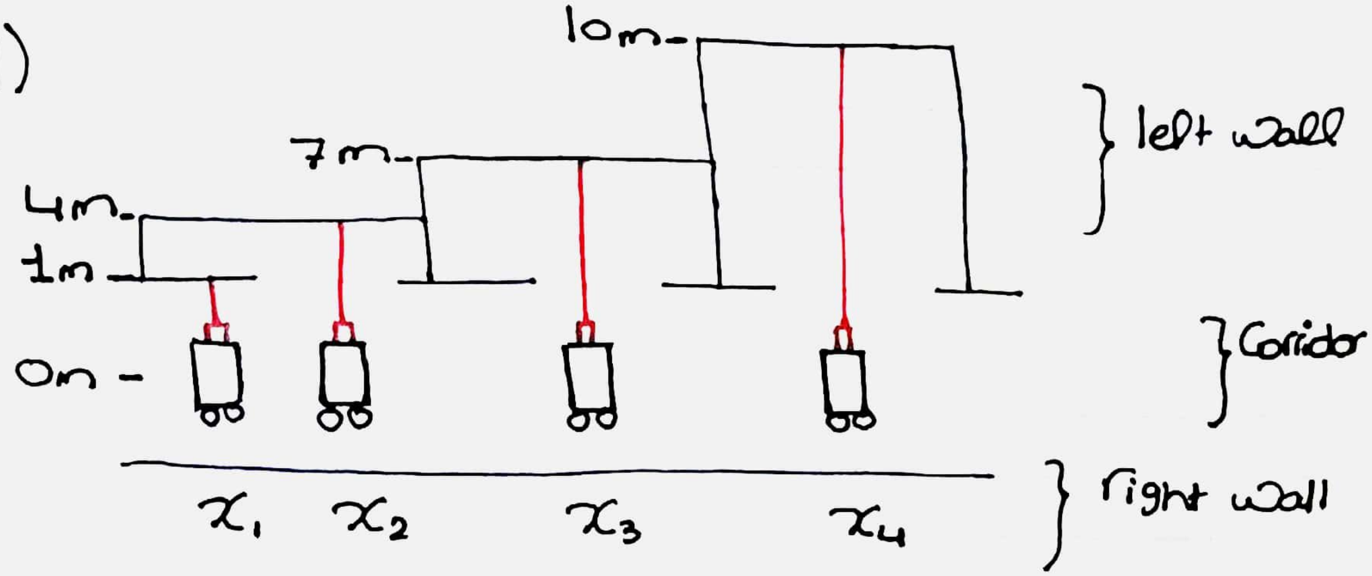
# CR Sheet 2

2.1)




1. Robot moves along corridor & has accurate map

2. At locations $x_1, x_2, x_3, x_4$ it takes measurment $Z_k$ using laser beam to find distance to left wall.

3. The measured distances are

| $Z_1$ | $Z_2$ | $Z_3$ | $Z_4$ |
|---|---|---|---|
| 2.5m | 8m | 6m | 9m |

and its known that they are corrupted with additive Gaussian noise $\sim N(\mu = 1m, \sigma = 2m)$

4. The mapping between $Z_k$ and $x_i$ is unknown

we don't know what $x_i$ caused reading $Z_k$

* We know that

$$Z_k = h(x) + \delta$$

noise

actual reading

reading corresponding to state ignoring noise

- Known for $x_1, x_2, x_3, x_4$ from map

a) • For each of the four measurements, determine the most likely robot pose

• Assume an equally distributed prior & assert that in the end $\sum_i P(x_i|z) = 1$ ①

⑤

⇒ Sol.

For each reading $z_K$
See which location $x_i$
is most probable given the reading $z_K$

i.e., we want $P(x_i|z_K)$ for $K = 1,2,3,4$ and
$i = 1,2,3,4$

• Suppose the reading $z_1$ was due to $x_2$, then

$$z_1 = h(x_2) + \delta = \quad + \delta \sim N(\mu = 7+1, \sigma = 2)$$

which corresponds to the probability

$$P(z_1|x_2) = \frac{1}{\sqrt{2\pi \cdot 2}} \cdot e^{-\frac{(z_1-8)^2}{8}}$$

} Can plug with our r

↳ Prob. of $z_1$ given that it was due to $x_2$.

← Can plug our $z_1$ reading

• Can as well find it for $x_1, x_3, x_4$

• Then by Bayes: $P(x_i|z_1) = \dfrac{P(z_1|x_i)P(x_i)}{P(z_1)}$

→ $P(x_i) = \frac{1}{4}$ due to ①

⟶ After computing for all $x_i$, predict the location to be the one that maximizes it.

• Generaling the Computation, we have:

$$P(Z_K|x_i) = \frac{1}{2\sqrt{2\pi}} \cdot e^{\frac{-(Z_K-(h(x_i)+1))^2}{8}}$$

$$P(x_i|Z_K) = \frac{P(Z_K|x_i)\,P(x_i)}{\sum_{j=1}^{4} P(Z_K|x_j)P(x_j)}$$

• $P(x_i) = \frac{1}{4}$

• Can cancel out $\frac{1}{2\sqrt{2\pi}}$

$$= \frac{e^{-(Z_K-h(x_i)-1)^2/8}}{\sum_{j=1}^{4} e^{-(Z_K-h(x_j)-1)^2/8}} \quad (1)$$

| | $h(x_1)=1$ <br> $x_1$ | $h(x_2)=4$ <br> $x_2$ | $h(x_3)=7$ <br> $x_3$ | $h(x_4)=10$ <br> $x_4$ |
|---|---|---|---|---|
| $Z_1=2.5$ | 0.96 | 0.45 | 0.0228 | 0.0001 |
| $Z_2=8$ | 0.011 | 0.32 | 1 | 0.32 |
| $Z_3=6$ | 0.135 | 0.882 | 0.606 | 0.044 |
| $Z_4=9$ | 0.0022 | 0.135 | 0.882 | 0.606 |

• By Plugging for the numerator

• Dividing by the Sum of the row is eq. to dividing by (1)

• Circled is the best Pred. for each Z

Converting to Prob. (normalizing) we get

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | |
|---|---|---|---|---|---|
| $z_1 = 2.5$ | 0.67 | 0.314 | 0.016 | 0.0000698 | • Predict $x_1$ |
| $z_2 = 8$ | 0.0067 | 0.194 | 0.606 | 0.194 | • Predict $x_3$ |
| $z_3 = 6$ | 0.0809 | 0.529 | 0.363 | 0.026 | • Predict $x_2$ |
| $z_4 = 9$ | 0.0013 | 0.0832 | 0.543 | 0.372 | • Predict $x_3$ |

- Notice that it makes sense that the robot is less confident about its decision for $z_3$ and $z_4$ (highest Prob. not much higher than others) because the measured $z_3$ and $z_4$ lie between $x_2, x_3$ and $x_3, x_4$ respectively and each of these pairs involve two locations really close to each other (3 units difference when $\sigma = 2$)

b) Robot believes that taking measurements at positions $x_2$ and $x_3$ is in general two times as likely as doing so at $x_1$ and $x_4$

→ Use the new Prior info to Calculate the Probs

$$P(x_i | z_k) = \frac{P(z_k | x_i) P(x_i)}{\sum_{j=1}^{4} P(z_k | x_j) P(x_j)}$$

- We know

→ $P(x_1) + P(x_2) + P(x_3) + P(x_4) = 1$

→ $P(x_3) = P(x_2) = 2P(x_1) = 2P(x_4)$

Thus, $P(x_2) = P(x_3) = \frac{1}{3}$ and $P(x_1) = P(x_4) = \frac{1}{6}$

$$P(x_i \mid z_k) \propto \frac{1}{6} e^{-(z_k - h(x_i) - 1)^2/8} \qquad i = 1,4$$

$$\underbrace{P(x_i \mid z_k) \propto \frac{1}{3} e^{-(z_k - h(x_i) - 1)^2/8}}_{\text{numerator of last eqn.}} \qquad i = 2,3$$

- This implies multiplying the 1st & last col. in the table 2 pages ago by $\frac{1}{6}$, the 2nd & 3rd col. by $\frac{1}{3}$ then renormalizing to get table like in last page

c) - Laser scanner reports a faulty measurment of $Z = 30$ in 10% of all cases, no matter the actual distance

→ Means that given any $x_i$, there is 90% chance $Z_k$ will be caused by it (i.e., will come from the dist. $\frac{1}{\sqrt{2\pi}\,2} e^{-(Z_k - h(x_i) - 1)^2/8}$) and 10%

chance it will by 30 (i.e., will come from the dist* that's 0 everywhere but 1 at 30)

- Consequently,

$$P(Z_k \mid x_i) = 0.9\left(\frac{1}{\sqrt{2\pi}\,2} \cdot e^{-(z_k - h(x_i) - 1)^2/8}\right)$$

$$+$$

$$0.1\,(1 . \underbrace{I(Z_k = 30)}_{1 \text{ if } Z_k = 30 \text{ else zero}})$$

→ Since $z_k = 30$ never occurs for $z_1, z_2, z_3, z_4$ we have that

$$P(z_k | x_i) = 0.9 \frac{1}{\sqrt{2\pi}\,2} e^{-(z_k - h(x_i) - 1)^2/8} \quad , k = 1,2,3,4$$

- 0.9 will cancel out from the denominator & numerator while computing $P(x_i | z_k)$

→ hence, no probabibilities in a or b will change.

* Notice that this would've not been the case if

- $z_k = 30$ occured in our measurment (it would increase its prob. given $x_i$)
- The faulty measurment doesn't necessarily result in $z = 30$ (e.g. follows some dist.) ← add instead of $\mathbb{1}$
- Perhaps, will see more of this when we get to sensor models.

2.2)

Extended Kalman Filter ($\mu_{t-1}, \Sigma_{t-1}$)

Prediction:

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

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

Correction:

$$\hat{z}_t = h(\bar{\mu}_t)$$

$$S_t = H_t \bar{\Sigma}_t H_t^T + Q_t$$

$$K_t = \bar{\Sigma}_t H_t^T S_t^{-1}$$

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

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

* Will continue solving in word.

| Component | What it represents | Why its needed |
|---|---|---|
| $\boldsymbol{\mu}_{t-1}$ | 1. The mean of the robot's belief about the state in the last time step<br>2. Because the distribution its Gaussian, it's also the best candidate for the estimated state in that time step. | 1. To predict the state (belief mean) in the current time step after executing an action $\bar{\mu}_t$<br>2. Can also be used for correction $\mu_t$ when there's no action |
| $\boldsymbol{\Sigma}_{t-1}$ | 1. The covariance of the robot's belief about the state in the last time step | 1. To predict the state (belief mean) in the current time step after executing an action $\bar{\Sigma}_t$<br>2. Can also be used for correction $\boldsymbol{\Sigma}_t$ when there's no action |
| $\bar{\boldsymbol{\mu}}_t$ | The mean of the predicted belief (also corresponds to predicted state) | Gives an estimate of the belief's mean after an action is executed. E.g., needed so the robot can know where it is after executing action. |
| $\bar{\boldsymbol{\Sigma}}_t$ | The covariance of the predicted belief | Gives an estimate of the belief's covariance after an action is executed. E.g., needed so the uncertainty in the robot's prediction of the state can be quantified. |
| $\boldsymbol{g}(\boldsymbol{u}_t, \boldsymbol{x}_{t-1})$ | The ideal, generally nonlinear mapping from previous state and action to next state. It represents the action/motion model. | 1. For the robot to predict the mean of the belief after executing an action $\bar{\boldsymbol{\mu}}_t = \boldsymbol{g}\left(\boldsymbol{u}_t, \boldsymbol{\mu}_{t-1_{t-1}}\right)$<br>2. Also needed for $G_t$ |
| $\boldsymbol{G}_t$ | The Jacobian matrix of $g(u_t, x_{t-1})$ with the respect to the state $x_{t-1}$ and evaluated at $x_{t-1} = \mu_{t-1}$ | Provides a first-order approximation of g(.) which justifies the predicted mean computation as well as the predicted covariance where its directly involved $\bar{\boldsymbol{\Sigma}}_t = \boldsymbol{G}_t \boldsymbol{\Sigma}_{t-1} \boldsymbol{G}_t^T + \boldsymbol{R}_t$ |

| | | |
|---|---|---|
| $\boldsymbol{R_t}$ | The covariance matrix of the zero-mean additive Gaussian noise that affects the action model | Needed to perform the covariance prediction (as above) which model's the robot's uncertainty after doing an action |
| $\boldsymbol{\hat{z}_t}$ | The mean of the predicted measurement, given the noisy sensor model | Needed for comparison with the actual measurement later to decide, based on the amount of uncertainty in that prediction, the amount of correction needed to compute $\mu_t$ |
| $\boldsymbol{S_t}$ | The covariance of the predicted measurment, given the noisy sensor model. It represents the uncertainty in the predicted measurment | Needed to compute the Kalman Gain $K_t$ |
| $\boldsymbol{H_t}$ | The Jacobian matrix of h($x_t$) with the respect to the state $x_t$ and evaluated at $x_t = \bar{\mu}_t$ | Provides a first-order approximation of h(.) which justifies the predicted measurment mean computation as well as the predicted measurment covariance where its directly involved $S_t = \boldsymbol{H_t}\bar{\Sigma}_t\boldsymbol{H_t^T} + \boldsymbol{Q_t}$ |
| $\boldsymbol{Q_t}$ | The covariance matrix of the zero-mean additive Gaussian noise that affects the sensor model | Needed to perform the covariance prediction (as above) which model's the uncertainty in the predicted measurment |
| $\boldsymbol{K_t}$ | The Kalman Gain. Represents the uncertainty of the robot's prediction of the measurment assuming no noise relative to the uncertainty in the measurment assuming noise. | Decides how much of the difference between the actual measurment and the predicted one should be propagated as a correction. |
| $\boldsymbol{\mu_t}$ | The mean of the corrected belief | Estimate the current state and used to estimate next state belief |
| $\boldsymbol{\Sigma_t}$ | The covariance of the corrected belief | Know the uncertainty in the current state estimation and used to estimate next state belief |

2.3)

| EKF | UKF |
|---|---|
| **Commonalities** | |
| Both generalize the Kalman filter to handle nonlinear action/sensor models where the world is no longer Guaranteed to be Gaussian. | |
| Both can hence be perceived as Bayes filters that approximate the world as Gaussian | |
| Neither is optimal, they are just approximations | |
| Both can diverge if the non-linearities are extreme enough | |
| Both have the same computational complexity (highly efficient) | |
| **Differences** | |
| In practice, faster with a constant factor | |
| Accurate to only 1st order terms in the Taylor expansion of the nonlinearity | Better approximation; accurate up to the 2nd order terms |
| Linearizes the non-linearity then transforms the previous Gaussian through it to get the new Gaussian | Choose sigma points and transforms them through the non-linear Gaussian then uses weights to construct the new Gaussian |
| Need to be able to compute the Jacobians of the nonlinearities | It's a derivative-free filter. Can handle cases where the Jacobians are hard to compute or can't be computed (indifferentiability). |
| | Involves tuning hyperparameters |
| Can be perceived as a Kalman filter if we let A=G and C=H while using the linearized equations/non-linearities for mean updates. | |