



# Drones Demystified!

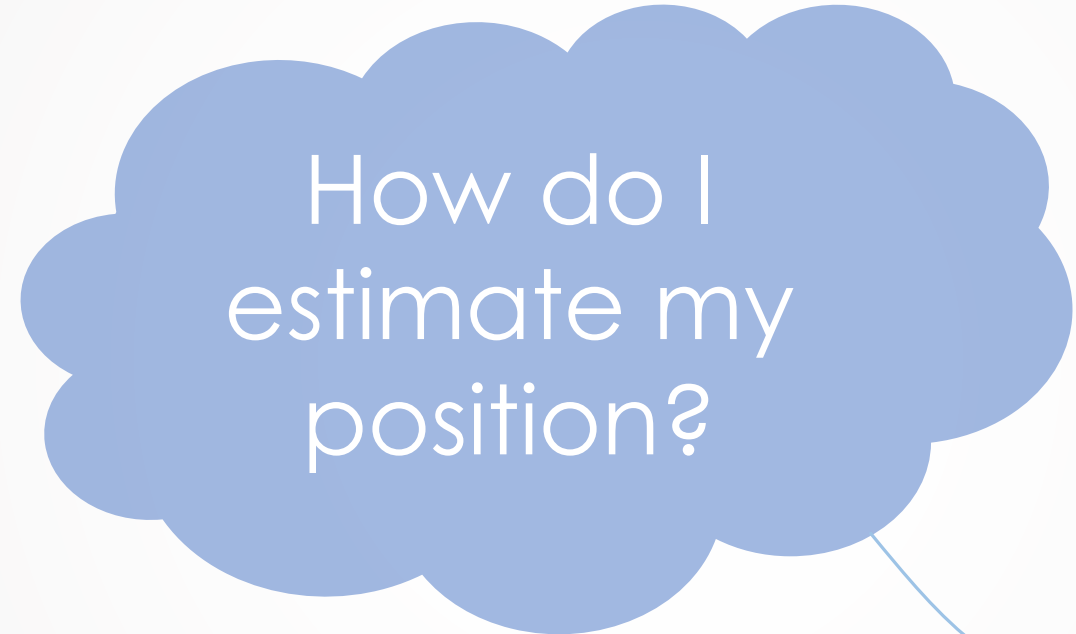

K. Alexis, C. Papachristos, Autonomous Robots Lab, University of Nevada, Reno

A. Tzes, Autonomous Robots & Intelligent Systems Lab, NYU Abu Dhabi

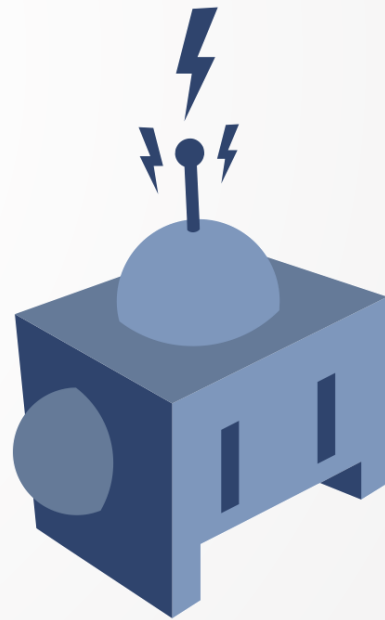


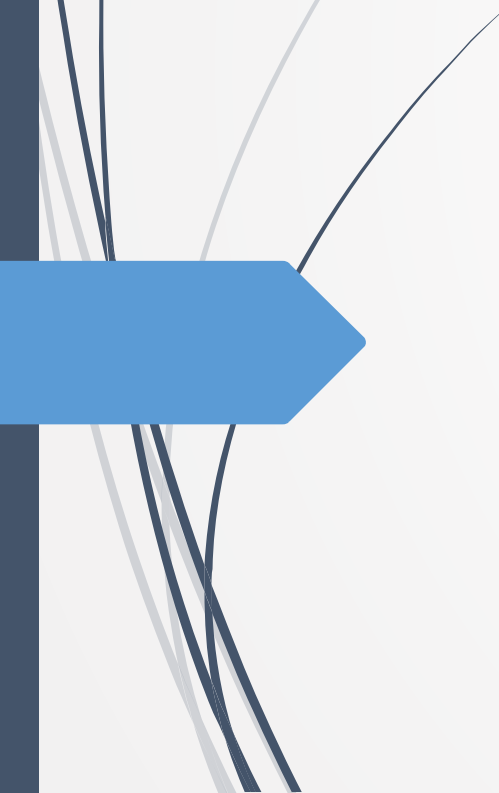
# Drones Demystified!

## Topic: State Estimation



How do I  
estimate my  
position?



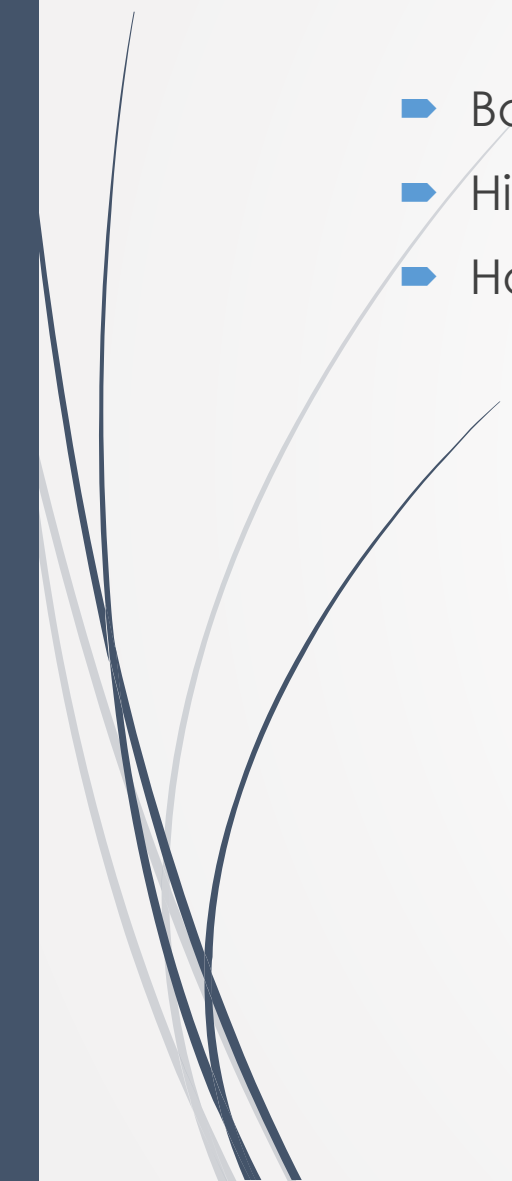


# Drones Demystified!

**Topic: State Estimation – Kalman Filter**

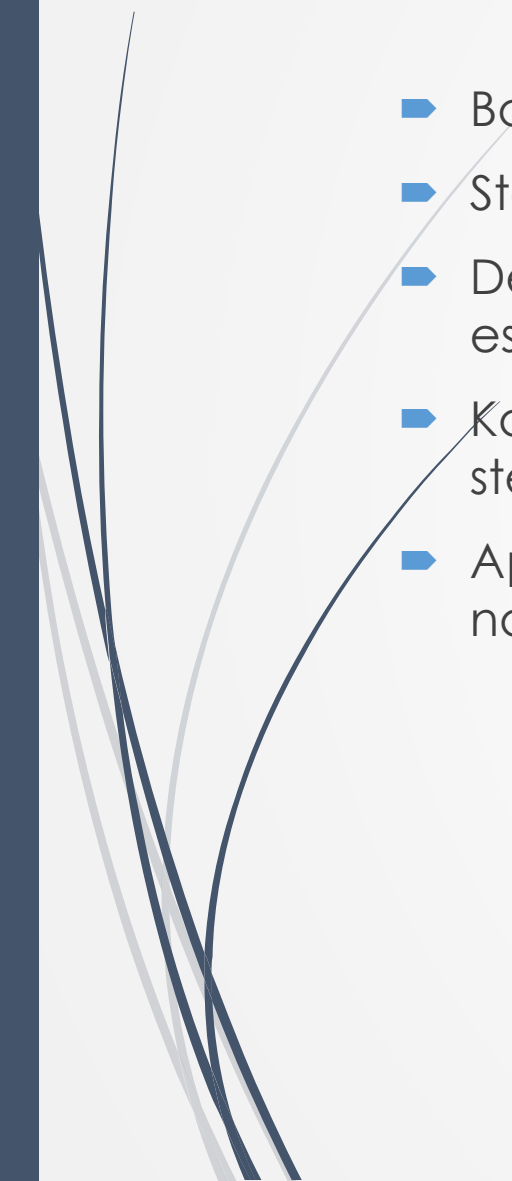


# Kalman Filter

- ▶ Bayes filter is a useful tool for state estimation.
  - ▶ Histogram filter with grid representation is not very efficient.
  - ▶ How can we represent the state more efficiently?
- 

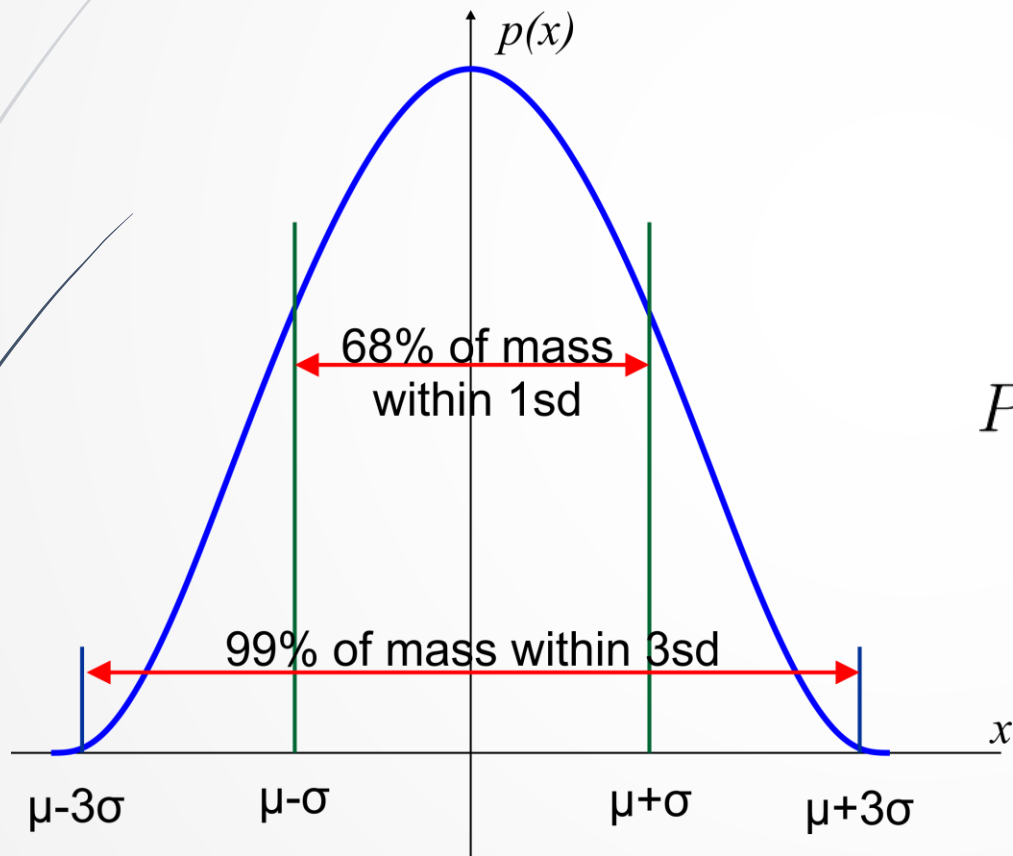


# Kalman Filter

- ▶ Bayes filter with continuous states
  - ▶ State represented with a normal distribution
  - ▶ Developed in the late 1950's. A cornerstone. Designed and first application: estimate the trajectory of the Apollo missiles.
  - ▶ Kalman Filter is very efficient (only requires a few matrix operations per time step).
  - ▶ Applications range from economics, weather forecasting, satellite navigation to robotics and many more.
- 

# Kalman Filter

► Univariate distribution



$$X \sim \mathcal{N}(\mu, \sigma^2)$$

mean

Variance (squared standard deviation)

$$P(X = x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2} \frac{(x - \mu)^2}{\sigma^2}\right)$$

# Kalman Filter

- ▶ Multivariate normal distribution:  $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
- ▶ Mean:  $\boldsymbol{\mu} \in \mathcal{R}^n$
- ▶ Covariance:  $\boldsymbol{\Sigma} \in \mathbf{R}^{n \times m}$
- ▶ Probability density function:

$$p(\mathbf{X} = \mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{n/2} |\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right)$$



# Properties of Normal Distributions

- Linear transformation – remains Gaussian

$$\begin{aligned}\mathbf{X} &\sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \mathbf{Y} \sim \mathbf{A}\mathbf{X} + \mathbf{B} \\ \Rightarrow \mathbf{Y} &\sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu} + \mathbf{B}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T)\end{aligned}$$

- Intersection of two Gaussians – remains Gaussian

$$\mathbf{X}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \mathbf{X}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$

$$p(\mathbf{X}_1)p(\mathbf{X}_2) = \mathcal{N}\left(\frac{\boldsymbol{\Sigma}_2}{\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2}\boldsymbol{\mu}_1 + \frac{\boldsymbol{\Sigma}_1}{\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2}\boldsymbol{\mu}_2, \frac{1}{\boldsymbol{\Sigma}_1^{-1} + \boldsymbol{\Sigma}_2^{-1}}\right)$$

# Properties of Normal Distributions

- Linear transformation – remains Gaussian

$$\begin{aligned}\mathbf{X} &\sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \mathbf{Y} \sim \mathbf{A}\mathbf{X} + \mathbf{B} \\ \Rightarrow \mathbf{Y} &\sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu} + \mathbf{B}, \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^T)\end{aligned}$$

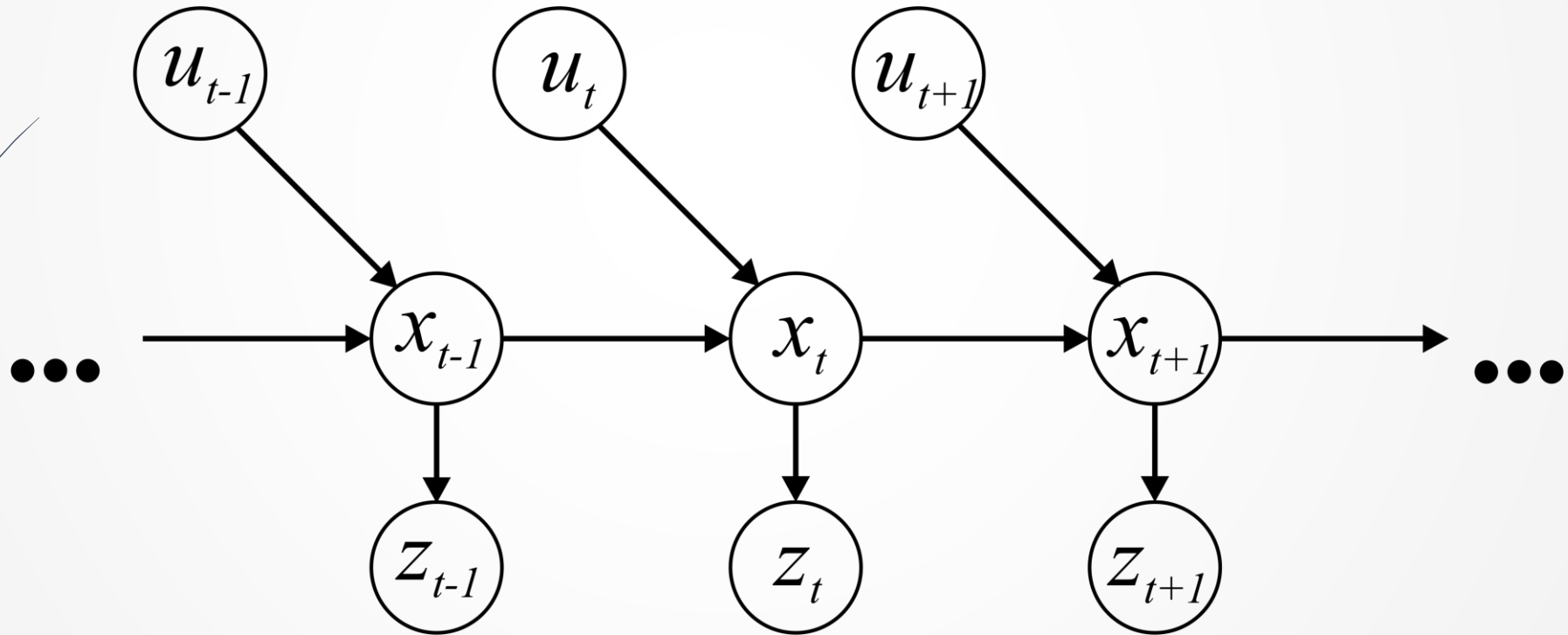
- Intersection of two Gaussians – remains Gaussian

$$\mathbf{X}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \mathbf{X}_2 \sim \mathcal{N}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$

$$p(\mathbf{X}_1)p(\mathbf{X}_2) = \mathcal{N}\left(\frac{\boldsymbol{\Sigma}_2}{\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2}\boldsymbol{\mu}_1 + \frac{\boldsymbol{\Sigma}_1}{\boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_2}\boldsymbol{\mu}_2, \frac{1}{\boldsymbol{\Sigma}_1^{-1} + \boldsymbol{\Sigma}_2^{-1}}\right)$$

# Linear Process Model

- Consider a time-discrete stochastic process (Markov chain)



# Linear Process Model

- ▶ Consider a time-discrete stochastic process
- ▶ Represent the estimated state (belief) with a Gaussian

$$\mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

# Linear Process Model

- ▶ Consider a time-discrete stochastic process
- ▶ Represent the estimated state (belief) with a Gaussian

$$\mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

- ▶ Assume that the system evolves linearly over time, then

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1}$$

# Linear Process Model

- ▶ Consider a time-discrete stochastic process
- ▶ Represent the estimated state (belief) with a Gaussian

$$\mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

- ▶ Assume that the system evolves linearly over time, then depends linearly on the controls

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t$$

# Linear Process Model

- ▶ Consider a time-discrete stochastic process
- ▶ Represent the estimated state (belief) with a Gaussian

$$\mathbf{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$$

- ▶ Assume that the system evolves linearly over time, then depends linearly on the controls, and has zero-mean, normally distributed process noise

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t + \boldsymbol{\epsilon}_t$$

- ▶ With  $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$



# Linear Observations

- ▶ Further, assume we make observations that depend linearly on the state

$$\mathbf{z}_t = \mathbf{C}\mathbf{x}_t$$



# Linear Observations

- ▶ Further, assume we make observations that depend linearly on the state and that are perturbed zero-mean, normally distributed observation noise

$$\mathbf{z}_t = \mathbf{C}\mathbf{x}_t + \delta_t$$

- ▶ With  $\delta_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$

# Kalman Filter

- Estimates the state  $x_t$  of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t + \epsilon_t$$

- And (linear) measurements of the state

$$\mathbf{z}_t = \mathbf{C}\mathbf{x}_t + \delta_t$$

- With  $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$  and  $\delta_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$

# Kalman Filter

▶ State  $\mathbf{x} \in \mathbb{R}^n$

▶ Controls  $\mathbf{u} \in \mathbb{R}^l$

▶ Observations  $\mathbf{z} \in \mathbb{R}^k$

▶ Process equation  $\mathbf{x}_t = \underset{nxn}{\mathbf{A}}\mathbf{x}_{t-1} + \underset{nxl}{\mathbf{B}}\mathbf{u}_t + \epsilon_t$

▶ Measurement equation  $\mathbf{z}_t = \underset{nxk}{\mathbf{C}}\mathbf{x}_t + \delta_t$

# Kalman Filter

- ▶ Initial belief is Gaussian

$$Bel(x_0) = \mathcal{N}(\mathbf{x}_0; \mu_0, \Sigma_0)$$

- ▶ Next state is also Gaussian (linear transformation)

$$\mathbf{x}_t \sim \mathcal{N}(\mathbf{A}\mathbf{x}_t + \mathbf{B}\mathbf{u}_t, \mathbf{Q})$$

- ▶ Observations are also Gaussian

$$\mathbf{z}_t \sim \mathcal{N}(\mathbf{C}\mathbf{x}_t, \mathbf{R})$$

# Recall: Bayes Filter Algorithm

- ▶ For each step, do:
  - ▶ Apply motion model

$$\overline{Bel}(\mathbf{x}_t) = \int p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) Bel(\mathbf{x}_{t-1}) d\mathbf{x}_{t-1}$$

- ▶ Apply sensor model

$$Bel(\mathbf{x}_t) = \eta p(\mathbf{z}_t | \mathbf{x}_t) \overline{Bel}(\mathbf{x}_t)$$

# From Bayes Filter to Kalman Filter

- ▶ For each step, do:
  - ▶ Apply motion model

$$\overline{Bel}(\mathbf{x}_t) = \int \underbrace{p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)}_{\mathcal{N}(\mathbf{x}_t; \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t, \mathbf{Q})} \underbrace{Bel(\mathbf{x}_{t-1})}_{\mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1})} d\mathbf{x}_{t-1}$$

# From Bayes Filter to Kalman Filter

- ▶ For each step, do:
  - ▶ Apply motion model

$$\begin{aligned}\overline{Bel}(\mathbf{x}_t) &= \int \underbrace{p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t)}_{\mathcal{N}(\mathbf{x}_t; \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_t, \mathbf{Q})} \underbrace{Bel(\mathbf{x}_{t-1})}_{\mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}, \Sigma_{t-1})} d\mathbf{x}_{t-1} \\ &= \mathcal{N}(\mathbf{x}_t; \mathbf{A}\mu_{t-1} + \mathbf{B}\mathbf{u}_t, \mathbf{A}\Sigma\mathbf{A}^T + \mathbf{Q}) \\ &= \mathcal{N}(\mathbf{x}_t; \bar{\mu}_t, \bar{\Sigma}_t)\end{aligned}$$

# From Bayes Filter to Kalman Filter

- ▶ For each step, do:
  - ▶ Apply sensor model

$$\begin{aligned}\overline{Bel}(\mathbf{x}_t) &= \eta \underbrace{p(\mathbf{z}_t | \mathbf{x}_t)}_{\mathcal{N}(\mathbf{z}_t; \mathbf{C}\mathbf{x}_t, \mathbf{R})} \underbrace{\overline{Bel}(\mathbf{x}_t)}_{\mathcal{N}(\mathbf{x}_t; \bar{\boldsymbol{\mu}}_t, \bar{\boldsymbol{\Sigma}}_t)} \\ &= \mathcal{N}(\mathbf{x}_t; \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\boldsymbol{\mu}}), (\mathbf{I} - \mathbf{K}_t)\mathbf{C})\bar{\boldsymbol{\Sigma}}) \\ &= \mathcal{N}(x_t; \mu_t, \boldsymbol{\Sigma}_t)\end{aligned}$$

- ▶ With  $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t \mathbf{C}^T (\mathbf{C} \bar{\boldsymbol{\Sigma}}_t \mathbf{C}^T + \mathbf{R})^{-1}$  (Kalman Gain)



# From Bayes Filter to Kalman Filter

Blends between our previous estimate  $\bar{\mu}_t$  and the discrepancy between our sensor observations and our predictions.

The degree to which we believe in our sensor observations is the Kalman Gain. And this depends on a formula based on the errors of sensing etc. In fact it depends on the ratio between our uncertainty  $\Sigma$  and the uncertainty of our sensor observations  $R$ .

$$\bar{\mu}_t + \mathbf{K}_t (\mathbf{z}_t - \mathbf{C}\bar{\mu})$$

old mean                      Kalman Gain

# From Bayes Filter to Kalman Filter

- ▶ For each step, do:
  - ▶ Apply sensor model

$$\begin{aligned}\overline{Bel}(\mathbf{x}_t) &= \eta \underbrace{p(\mathbf{z}_t | \mathbf{x}_t)}_{\mathcal{N}(\mathbf{z}_t; \mathbf{C}\mathbf{x}_t, \mathbf{R})} \underbrace{\overline{Bel}(\mathbf{x}_t)}_{\mathcal{N}(\mathbf{x}_t; \bar{\boldsymbol{\mu}}_t, \bar{\boldsymbol{\Sigma}}_t)} \\ &= \mathcal{N}(\mathbf{x}_t; \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\boldsymbol{\mu}}), (\mathbf{I} - \mathbf{K}_t)\mathbf{C})\bar{\boldsymbol{\Sigma}}) \\ &= \mathcal{N}(x_t; \mu_t, \boldsymbol{\Sigma}_t)\end{aligned}$$

- ▶ With  $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t \mathbf{C}^T (\mathbf{C} \bar{\boldsymbol{\Sigma}}_t \mathbf{C}^T + \mathbf{R})^{-1}$  (Kalman Gain)

# Kalman Filter Algorithm

- ▶ For each step, do:
  - ▶ Apply motion model (prediction step)

$$\bar{\boldsymbol{\mu}}_t = \mathbf{A}\boldsymbol{\mu}_{t-1} + \mathbf{B}\mathbf{u}_t$$

$$\bar{\boldsymbol{\Sigma}}_t = \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\top + \mathbf{Q}$$

- ▶ Apply sensor model (correction step)

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\boldsymbol{\mu}}_t)$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\bar{\boldsymbol{\Sigma}}_t$$

- ▶ With  $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t\mathbf{C}^\top (\mathbf{C}\bar{\boldsymbol{\Sigma}}_t\mathbf{C}^\top + \mathbf{R})^{-1}$

# Kalman Filter Algorithm

Prediction & Correction steps  
can happen in any order.

- ▶ For each step, do:
  - ▶ Apply motion model (**prediction step**)

$$\bar{\boldsymbol{\mu}}_t = \mathbf{A}\boldsymbol{\mu}_{t-1} + \mathbf{B}\mathbf{u}_t$$

$$\bar{\boldsymbol{\Sigma}}_t = \mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\top + \mathbf{Q}$$

- ▶ Apply sensor model (**correction step**)

$$\boldsymbol{\mu}_t = \bar{\boldsymbol{\mu}}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\boldsymbol{\mu}}_t)$$

$$\boldsymbol{\Sigma}_t = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\bar{\boldsymbol{\Sigma}}_t$$

- ▶ With  $\mathbf{K}_t = \bar{\boldsymbol{\Sigma}}_t\mathbf{C}^\top (\mathbf{C}\bar{\boldsymbol{\Sigma}}_t\mathbf{C}^\top + \mathbf{R})^{-1}$

# Kalman Filter Algorithm

Prediction & Correction steps  
can happen in any order.

## Prediction

$$\bar{\mu}_t = \mathbf{A}\mu_{t-1} + \mathbf{B}u_t$$

$$\bar{\Sigma}_t = \mathbf{A}\Sigma\mathbf{A}^\top + \mathbf{Q}$$

## Correction

$$\mu_t = \bar{\mu}_t + \mathbf{K}_t(\mathbf{z}_t - \mathbf{C}\bar{\mu}_t)$$

$$\Sigma_t = (\mathbf{I} - \mathbf{K}_t\mathbf{C})\bar{\Sigma}_t$$

$$\mathbf{K}_t = \bar{\Sigma}_t\mathbf{C}^\top(\mathbf{C}\bar{\Sigma}_t\mathbf{C}^\top + \mathbf{R})^{-1}$$

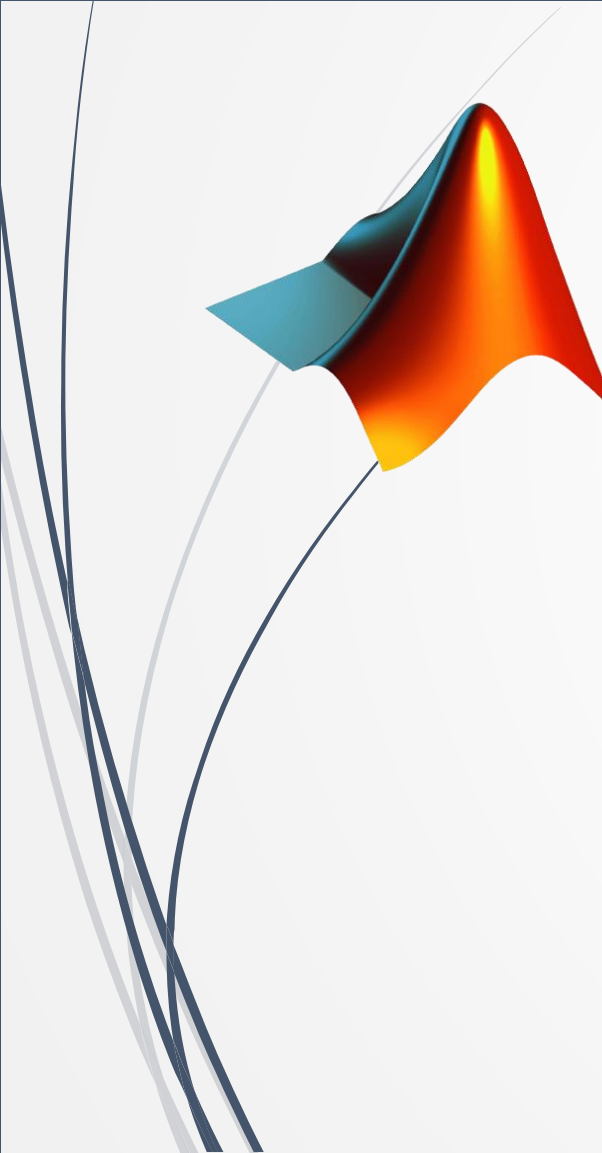
# Complexity

- ▶ Highly efficient: Polynomial in the measurement dimensionality  $k$  and state dimensionality  $n$

$$O(k^{2.376} + n^2)$$

- ▶ Optimal for linear Gaussian systems
  - ▶ But most robots are nonlinear! This is why in practice we use Extended Kalman Filters and other approaches.

# Code Examples and Tasks

- 
- ▶ KF, EKF, UKF
    - ▶ Kalman Filter: [https://github.com/unr-arl/drones\\_demystified/tree/master/matlab/state-estimation/kalman-filter](https://github.com/unr-arl/drones_demystified/tree/master/matlab/state-estimation/kalman-filter)

# Find out more

- ▶ <http://www.autonomousrobotslab.com/the-kalman-filter.html>
- ▶ <http://aerostudents.com/files/probabilityAndStatistics/probabilityTheoryFullVersion.pdf>
- ▶ <http://www.cs.unc.edu/~welch/kalman/>
- ▶ [http://home.wlu.edu/~levys/kalman\\_tutorial/](http://home.wlu.edu/~levys/kalman_tutorial/)
- ▶ <https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python>
- ▶ <http://www.autonomousrobotslab.com/literature-and-links.html>





**Thank you!**

Please ask your question!