Lecture 11, Oct 12, 2023

Control (Actuator) Noise

- Consider the control input having some noise, so $\boldsymbol{u}_k = \boldsymbol{u}^*_{k|k} + \boldsymbol{n}_k$ where $\boldsymbol{u}_k$ is the actual control input delivered, $\boldsymbol{u}^*_{k|k}$ is the requested control input, and $\boldsymbol{n}_k$ is some zero-mean, Gaussian noise with covariance $\boldsymbol{N}_k$
- After linearization $\boldsymbol{x}_{k+1} = \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{A}_k(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k}) + \boldsymbol{B}_k(\boldsymbol{u}_k - \boldsymbol{u}^*_{k|k}) + \boldsymbol{v}_k$
- In this case our a priori covariance estimate is $\boldsymbol{A}_k\boldsymbol{P}_{k|k}\boldsymbol{A}_k^T + \boldsymbol{B}_k\boldsymbol{N}_k\boldsymbol{B}_k^T + \boldsymbol{Q}_k$
- The rest stays unchanged

Kalman Filtering Example – Observability, Controllability, Detectability and Stabilizability

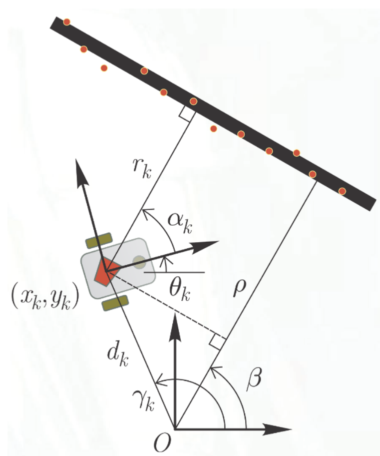

Figure 1: Example scenario.

- Consider a differentially steered robot: $\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k, \boldsymbol{u}_k) = \boldsymbol{A}_k\boldsymbol{x}_k + \underbrace{\begin{bmatrix} \frac{1}{2}\cos\theta_k & \frac{1}{2}\cos\theta_k \\ \frac{1}{2}\sin\theta_k & \frac{1}{2}\sin\theta_k \\ -b^{-1} & -b^{-1} \end{bmatrix}}_{\boldsymbol{B}_k} \underbrace{\begin{bmatrix} \Delta s_{l,k} \\ \Delta s_{r,k} \end{bmatrix}}_{\boldsymbol{u}_k} + \boldsymbol{v}_k$
- The robot measures the angle and perpendicular distance to a wall; the wall is specified in the map as an angle β and distance ρ
- The measurement model is: $\boldsymbol{z}_k = \begin{bmatrix} \alpha_k \\ r_k \end{bmatrix} = \begin{bmatrix} \beta - \theta_k \\ \rho - x_k\cos\beta - y_k\sin\beta \end{bmatrix} + \boldsymbol{w}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{w}_k$
  - Linearize: $\boldsymbol{D}_{k+1} = \dfrac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}_{k+1}^T} = \begin{bmatrix} 0 & 0 & -1 \\ -\cos\beta & -\sin\beta & 0 \end{bmatrix}$
- Note that for Kalman filters to work, the system has to be *observable*, that is, using sufficient measurements of $\boldsymbol{z}$, we can reconstruct $\boldsymbol{x}$
  - A system $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u}, \boldsymbol{z} = \boldsymbol{D}\boldsymbol{x}$ is *observable* if the observability matrix $\boldsymbol{O} = \begin{bmatrix} \boldsymbol{D} \\ \boldsymbol{D}\boldsymbol{A} \\ \vdots \\ \boldsymbol{D}\boldsymbol{A}^{n-1} \end{bmatrix}$ has rank n
- The dual of observability is *controllability*, the ability to achieve any system state by using a sequence of control inputs $\boldsymbol{u}$
  - The controllability matrix is $\boldsymbol{C} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{A}\boldsymbol{B} & \boldsymbol{A}^2\boldsymbol{B} & \cdots & \boldsymbol{A}^n\boldsymbol{B} \end{bmatrix}$, which needs to be rank n for the system to be controllable

- Our system is not observable, since $\boldsymbol{A}$ is the identity and $\boldsymbol{D}$ has rank 2
  - This corresponds to the fact that we don't get enough information by just looking at the wall; we could be anywhere along the wall and still get the same measurement
  - In this case, the output of the filter is not guaranteed to be correct (but we have no way of telling this)
- *Detectability* is a weaker form of observability which requires an $\boldsymbol{L}$ to exist such that $\boldsymbol{A} + \boldsymbol{L}\boldsymbol{D}$ is stable
  - This means that the unobservable states are stable according to their own dynamics
- *Stabilizability* is the dual of detectability, which requires $\boldsymbol{K}$ to exist such that $\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}$ is stable
  - This means that the uncontrollable states are stable according to their own dynamics
- If a system is not observable, but it is still detectable and stabilizable, then the Kalman filter will still converge
- Note that stability in a discrete system requires that all $|\lambda| < 1$ (since we have $\boldsymbol{x}_n = \boldsymbol{A}^n\boldsymbol{x}_0$), whereas in a continuous system we require the eigenvalues to have negative real parts
  - This is known as *Schur stability*

Mapping

- Before, we assumed that we had the location of landmarks; how do we get those landmark locations in the first place?
  - To build a map, we need to localize; to localize, we need a map, leading to a chicken-and-egg problem
  - For now, we will assume we have perfect localization, and see how we can build a map of landmarks
- Suppose we have m landmarks each with coordinate $\boldsymbol{\xi}^{(i)}$; we want to estimate $\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}^{(1)} \\ \vdots \\ \boldsymbol{\xi}^{(m)} \end{bmatrix}$
- We can try to use Kalman filtering!
- Since landmarks don't move, $\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k \implies \boldsymbol{A} = \boldsymbol{1}, \boldsymbol{B} = \boldsymbol{0}$, and there is no noise
- Measurements are modelled by $\boldsymbol{\zeta}_k = \boldsymbol{\eta}(\boldsymbol{x}_k, \boldsymbol{\xi}_k) + \boldsymbol{\varpi}_k$
  - Example: if we treat landmarks as points $(\xi^{(i)}, \eta^{(i)})$ and we measure their bearing ρ and distance ϕ, then:
    * $\rho_k^{(i)} = \sqrt{(\xi_k^{(i)} - x_k)^2 + (\eta_k^{(i)} - y_k)^2}$
    * $\phi_k^{(i)} = \tan^{-1} \dfrac{\eta_k^{(i)} - y_k}{\xi_k^{(i)} - x_k} - \theta_k$
  - This can then be linearized to obtain $\boldsymbol{D}_{k+1}^{(i)}$
- Apply EKF:
  - State estimation:
    1. $\hat{\boldsymbol{\xi}}_{k+1|k} = \hat{\boldsymbol{\xi}}_{k|k}$
    2. $\boldsymbol{\zeta}_{k+1|k} = \boldsymbol{\eta}(\boldsymbol{x}_k, \hat{\boldsymbol{\xi}}_{k+1|k})$
    3. $\boldsymbol{\nu}_{k+1} = \boldsymbol{\zeta}_{k+1} - \hat{\boldsymbol{\zeta}}_{k+1|k}$
    4. $\hat{\boldsymbol{\zeta}}_{k+1|k+1} = \hat{\boldsymbol{\zeta}}_{k+1|k} + \boldsymbol{W}_{k+1}\boldsymbol{\nu}_{k+1}$
  - Covariance estimation:
    1. $\boldsymbol{P}_{k+1|k} = \boldsymbol{P}_{k|k}$
    2. $\boldsymbol{S}_{k+1} = \boldsymbol{D}_{k+1}\boldsymbol{P}_{k+1|k}\boldsymbol{D}_{k+1}^T + \boldsymbol{R}_{k+1}$
    3. $\boldsymbol{W}_{k+1} = \boldsymbol{P}_{k+1|k}\boldsymbol{D}_{k+1}^T\boldsymbol{S}_{k+1}^{-1}$
    4. $\boldsymbol{P}_{k+1|k+1} = \boldsymbol{P}_{k+1|k} - \boldsymbol{W}_{k+1}\boldsymbol{S}_{k+1}\boldsymbol{W}_{k+1}^T$
- However, unlike state, with landmarks we may wish to add new ones during the course of estimation
  - Suppose we want to add new variables to $\boldsymbol{\xi}, \boldsymbol{P}$; we do this at the a priori stage
  - The new landmark state is $\boldsymbol{\xi}_{k|k}^{\text{new}} = \begin{bmatrix} \boldsymbol{\xi}_{k|k} \\ \boldsymbol{\xi}_{k|k}^{(m+1)} \end{bmatrix}$
  - To kick start the state: $\boldsymbol{\xi}_{k|k}^{(m+1)} = \boldsymbol{\gamma}^{(m+1)}(\boldsymbol{x}_k, \boldsymbol{\zeta}_k^{(m+1)})$, where $\boldsymbol{\gamma}$ is the inverse of $\boldsymbol{\eta}$, so we initialize

the new state by inverting the new measurement

  - For $\boldsymbol{P}$, we have $\boldsymbol{P}_{k|k}^{\text{new}} = \begin{bmatrix} \boldsymbol{P}_{k|k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{P}_{k|k}^{(m+1)} \end{bmatrix}$
  - Kick start with $\boldsymbol{P}_{k|k}^{(m+1)} = \boldsymbol{G}_k^{(m+1)} \boldsymbol{\Sigma}_{\zeta}^{(m+1)} \boldsymbol{G}_k^{(m+1)^T} + \boldsymbol{R}_k^{(m+1)}$ where $\boldsymbol{G}^{(m+1)} = \frac{\partial \boldsymbol{\gamma}^{(m+1)}}{\partial \boldsymbol{\zeta}^{(m+1)^T}}$ and $\boldsymbol{R} = \text{cov}(\boldsymbol{\varpi}, \boldsymbol{\varpi})$

Simultaneous Localization and Mapping (SLAM)

- If we need to localize and map at the same time, we need to estimate the robot pose and landmark position simultaneously
- Our overall state just becomes the combination of the robot state $\boldsymbol{x}$ and map $\boldsymbol{\xi}$
- $\begin{bmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\xi}_{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_k & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_k \\ \boldsymbol{\xi}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{B}_k \\ \boldsymbol{0} \\ \boldsymbol{u}_k \end{bmatrix} + \begin{bmatrix} \boldsymbol{v}_k \\ \boldsymbol{0} \end{bmatrix}$
- $\begin{bmatrix} \boldsymbol{z}_k \\ \boldsymbol{\zeta}_k \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}(\boldsymbol{x}_k, \boldsymbol{\xi}_k) \\ \boldsymbol{\eta}(\boldsymbol{x}_k, \boldsymbol{\xi}_k) \end{bmatrix} + \begin{bmatrix} \boldsymbol{w}_k \\ \boldsymbol{\varpi}_k \end{bmatrix}$
- Many SLAM approaches are available, but this is the essence of SLAM