next state transitions.



- The extended Kalman filter (EKF) overcomes the linearity assumption, assuming instead that the next state probability and the measurement probabilities are governed by nonlinear functions g and h.

$$x_t = g(u_t,\ x_t) + \epsilon_t$$

$$z_t = h(x_t) + \delta_t$$

- Unfortunately, with arbitrary functions g and h, we are no longer guaranteed that the belief distribution estimated by the Kalman Filter algorithm is a Gaussian.
- The extended Kalman filter (EKF) overcomes this problem by calculating a linear approximation for the functions g and h, to approximate the true belief.
- Once g and h are linearized, the mechanics of belief propagation are equivalent to those of the Kalman filter

Taylor expansion:

Slope is given by the partial derivative:

$$g^{'}(u_t, x_{t-1}) := \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}}$$

and we approximate:

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + g^{'}(u_t, \mu_{t-1})(x_{t-1} - \mu_{t-1})$$

Defining, $G_t := g^{'}(u_t, x_{t-1})$ (Jacobian - matrix of size $n \times n$)

We have,

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1})$$

So that next state probability $p(x_t | u_t,\ x_{t-1})$ is approximated to:

$$\mathcal{N}(g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1}),\ R_t)$$

Similarly,

We approximate:

$$\begin{aligned} h(x_t) &\approx h(\overline{\mu_t}) + h^{'}(\overline{\mu_t})(x_t - \overline{\mu_t}) \\ &= h(\overline{\mu_t}) + H_t(x_t - \overline{\mu_t}) \end{aligned}$$

(Where $\overline{\mu_t}$ is the state deemed most likely at the time of linearizing h)

So that measurement probability is approximated to:

$$\mathcal{N}(h(\overline{\mu_t}) + H_t(x_t - \overline{\mu_t}),\ Q_t)$$

- The linear predictions in Kalman filters are replaced by their nonlinear generalizations in EKFs. Moreover, EKFs use Jacobians G_t and H_t instead of the corresponding linear system matrices A_t, B_t, and C_t in Kalman filters.

Algorithm Extended_Kalman_filter($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)**:**

$\quad \bar{\mu}_t = g(u_t, \mu_{t-1})$

$\quad \bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^T + R_t$

$\quad K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$

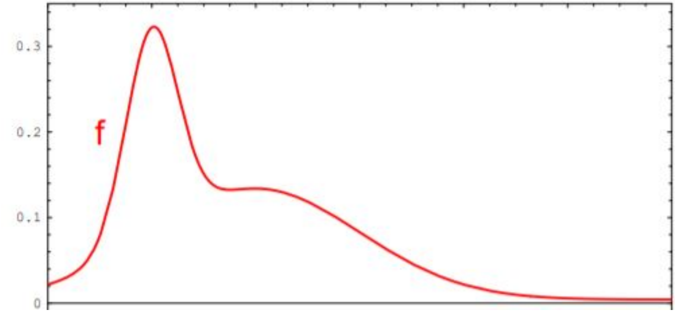
$\quad \mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$

$\quad \Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$

$\quad$ return μ_t, Σ_t

Non-Parametric Filters

- Alternative to Gaussian Filters, non-parametric filters do not assume a fixed functional form over the belief distribution, allowing us to model more complex, non-gaussian probability distributions.



- They estimate the belief distribution at a fixed number of positions in the state space.

The Particle Filter

- Instead of representing the belief distribution by a parametric form, particle filters represent this distribution by a set of samples drawn from the distribution.
- The samples drawn from the distribution are called particles, and are denoted by:

  $X_t := x_t^{[1]}, x_t^{[2]}, ..., x_t^{[M]}$

  Each particle is a hypothesis as to what the true world state may be at time t.

  $x_t^{[m]} \sim p(x_t |\ z_{1:t},\ u_{1:t})$

  As a consequence, as $M \to \infty$, the denser a sub-region of the state space is populated by samples, the more likely it is that the true state falls into this region.

Non-Parametric Filters

- As with the previous algorithms, the particle filter algorithm estimates the belief $bel(x_t)$ recursively from the belief $bel(x_{t-1})$.

Algorithm Particle_filter($\mathcal{X}_{t-1}, u_t, z_t$):

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$

for $m = 1$ to M do

  sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$

  $w_t^{[m]} = p(z_t \mid x_t^{[m]})$

  $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$

endfor

for $m = 1$ to M do

  draw i with probability $\propto w_t^{[i]}$

  add $x_t^{[i]}$ to $\mathcal{X}_t$

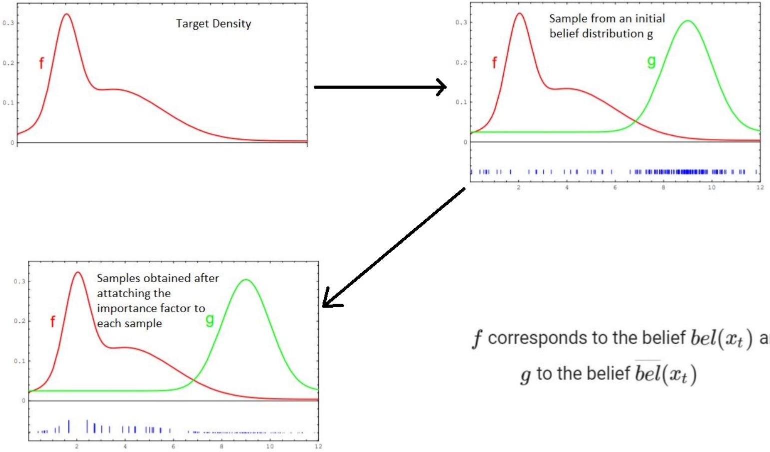
endfor

return $\mathcal{X}_t$

- The set of particles resulting from iterating Step 4 M times is the filter's representation of $\overline{bel}(x_t)$
- Line 5 calculates for each particle $x_t^{[m]}$ the so-called importance factor, denoted by $w_t^{[m]}$. Importance factors are used to incorporate the measurement z_t into the particle set.

- lines 8-11 implement what is known as *resampling* or importance resampling.
  - The algorithm draws with replacement M particles from the temporary set $\overline{X_t}$. The probability of drawing each particle is given by its importance weight.
  - Resampling transforms a particle set of M particles into another particle set of the same size.
  - By incorporating the importance weights into the resampling process, the distribution of the particles change: whereas before the resampling step, they were distributed according to $\overline{bel}(x_t)$, after the resampling they are distributed (approximately) according to the posterior $bel(x_t) = \eta p(z_t|x_t^{[m]})\overline{bel}(x_t)$

An Illustration of the Particle Filter Algorithm



f corresponds to the belief $bel(x_t)$ and
g to the belief $\overline{bel}(x_t)$

The Binary Bayes Filter (with static state)

- Certain problems are best formulated as binary state problems (i.e with states x and $\neg x$), where the robot needs to estimate a static state from a sequence of sensor measurements.
- Since the state is static, the belief is a function of the measurements

  $$bel_t(x) = p(x|z_{1:t}, u_{1:t}) = p(x|z_{1:t})$$

  Note, in such problems:

  $$bel_t(\neg x) = 1 - bel_t(x)$$

  The lack of time index indicates static state.

Non-Parametric Filters

- The belief in such problems is commonly implemented as a log odds ratio.

  Where, odds of an event x is defined as: $\frac{p(x)}{p(\neg x)} = \frac{p(x)}{1-p(x)}$

  log odds of this expression is:

  $l(x) := \log \frac{p(x)}{1-p(x)}$

- The Bayes filter for updating beliefs in log odds representation is computationally elegant.

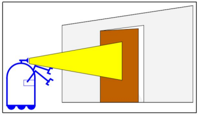
```
Algorithm binary_Bayes_filter(l_{t-1}, z_t):
    l_t = l_{t-1} + log p(x|z_t)/(1-p(x|z_t)) - log p(x)/(1-p(x))
    return l_t
```

Algorithm binary_Bayes_filter(l_{t-1}, z_t):

$$l_t = l_{t-1} + \log \frac{p(x|z_t)}{1-p(x|z_t)} - \log \frac{p(x)}{1-p(x)}$$

return l_t

- This binary Bayes filter uses an inverse measurement model $p(x|z_t)$, instead of the familiar forward model $p(z_| x)$.
- Inverse models are often used in situations where measurements are more complex than the binary state.
- For example, it is easier to devise a function that calculates a probability of a door being closed from a camera image, than describing the distribution over all camera images that show a closed door. In other words, it is easier to implement an inverse than a forward sensor.



- Here the state is extremely simple, but the space of all measurements is huge.

The belief can be recovered from the log odds ratio by the following equation:

$$bel_t(x) = 1 - \frac{1}{1+\exp(l_t(x))}$$

Correctness of the Binary Bayes Filter Algorithm

$$bel(x) = p(x|z_{1:t},\ u_{1:t}) = p(x|z_{1:t})$$

Using Bayes rule $\{p(A|B) = \frac{p(B|A)p(A)}{p(B)}\}$,

$$= \frac{p(z_t|x, z_{1:t-1})p(x|z_{1:t-1})}{p(z_t|z_{1:t-1})}$$

$$= \frac{p(z_t|x)p(x|z_{1:t-1})}{p(z_t|z_{1:t-1})}$$

Applying Bayes rule to the measurement model,

$$p(z_t|x) = \frac{p(x|z_t)p(z_t)}{p(x)}$$

Substituting,

$$p(x|z_{1:t}) = \frac{p(x|z_t)p(z_t)p(x|z_{1:t-1})}{p(x)p(z_t|z_{1:t-1})}$$

Similarly,

$$p(\neg x|z_{1:t}) = \frac{p(\neg x|z_t)p(z_t)p(\neg x|z_{1:t-1})}{p(\neg x)p(z_t|z_{1:t-1})}$$

Consequently,

$$\textbf{odds} = \frac{p(x|z_{1:t})}{p(\neg x|z_{1:t})} = \frac{p(x|z_t)}{p(\neg x|z_t)} \frac{p(x|z_{1:t-1})}{p(\neg x|z_{1:t-1})} \frac{p(\neg x)}{p(x)}$$

$$= \frac{p(x|z_t)}{1-p(x|z_t)} \frac{p(x|z_{1:t-1})}{1-p(x|z_{1:t-1})} \frac{1-p(x)}{p(x)}$$

So, $l_t(x) = \log \frac{p(x|z_t)}{1-p(x|z_t)} + \log \frac{p(x|z_{1:t-1})}{1-p(x|z_{1:t-1})} + \log \frac{1-p(x)}{p(x)}$

$$= \log \frac{p(x|z_t)}{1-p(x|z_t)} - \log \frac{p(x)}{1-p(x)} + l_{t-1}(x)$$