Lecture 3 - Bayes Filters (localization)

State Estimation

- estimate state x given controls & observations, u & z (of any system)
  ↳ position of anything in the world.

↳ $P(x \mid z, u)$ → state component of online SLAM

↳ one way is using bayes filters

Recursive Bayes Filter

$bel(x_t) = p(x_t \mid z_{1:t}, u_{1:t}) = \eta P(z_t \mid x_t, z_{1:t-1}, u_{1:t}) P(x_t \mid z_{1:t-1}, u_{1:t})$

Definition of belief

↳ Bayes Rule:

$$P(A \mid B, C) = \frac{P(B \mid A, C) P(A \mid C)}{P(B \mid C)}$$

$$\boxed{\equiv \eta P(B \mid A, C) P(A \mid C)}$$

↳ denom $P(B \mid C)$ is a normalizing (constant) term so not interested

Markov Standard Assumption

$= \eta P(z_t \mid x_t) P(x_t \mid z_{1:t-1}, u_{1:t})$

↳ given we know state of world x_t, we can ignore prev. commands & observations z & u

$= \eta P(z_t \mid x_t) \int_{x_{t-1}} P(x_t \mid x_{t-1}, z_{1:t-1}, u_{1:t}) P(x_{t-1} \mid z_{1:t-1}, u_{1:t}) \, dx_{t-1}$

↳ Law of Total Probability:

$$\boxed{P(A) = \int P(A \mid B) \cdot P(B) \, dB}$$

$= \eta P(z_t \mid x_t) \int_{x_{t-1}} P(x_t \mid x_{t-1}, u_t) P(x_{t-1} \mid z_{1:t-1}, u_{1:t-1}) \, dx_{t-1}$

↳ Markov assumption again, on both integral terms

$$\boxed{bel(x_t) = \eta P(z_t \mid x_t) \int_{x_{t-1}} P(x_t \mid x_{t-1}, u_t) \, bel(x_{t-1}) \, dx_{t-1}}$$

↳ from def. of belief

↳ Recursive system allows us to compute current prob. distr. of state given commands & observations (prev.)

→ represents sensor reliability.

Prediction & Correction Steps

- Bayes Filter can be written as two steps

1) Prediction Step:

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

2) Correction Step:

$$bel(x_t) = \eta \underbrace{p(z_t \mid x_t)}_{\text{Observation Model}} \overline{bel}(x_t)$$

Different Realizations

↳ Bayes Filter is a framework, there are different realizations

- Different properties:
  - Linear vs. Non-Linear models (for motion & observations)
  - Gaussian only distributions?
  - Parametric vs. Non-Parametric filters
  - ...

In This Course

- Kalman Filter
  - gaussians
  - Linear models

  } better if you assume linear models

- Particle Filter
  - Non-Parametric
  - Arbitrary models

  } more general but inc. computation

Motion Models

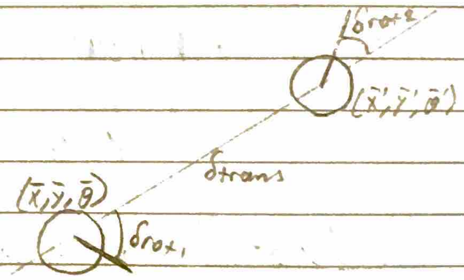
$\overline{bel}(x_t) = \int \boxed{p(x_t \mid u_t, x_{t-1})}\, bel(x_{t-1})\, dx_{t-1}$

↳ estimate current state given prev. state & command.

- 2 types of motion models
  1) odometry based (counting wheel revolutions)
  2) velocity based (used in arial vehicles)

Odometry Motion Model

- Robot moves from $(\bar{x}, \bar{y}, \bar{\theta})$ to $(\bar{x}', \bar{y}', \bar{\theta}')$
- odometry info: $u = (\delta_{rot1}, \delta_{trans}, \delta_{rot2})$
  - $\delta_{trans} = \sqrt{(\bar{x}' - \bar{x})^2 + (\bar{y}' - \bar{y})^2}$
  - $\delta_{rot1} = \operatorname{atan2}(\bar{y}' - \bar{y}, \bar{x}' - \bar{x}) - \bar{\theta}$
  - $\delta_{rot2} = \bar{\theta}' - \bar{\theta} - \delta_{rot1}$

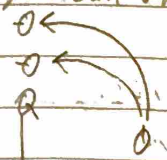


Velocity Motion Model

- Robot moves from (x, y, θ) to (x', y', θ')
- velocity info: $u = (v, w)$ ⟷ (translational velocity, rotational velocity)

↳ you can only travel in arcs

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} -\frac{v}{w}\sin\theta + \frac{v}{w}\sin(\theta + w\Delta t) \\ \frac{v}{w}\cos\theta - \frac{v}{w}\cos(\theta + w\Delta t) \\ w\Delta t + \underbrace{\gamma \Delta t} \end{bmatrix}$$



↳ if you want to end up here, need an additional term for final rotation.

- velocity model usually has more uncertainty in distribution, odometry is more accurate.

Sensor Models

$bel(x_t) = \eta \boxed{p(z_t \mid x_t)} \, \overline{bel}(x_{t-1})$

- will look first @ models for lazer range finders

Model for Lazer Scanners

- Scan Z consists of K measurments

$$z_t = \{ z_t^1, \dots, z_t^K \}$$

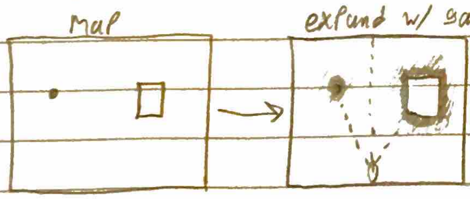
- Individual measurments are independent given the robot position

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

a) Beam-Endpoint Model

↳ is essentially a look-up in a map.

Map ... expand w/ gaussian kernel



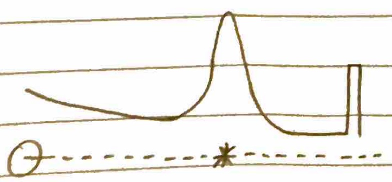
→ just look & take that value

- Computationally inexpensive but works surprisingly well

Ray-Cast Model

↳ more expensive but more accurate

- Consider first obsticle along line of sight



→ Consists of 4 Components

1) gaussian dist. around obsticle
2) exp. decay, allows us to cover dynamic obsticles like ppl or animals (only infront of *)
3) max range reading @ the end
4) Small over all uniform dist. (not explained)

Range-Bearing Model - for Perceiving Landmarks

- Range-bearing: $z_t^i = (r_t^i, \phi_t^i)^T$ → r = distance
  ϕ = orientation of beam w.r.t. robot heading
- Robot Pose: $(x, y, \theta)^T$
- observation of feature j @ location: $(m_{j,x}, m_{j,y})^T$

$$\begin{bmatrix} r_t^i \\ \phi_t^i \end{bmatrix} = \begin{bmatrix} \sqrt{(m_{j,x}-x)^2 + (m_{j,y}-y)^2} \\ \operatorname{atan2}(m_{j,y}-y, m_{j,x}-x) - \theta \end{bmatrix} + Q_t$$

↳ Some noise (usually Gaussian)

Summary

- Bayes Filter is a framework for State estimation
- motion & sensor models are the central models for Bayes filters
- Picking a motion model & a Sensor model you can implement a Bayes filter which is used in SLAM localization