Robotic SLAM

1 Introduction

Terms

- State estimation - find out the pose
- Localisation - pose w.r.to landmark or map
- Mapping
- navigation and motion planning - a star, wave front dijkstra

1.1 What is SLAM

Computing robot's poses and the map of the environment at the same time.
Localisation : estimating robots location
Mapping : building a MAP

Given

- Robots control inputs

$$u_{1:T} = \{u_1, u_2, u_3....u_T\}$$

- Observations

$$z_{1:T} = \{z_1, z_2, z_3, ..., z_T\}$$

Wanted

- Map of the environment

$$m$$

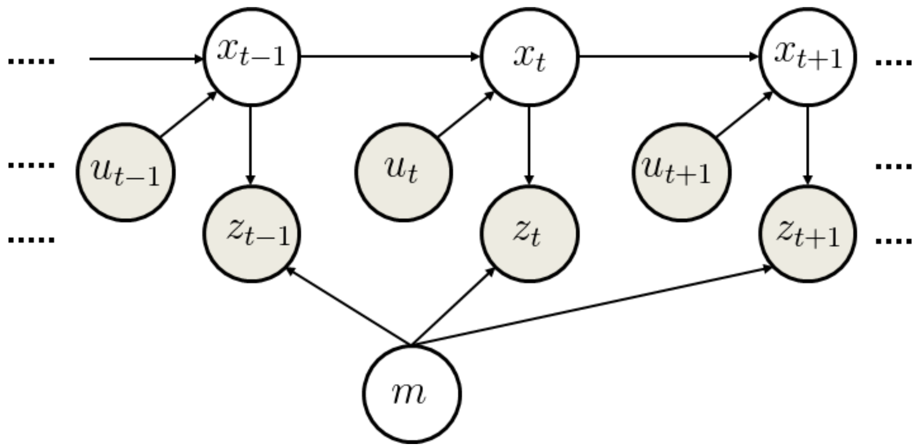
- path of the Robot

$$x_{0:T} = \{x_0, x_1, x_2, ..., x_T\}$$

Using the robots control inputs we can predict the position of the robot. From the observations $z_{1:T}$, we can calculate the position of the robot. Both the steps have some error associcated with it . Lets call the first one the model noise and second one the sensor noise. So we have to associate a probability with both of them. The error accumulates over time(even if the error in individual measurements is really small)

So in the probalistic terms our problem minimises to

$$p(x_{0:T}, m|z_{1:T}, u_{1:T})$$



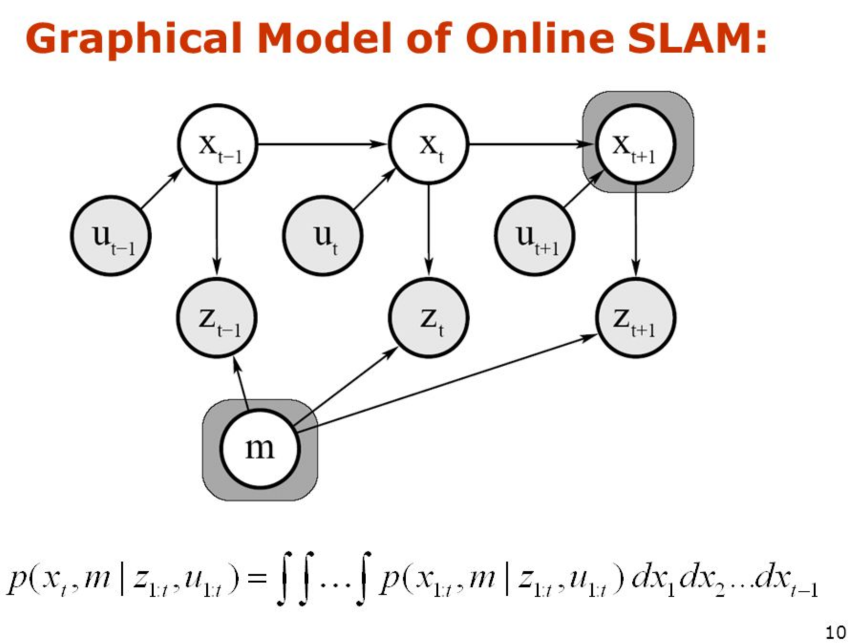
1.2 Full Slam vs online SLAM

- Full SLAM estimates the entire path

$$p(x_{0:T}, m | z_{1:T}, u_{1:T})$$

- Online SLAM estimates only the most recent pose

$$p(x_t, m | z_{1:T}, u_{1:T})$$



1.3 Types of SLAM

occupancy maps created from lidars, sonars etc. - volumetric SLAM feature based approach - store features and localise based on that volumetric SLAM maybe better for navigation applications . Topological representations vs geometric representations. Static vs dynamic features. Active - robot decides the path so as to build a map vs passive slam - may follow a fixed path i.e. path not optimised for mapping/ exploration

2 Bayes Filter

2.1 State Estimation

Goal $p(x|z,u)$
Recursive Bayes Filter

$$\begin{aligned}
bel(x_t) &= p(x_t|z_{1:t}, u_{1:t}) \\
&= \eta p(z_t|x_t, z_{1:t-1}, u_{1:t}) * p(x_t|z_{1:t-1}, u1:t) \\
&= \eta p(z_t|x_t) * p(x_t|z_{1:t-1}, u1:t) \\
&= \eta p(z_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t}) * p(x_{t-1}|z_{1:t-1} u_{1:t-1}) dx_{t-1} \\
&= \eta p(z_t|x_t) \int_{x_{t-1}} p(x_t|x_{t-1}, u_t) * bel(x_{t-1}) dx_{t-1}
\end{aligned}$$

we can split this into predict and update steps where
Predict Step

$$\overline{bel(x_t)} = \int_{x_{t-1}} p(x_t|x_{t-1}, u_t) * bel(x_{t-1}) dx_{t-1}$$

Update Step

$$bel(x_t) = \eta * p(z_t|x_t) * \overline{bel(x_t)}$$

Bayes filter gives a framework for recursive state estimation using the above equations. The actual realisation may be kalman filtering , EKF or particle filter (Linear non linear motion models) (distributions) *Kalman Filter* - Gaussians , requires linear or linearised model *Particle filter* - Non-parametric , Arbitrary models

2.2 Probability motion models

$p(x_t|u_t, x_{t-1})$ we can model this in two ways

- odometry models - measurement of velocity (tends to be more accurate)
- velocity models - we know the input commands, but no measurement of vel

2.3 Model for laser scanners

scan z consisits of k beams $z_t \,\epsilon\, \Re^k$ i.e. $z_t = \{z_t^1, z_t^2, ...z_t^k\}$,
Assuming beams are independednt, then

$$p(z_t|x_t, m) = \prod_{i=1}^{k} p(z_t^i|x_t, m)$$

- Beam endpoint model (likelihood calculated as gaussian blur on occupancy map)
- Ray cast model (occlusion, sensor accuray, satuaration , random)
- model for range bearing sensors $z_t^i = (r_t^i, \phi_t^i)^T$

$$r_t^i = ||m - x|| + gaussian$$

$$\phi_t^i = \angle(m - x) - \theta + gaussian$$

3 Kalman filter Equations

$$\begin{aligned}
\bar{\mu_t} &= g(u_t, \mu_{t-1}) \\
\bar{\Sigma_t} &= G_t \Sigma_{t-1} G_t^T + R_t \\
K_t &= \bar{\Sigma_t} H_t^T (H_t \bar{\Sigma_t} H_t^T + Q_t)^{-1} \\
\mu_t &= \bar{\mu_t} + K_t(z_t - h(\bar{\mu_t})) \\
\Sigma_t &= (I - K_t H_t)\bar{\Sigma_t}
\end{aligned}$$

3.1 Summary

- Diverge for large non linearities
- Can deal only single modes
- Successful in medium scale scenes with good data associations
- Approximations exists to reduce the computational complexity

commonly used Datasets:

- Victoria Park Data sets - Trees are the landmark -Data assosciation -girth and height
- Tennis Court Dataset - for mapping precescion

4 Extended Kalman Filter vs Unscented Kalman Filter

EKF works by linearising the state transfer equations thus making sure that all the conditional and marginal distributions will remain gaussian. Gaussians are closed space. **UKF** uses the nonlinear state transmission equation, then tries to sample a gaussian distribution from the non-gaussian resulted from the non-linear operation.

4.1 Strategy for chosing sampling points and weights for UKF

$$
\begin{aligned}
X^{[0]} &= \mu \\
X^{[i]} &= \mu + (\sqrt{(n+\lambda)\Sigma})_i \; for \; i = 1, ..., n \\
X^{[i]} &= \mu - (\sqrt{(n+\lambda)\Sigma})_i \; for \; i = n+1, ..., 2n \\
w_m^{[0]} &= \frac{\lambda}{n+\lambda} \\
w_c^{[0]} &= w_m^{[0]} + (a - \alpha^2 + \beta) \\
w_m^{[i]} = w_c^{[i]} &= \frac{1}{2(n+\lambda)} \; for \; i = 1, ..., 2n
\end{aligned}
$$

where $\alpha \, \epsilon \, (0, 1]$; $k >= 0$; $\lambda = \alpha^2(n+k) - n$;
too small value of K will lead to UKF EKF . Too large - diverge
$\sqrt{\Sigma} = VD^{1/2}V^{-1}$, we are sampling the gaussian along the eigen vectors of the covariance matrix.

$$\bar{\mu}_t = \sum_{1=0}^{2n} w^{[i]} g(X^{[i]})$$

$$\bar{\Sigma}_t = \sum_{1=0}^{2n} w^{[i]} (g(X^{[i]}) - \bar{\mu}_t)(g(X^{[i]}) - \bar{\mu}_t)^T + R_t$$

5 Grid Maps

SLAM problems can make one of the types of maps

- **Feature Based Maps:** Maps is repsented by a few lanndmarks. Advantage of this type of maps is the space time complexity can be smaller compared to Volumetric Maps. On the downside, we have to implement a feature detector - i.e. observations are not directly used in this method.
- **Volumetric Maps:** Volumetric Maps on the other hand uses all observations and represent the map using a 3D(lidar point cloud) or 2D(range

Figure 1: Feature map overlayed on victoria park

(a) 3D grid map from lidar point cloud

(b) Occupancy Grid Map

Figure 2: Volumetric Maps

sensor) grid. Occupancy is often shown with a black pixel and free areas are white. Unseen areas are marked with grey. Feature based maps are used in kalman filter implementations, Particle filter based approaches use Grid mapping and sometimes feature based maps.

5.1 Representation - Occupancy Grid Map

The **Occupancy grid map** partitions the space into finitely many grid cells. $m = \{m_i\} \ni$ every m_i forms a binary random variable . Size of grid cell is a free parameter which decides the trade of between computational complexity and approximation Error.

- $p(m_i) \to 1 \Rightarrow$ high confidence that grid cell is occupied
- $p(m_i) \to 0 \Rightarrow$ high confidence that grid cell is free
- $p(m_i) \to 0.5 \Rightarrow$ no information about occupancy of grid cell

Problem statement: calculate the posterior $p(m|z_{1:t}, x_{1:t})$

Assumptions:

- A grid is either fully occupied or fully free.
- Conditional independence of grid cells i.e. $p(m) = \prod p(m_i)$

5.2 Algorithm

Log odds is defined as

$$l_o = log\frac{p(m_i)}{1 - p(m_i)}$$

$$p(m_i) = 1 - \frac{1}{1 + exp(l_o)}$$

We are hunting for the posterior $p(m_i|z_{1:t}, x_{1:t})$. If we express this posterior in odds form , we have

$$o_{t,i} = \frac{p(m_i|z_{1:t}, x1:t)}{p(\neg m_i|z_{1:t}, x1:t)}$$

$$= \frac{p(m_i|z_t, x_t)}{p(\neg m_i|z_t, x_t)} . \frac{p(m_i|z_{1:t-1}, x_{1:t-1})}{p(\neg m_i|z_{1:t-1}, x_{1:t-1})} . \frac{p(\neg m_i)}{p(m_i)}$$

In log odds form

$$l_{t,i} = inverse\_sensor\_model + l_{t-1,i} - l_o$$

In each step we will update only the grids which are in the perception range of the sensor at that particular time instant

5.3 Inverse Sensor Model

inverse_sensor_model: is the implementation of inverse measurement model $p(m_i|z_t, x_t)$ in log odds form.

5.4 Scan Matching

"The assumption that poses is known is fundamentally flawed". If we use the pose from raw odometry data the grid map which we recieve is not generally usable.

"**Scan-matching** tries to incrementally align two scans or a map to scan without revisiting the past/map. Some common methods used for scan matching - Iterative Closest point match, RANSAC for outlier rejection . A standard cost function is given below

$$x_t^* = \arg\max_{x_t} \{p(z_t|x_t, m_{t-1})p(x_t|u_{t-1}, x_{t-1}^*)\}$$

5.5 Summary

- Occupancy Grid maps are used in conjuction with Particle filter based SLAM, feature based maps are used in kalman filter based SLAM.
- Occupancy Grid map by itself is not a SLAM algorithm. It assumes known poses.
- Scan matching is used to build usable maps .

6 Fast SLAM

6.1 Review of Particle filters

- Non-parametric recursive Bayes filter
- uses samples to represent belief distribution

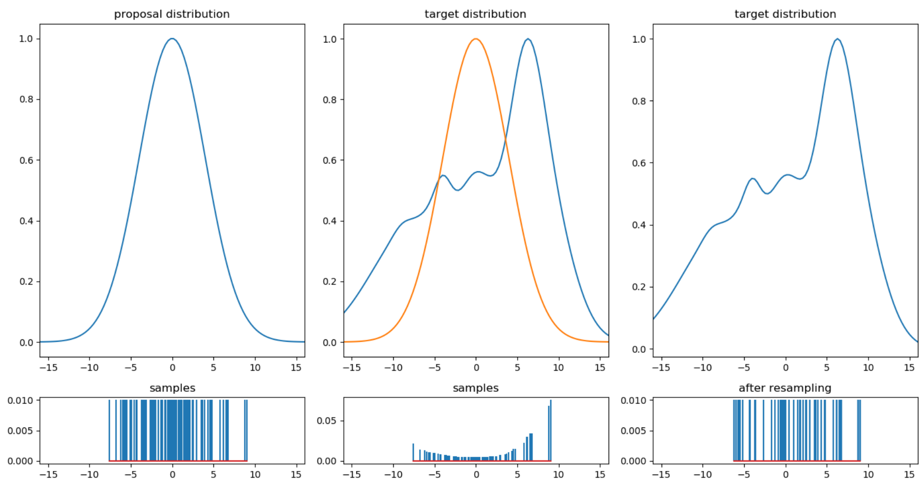


Figure 3: Generating samples from arbitrary distribution using importance sampling

- as opposed to kalman filter family it can model multi modal distributions.
- Uses importance sampling principle to draw samples from arbitrary distributions
  - sampling from proposal distribution :
  $$x_t^{[j]} \sim \pi(x_t|x_{t-1}, u_{t-1})$$
  - Importance weighting :
  $$w_t^{[j]} = \frac{target(x_t^{[j]})}{proposal(x_t^{[j]})} = p(z_t|x_t)$$
  - Resampling : Draw sample i with probability $w_t^{[j]}$

6.2 Rao-Blackwellization for SLAM

SLAM problem is represented as finding the posterior

$$p(x_{1:t}, m_{1,x}, m_{1,y}, \ldots, m_{M,x}, m_{M,y})$$

Particle filters are ideal for low dimensional problems . Hence cant be directly used to solve the SLAM problem(dimension 2N + 3). One approach is to estimate the pose using particle filter and then computing map . Mathematically

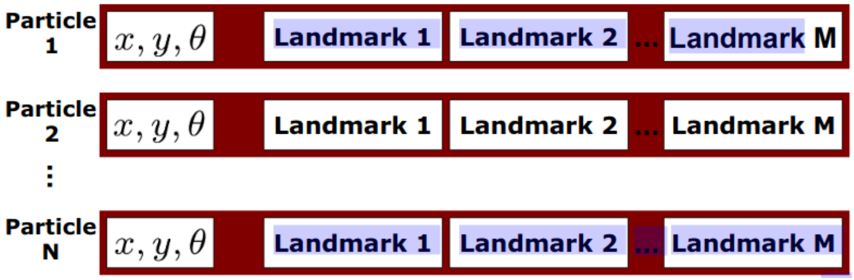


Figure 4: Particle structure in RBPF-slam

this factorisation can be expressed as

$$\begin{aligned} p(a,b) &= p(b|a).p(a) \\ \sim p(x_{0:t}, m_{1:M}|z_{1:t}, u_{1:t}) &= p(x_{0:t}|z_{1:t}, u_{1:t}).p(m_{1:M}|z_{1:t}, u_{1:t}) \\ &= p(x_{0:t}|z_{1:t}, u_{1:t})\Pi p(m_i|z_{1:t}, u_{1:t}) \end{aligned}$$

Effectively we have split it into a path posterior and a map posterior. each particle respresents a path hypothesis and it has an assosicated map with it. Splitting $p(m_{1:M})$ into $\Pi p(m_i)$ reduces the computational complexity further, each of this is calculated by a 2x2 EKF.

Each particle maintains M 2x2 EKF along with the 3 pose variables

6.3 Algorithm

from slides by Prof Syrill Stachniss

FastSLAM1.0_known_correspondence(z_t, c_t, u_t, X_{t-1}):

for k = 1 to N do loop through N particles

Let $\langle x_{t-1}^{[k]}, \langle \mu_{1,t-1}^{[k]}, \Sigma_{1,t-1}^{[k]} \rangle \dots \langle \mu_{M,t-1}^{[k]}, \Sigma_{M,t-1}^{[k]} \rangle \rangle$ be a particle in X_{t-1}

$x_t^{[k]} \sim p(x_t|x_{t-1}, u_t)$ sample pose

$j = c_t$ observed feature with correspondence

if feature j never seen before:

$\mu_{j,t}^{[k]} = h^{-1}(z_t, x_t^{[k]})$ initialize mean

$H = h'(\mu_{j,t}^{[k]}, x_t^{[k]})$ calculate Jacobian

$\Sigma_{j,t}^{[k]} = H^{-1} Q_t (H^{-1})^T$ initialize covariance

$w^[k] = p_0$ default importance weight

else

$\hat{z}^{[k]} = h(\mu_{j,t-1}^{[k]}, x_t^{[k]})$ measurement prediction
$H = h^{'}(\mu_{j,t-1}^{[k]}, x_t^{[k]})$ calculate Jacobian
$Q = H\Sigma_{j,t-1}H^T + Q_t$ measurement Covariance
$K = \Sigma_{j,t-1}^{[k]}H^T Q^{-1}$ calculate Kalman gain
$\mu_{j,t}^{[k]} = \mu_{j,t-1}^{[k]} + K(z_t - \hat{z}^{[k]})$ update mean
$\Sigma_{j,t}^{[k]} = (I - KH)\Sigma_{j,t-1}^{[k]}$ update covariance
$w^{[k]} = |2\pi Q|^{-\frac{1}{2}} exp\{-\frac{1}{2}(z_t - \hat{z}^{[k]})^T Q^{-1}(z_t - \hat{z}^{[k]})\}$

endif
for all unobserved features $j^{'}$ do:
$\langle \mu_{j,t}^{[k]}, \Sigma_{j,t}^{[k]} \rangle = \langle \mu_{j,t-1}^{[k]}, \Sigma_{j,t-1}^{[k]} \rangle$ leave unchanged
end for
$X_t = resample\langle x_{t-1}^{[k]}, \langle \mu_{1,t-1}^{[k]}, \Sigma_{1,t-1}^{[k]} \rangle \ldots, w^{[k]} \rangle$

7 Links and references

Webpage from Prof. Syrill Stachniss can be found here
pythonrobotics