EKF SLAM

Definition of the slam Problem

Given:

- Robots control $< u_1, u_2 \dots U_T >$
- observations $< z_1 \dots \dots z_T >$

Wanted:

- Map of the environment (m)
- Path of the robot $< x_0, x_1 \dots x_T >$

Bayes Filter (Recursive)

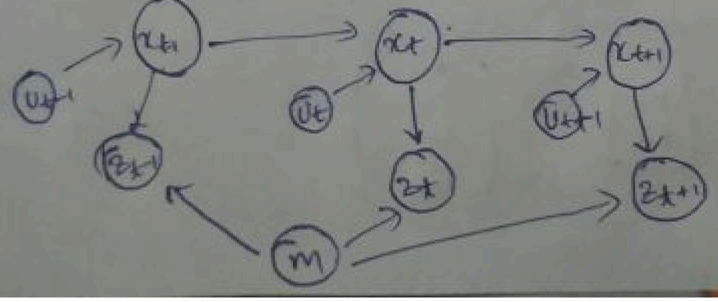
$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx_{t-1}$$

$$bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$$

EKF for Online (complete SLAM)

we consider here the Kalman filter as a solution to online slam problem

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



Extended_Karmal_Filter_algorithm ($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

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

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

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

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

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

return μ_t, Σ_t.

EKF SLAM

- It is the application of EKF to SLAM
- Helps estimate robot's pose & location of landmark in the environment.
- Assumption: Knows correspondence
- State space (for 2D plane) is

$$x_t = (\underbrace{x, y, \theta}_{\text{robots pose}}, \underbrace{m_{(1,x)}, m_{(1,y)}}_{\text{landmark 1}}, \ldots \underbrace{m_{(n,x)}, m_{(n,y)}}_{\text{landmark n}})^T$$

- The map is large state vector staking robot and landmark series

$$x = \begin{bmatrix} R \\ M \end{bmatrix} = \begin{pmatrix} R \\ L_1 \\ L_2 \\ L_n \end{pmatrix}$$

⊙ we assume Map with n- landmarks
(which will be $(3+2n)$ -D Gaussian)

⊙ Belief is given as

$$
\begin{bmatrix} x \\ y \\ \theta \\ m_{1,x} \\ m_{1,y} \\ \vdots \\ m_{n,x} \\ m_{n,y} \end{bmatrix}
\begin{bmatrix}
\sigma_{xx} & \sigma_{xy} & \sigma_{x\theta} & \sigma_{x m_{(1,x)}} & \sigma_{x m_{(1,y)}} & \cdots & \sigma_{x m_{(n,x)}} & \sigma_{x m_{(n,y)}} \\
\vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots \\
\sigma_{m_{1,x} x} & \sigma_{m_{1,x} y} & \sigma_{\theta} & \sigma_{m_{(1,x)} m_{(1,x)}} & \cdots & & & \vdots \\
\sigma_{m_{1,y} x} & \sigma_{m_{1,y} y} & \sigma_{\theta} & \vdots & & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots
\end{bmatrix}
$$

can be compactly written as :-

$$
\underbrace{\begin{pmatrix} x \\ m \end{pmatrix}}_{\mu}
\underbrace{\begin{pmatrix} \Sigma_{xx} & \Sigma_{xm} \\ \Sigma_{mx} & \Sigma_{mm} \end{pmatrix}}_{\Sigma}
$$

EKF SLAM filter cycle

1. State Prediction
2. Measurement prediction
3. Measurement (Actual)
4. Data association
5. Update

In EKF: map is modeled by Gaussian variable using mean and co-variance matrix of state vector
$(\bar{x})$ (P)

$$\bar{x} = \begin{bmatrix} \bar{R} \\ \bar{M} \end{bmatrix} = \begin{bmatrix} \bar{R} \\ \bar{L}_1 \\ \vdots \\ \bar{L}_n \end{bmatrix} \quad -(1)$$

$$P = \begin{bmatrix} P_{RR} & P_{RM} \\ P_{MR} & P_{MM} \end{bmatrix} = \begin{bmatrix} P_{RR} & P_{RL_1} & \cdots & P_{RL_n} \\ P_{L_1R} & P_{L_1L_1} & \cdots & P_{L_1L_n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{L_nR} & P_{L_nL_1} & \cdots & P_{L_nL_n} \end{bmatrix} \quad -(2)$$

The goal is to keep the map $\{\bar{x}, P\}$ up to date always

Operations of EKF-SLAM

1) Map initialization: map start with no landmarks also initial pose is considered origin.
therefore

$$x = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad P = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (3)$$

2) Robot motion
in regular EKF, x is state vector, u is control vector & n is pertubation vector, then generic ~~time~~ time-update function is

$$x \leftarrow f(x, u, n) \quad (4)$$

EKF Prediction step is

$$\bar{x} \leftarrow f(\hat{x}, u, 0) \quad — \quad (5)$$

$$P \leftarrow F_x P F_x^T + F_n N F_n^T \quad — \quad (6)$$

where $F_x = \frac{\partial f(\hat{x}, u)}{\partial x}$, $F_n = \frac{\partial f(\hat{x}, u)}{\partial n}$

N is Covariance matrix of perturbation n

~~for Slam~~

In SLAM, only part of state is time variant so most of map matrix is invariant of robot motion

$$R \leftarrow f_R(R, u, n) \quad — \quad (7)$$

$$M \leftarrow M \quad - \quad (8)$$

As ~~most~~ the first eqn is motion model & Sparse Jacobian Marix is

$$F_x = \begin{bmatrix} \frac{\partial f_R}{\partial R} & 0 \\ 0 & I \end{bmatrix} \qquad F_n = \begin{bmatrix} \frac{\partial R}{\partial n} \\ 0 \end{bmatrix} \quad — \quad (9)$$

if we avoid all trivial operations in eqn (6) we get

$$\bar{R} \leftarrow f_R(\hat{R}, u, 0) \quad - \quad (10)$$

$$\boxed{P_{RR} \leftarrow \frac{\partial f_R}{\partial R} P_{RR} \frac{\partial f_R}{\partial R}^T + \frac{\partial f_R}{\partial n} N \frac{\partial f_R}{\partial n}^T} \quad - \quad (11)$$

$$\boxed{P_{RM} \leftarrow \frac{\partial f_R}{\partial R} P_{RM}} \quad - \quad (12)$$

$$\boxed{P_{MR} \leftarrow P_{RM}^T} \quad - \quad (13)$$

More ~~of~~ over if we store Covariance matrix as upper triangular matrix then operation (13) is not required.

The algorithmic complexity of (12) is $O(n)$

(3) Observation of mapped landmark

generic EKF observation function

$$y = h(x) + v \quad \text{— (14)}$$

y = noisy measurement
x = full state
$h()$ = observation function
v = measurement noise

EKF correction steps

$$\bar{z} = y - h(\bar{x}) \quad \text{— (15)}$$

$$Z = H_x P H_x^T + R \quad \text{— (16)}$$

$$K = P H_x^T Z^{-1} \quad \text{— (17)}$$

$$\bar{x} \leftarrow \bar{x} + K\bar{z} \quad \text{— (18)}$$

$$P \leftarrow P - K Z K^T \quad \text{— (19)}$$

where $H_x = \frac{\partial h(x)}{\partial x}$, R is cov matrix of measured Noise

$\bar{z}, Z$ are innovations mean & Cov matrix
K = Kalman gain
$\bar{x}, P$ = filter updates

The landmark observation in EKF only depends on robot State R, sensor state S, & Particular Landmark Li Assuming landmark Li is observed.

$$y_i = h_i(R, S, L_i) + v \quad - (20)$$

which only depend on landmark Li & no other therefore Hx can be written as

$$H_x = [H_R, 0 \ldots\ldots H_L \ldots 0] \quad - (21)$$

with $H_R = \dfrac{\partial h_i(\bar{R}\, S\, \bar{L_i})}{\partial R}$

$$H_L = \frac{\partial h_i(\bar{R}\, S\, \bar{L_i})}{\partial L_i}$$

So, our set of eq^n becomes

$$\bar{Z} = y_i - h_i(\bar{R}\, S\, \bar{L_i}) \quad - (22)$$

$$Z = [H_R\ H_L] \begin{bmatrix} P_{RR} & P_{RL_i} \\ P_{L_iR} & P_{L_iL_i} \end{bmatrix} \begin{bmatrix} H_R^T \\ H_{L_i} \end{bmatrix} + R \quad - (23)$$

$$K = \begin{bmatrix} P_{RR} & P_{RL_i} \\ P_{MR} & P_{ML_i} \end{bmatrix} \begin{bmatrix} H_R^T \\ H_{L_i}^T \end{bmatrix} Z^{-1} \quad - (24)$$

$$\bar{x} \leftarrow x + K\bar{z} \quad - (25)$$

$$P \leftarrow P - KZK^T \quad - (26)$$

the complexity is of $O(n^2)$ due to eq^n

4- Landmark initialization For full observation

this is done when robot discovers Landmark that are not yet mapped. and decides to incorporate them in map.

this results in increase of State vector size. & then EKF becomes filter of state of dynamic size.

Landmark initialization is simple when sensor provide information about all degree of freedom. of new landmark when this happens we only need to ~~update~~ invert $h()$ (observation function) to compute new landmarks state L_{n+1} from robot state (R) sensor state (S) & observation (y_{n+1})

$$L_{n+1} = g(R, S, y_{n+1})$$

we proceed as,

$$\bar{L}_{n+1} = g(\bar{R}, S, y_{n+1})$$

$$G_R = \frac{\partial g(\bar{R}, S, y_{n+1})}{\partial R}$$

$$G_{y_{n+1}} = \frac{\partial g(\bar{R}, S, y_{n+1})}{\partial y_{n+1}}$$

Then calculate covariance P_{LL}, & cross-variance with rest of map P_{Lx}

$$P_{LL} \rightarrow G_R P_{RR} G_R^T + G_{y_{n+1}} R\, G^T_{y_{n+1}}$$

$$P_{Lx} = G_R P_{Rx} = G_R \left[P_{RR} \; P_{RM} \right]$$

Finally append this results to state mean & covariance matrix.

$$\bar{x} \leftarrow \begin{bmatrix} \bar{x} \\ \bar{l}_{n+1} \end{bmatrix}$$

$$P \leftarrow \begin{bmatrix} P & P_{Lx}^{T} \\ P_{Lx} & P_{LL} \end{bmatrix}$$

```
% INITIALIZATION
initialize map()
time = 0
% TIME LOOP
while (execution() == true) do:
    % LOOP ROBOTS
    for each robot in list of robots
        control = acquire control signal()
        move robot(robot, control)
        % LOOP SENSORS IN EACH ROBOT
        for each sensor in robot->list of sensors
            raw = sensor->acquire raw data()
            % LOOP OBSERVATIONS IN EACH SENSOR
            for each observation in sensor->feasible observations()
                % MEASURE LANDMARK AND CORRECT MAP
                measurement = find known feature(raw, observation)
                update map(robot, sensor, landmark, observation, measurement)
            end
            % DISCOVER NEW LANDMARKS WITH THE CURRENT SENSOR
            measurement = detect new feature(raw)
            % INITIALIZE LANDMARK
            landmark = init new landmark(robot, sensor, measurement)
            create new observation(sensor, landmark)
        end
    end
    time ++
end
```

FAST SLAM

FAST SLAM is Particle Filter

Particle filter are non-Parametric recursive Bayes filter. Posterior is represented by set of weighted Sample. These can model arbitrary distribution. FAST SLAM works in low dimensional spaces

3-Step Procedure

- Sampling from Proposal
- Importance weighting
- Resampling

Particle filter Algorithm

1. Sample particles from Proposal distribution

$$x_t^{(j)} \sim \pi(x_t \mid \ldots)$$

2. Compute the importance weights

$$w_t^{(j)} = \frac{\text{target } x_t^{(j)}}{\text{Proposal } x_t^{(j)}}$$

3. Resampling : draw Sample i with probability $w_t^{(i)}$ & repeat J times

Particle Representation

- Set of wt. samples $x = \{\langle x_i, w_i \rangle\}_{i=1,N}$
- Think of a sample as one hypothesis about state
- For feature based SLAM

$$x = \{x_{1:t}, m_{1,x}, m_{1,y} \ldots m_{M,x}, m_{M,y})^T$$

If we know the Poses of the Robot, Mapping is easy.

$x_{1:t}, m_1 \ldots \ldots m_M$

If we use the particle set only to model the robot's path, each sample is a path hypothesis for each sample we can compute individual map of landmarks.

Rao- Blackwellization

- Factorization to exploit dependencies between variables

$$P(a,b) = p(b|a)\,p(a)$$

- If $p(b|a)$ can be computed efficiently represented only $p(a)$ with samples.

- Factorization of simple SLAM ~~Problem~~ Posterior

$$P(\underbrace{x_{0:t}}_{\text{Poses}}, \underbrace{m_{1:M}}_{\text{map}} \mid \underbrace{z_{1:t}}_{\text{observation}}\ \underbrace{u_{1:t}}_{\text{movements}})$$

$$= \underbrace{P(x_{0:t} \mid z_{1:t}, u_{1:t})}_{\text{Path posterior}}\ \underbrace{P(m_{1:M} \mid x_{0:t}, z_{1:t})}_{\text{map posterior}}$$

How to compute this term

$$= \underbrace{P(x_{0:t} \mid z_{1:t}, u_{1:t})}_{\text{Particle filter}} \underbrace{\prod^{M} P(m_i \mid x_{0:t}, z_{1:t})}_{\text{2D-EKFs}}$$

Modelling of Robot Path

→ Sample based representation for

$$P(x_{0:t} \mid z_{1:t}, u_{1:t})$$

→ Each Sample is path hypothesis

x_0 x_1 x_2 - - .

Starting location (0, 0)

Pose hypothesis of time t = 1

→ Past poses of Sample are not revised

→ No need to maintain past poses in sample set

Key Steps of FastSLAM

→ Extend the path posterior by sampling a new pose for each sample. $x^k \sim P(x_t \mid x_{t-1}^k, u_t)$

→ Compute Particle wt.

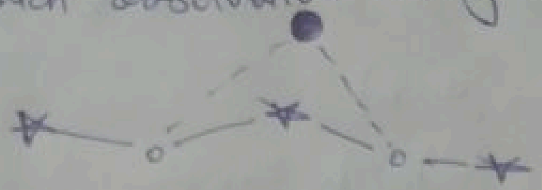
$$\omega^{(k)} = |2\pi Q|^{-1/2} \exp\left(-\frac{1}{2}(z_t - \hat{z}^{(k)})^T Q^{-1} (z_t - \hat{z}^{(k)}\right.$$

→ Update belief of observed landmark

→ resample

Data Assosiation Problem

↳ which observation belong to which landmark



- More than one possible association
- Potential data association depends on pose of robot.

Particle support for Multi hypothesis

- Decision on a per-particle basis
- Robot pose error is factored out of data association decisions.

→ two options for per-particle data association
1) choose most probable match
2) Pick a random association wt. by the observation of likelyhood.

→ if Probablity of an assignment is low generate new landmark