Lecture 5 | Kalman Filters

Secure Autonomous and Cyber-Physical Systems

CS 599 001/ECE 599 004

Winter 2022

Prof. Sibin Mohan

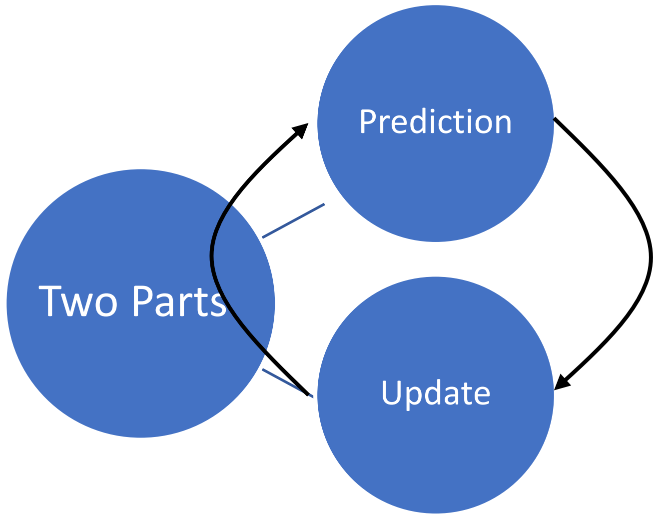
https://bit.ly/secureauto2022

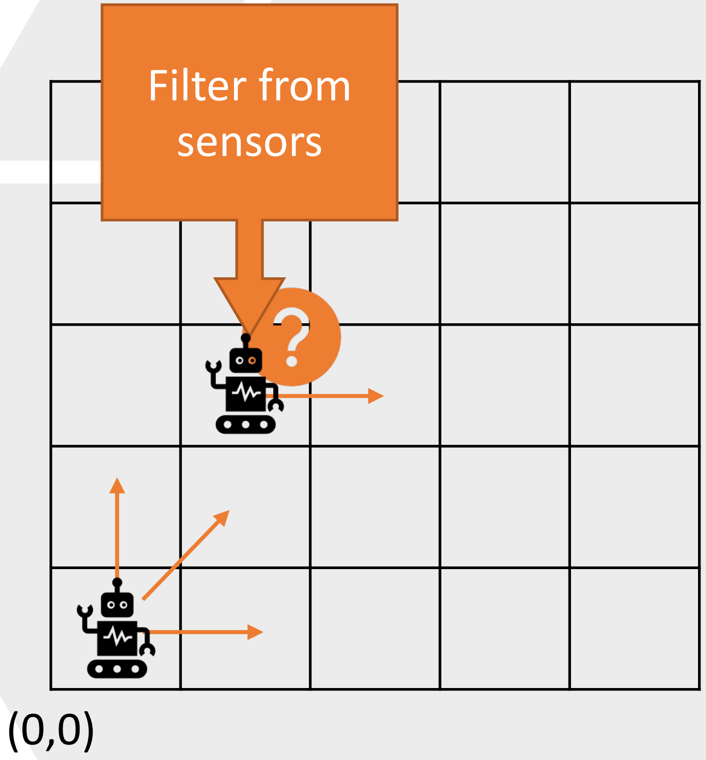
Kalman Filter

- For data fusion ➔ estimate the state of a dynamic system
  - In the present (filtering)
  - past (smoothing)
  - future (prediction)
- Estimate the **state** of a robot from odometry data+observations
- Bayesian Filter

Bayes Filter

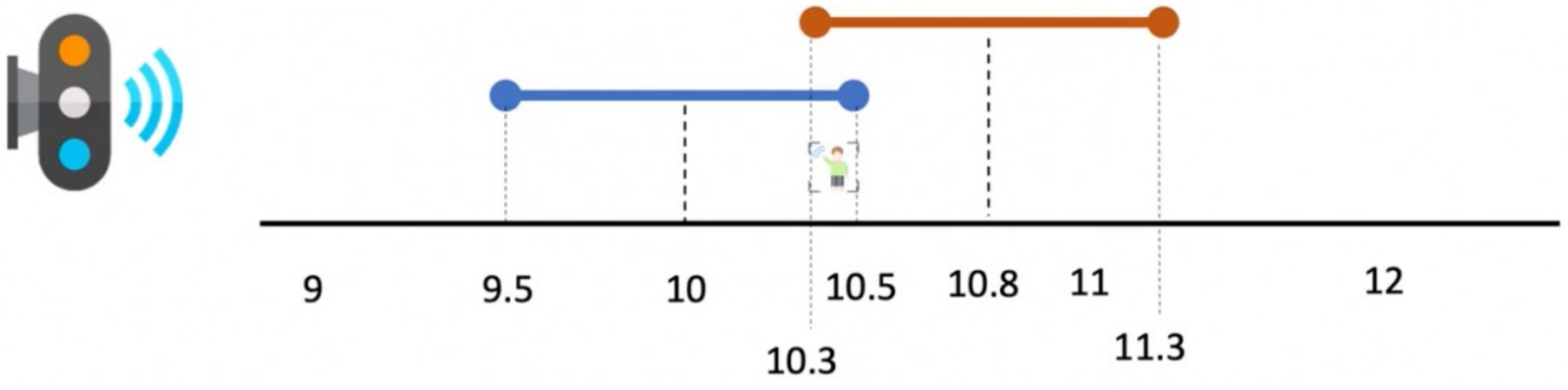
- probabilistic approach
- estimating an **unknown PDF**
  - recursively over time using
  - incoming **measurements** and
  - a mathematical process **model**





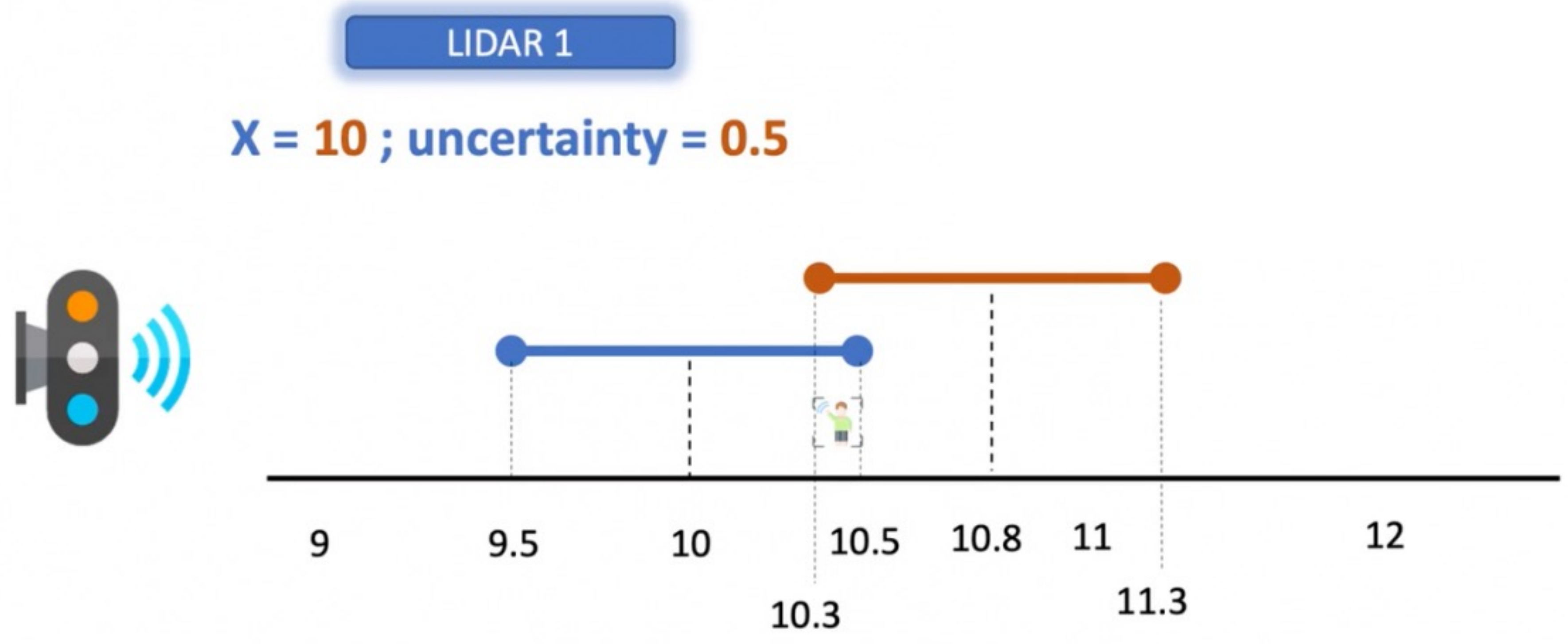
Kalman Filter

Sensors capture incomplete or noisy information



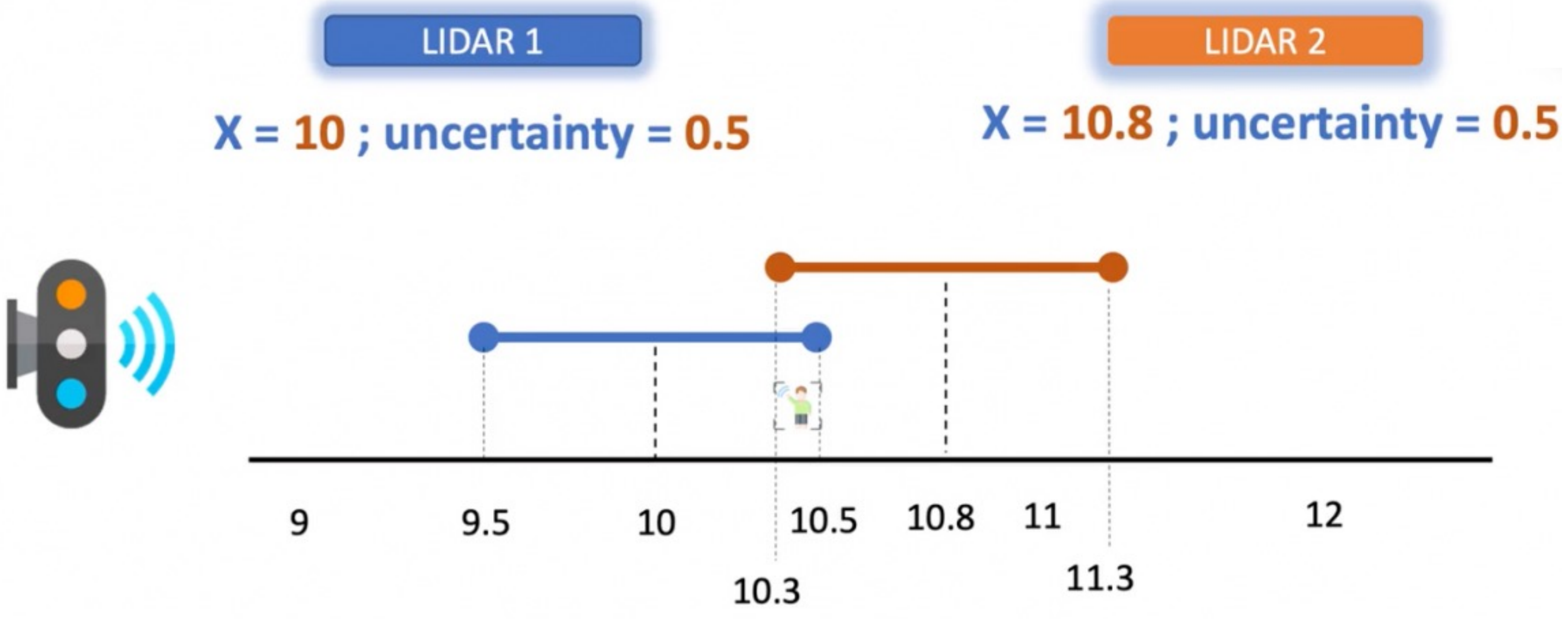
Kalman Filter

Sensors capture incomplete or noisy information



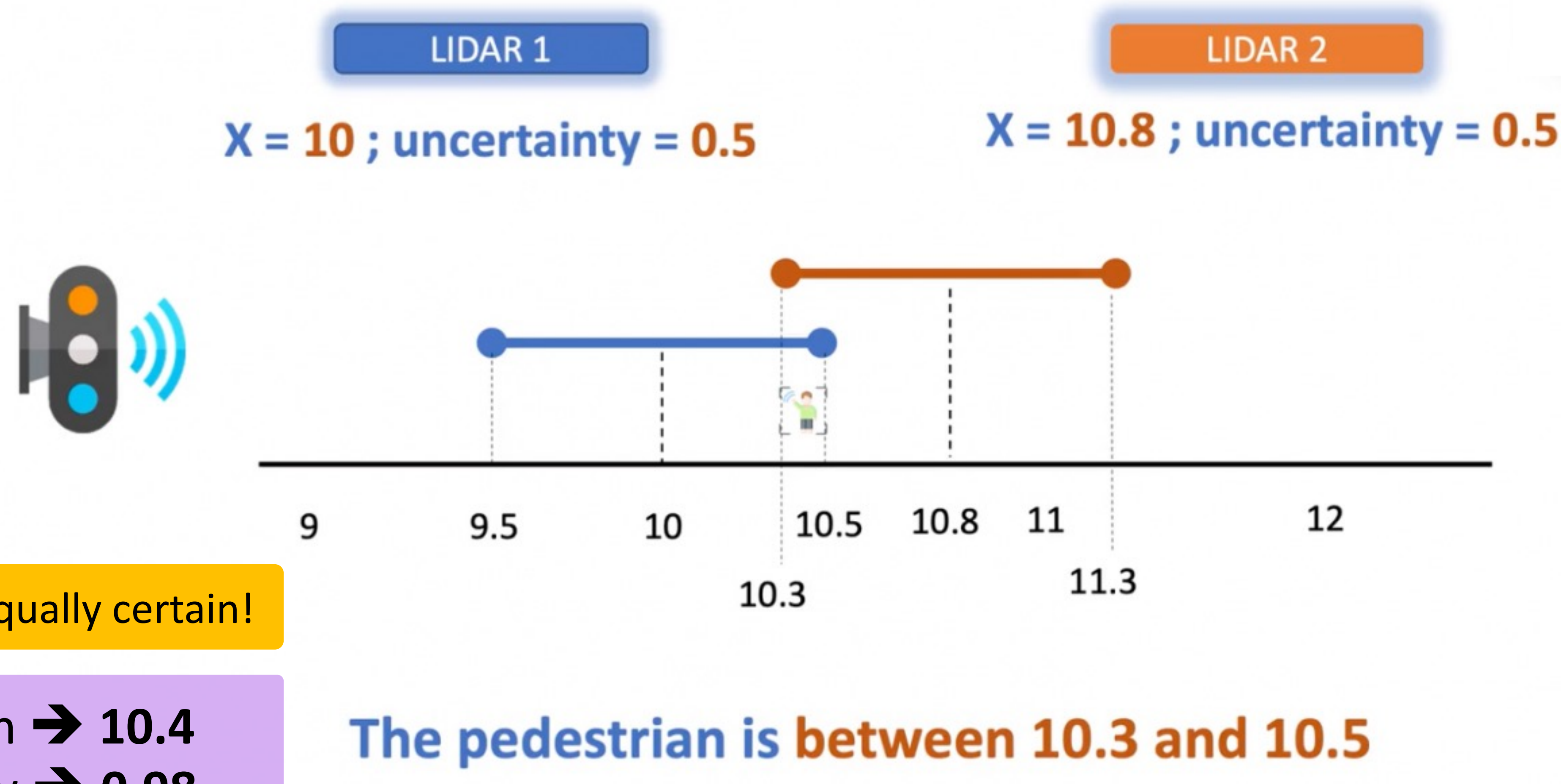
Kalman Filter

Sensors capture incomplete or noisy information



Kalman Filter

Sensors capture incomplete or noisy information

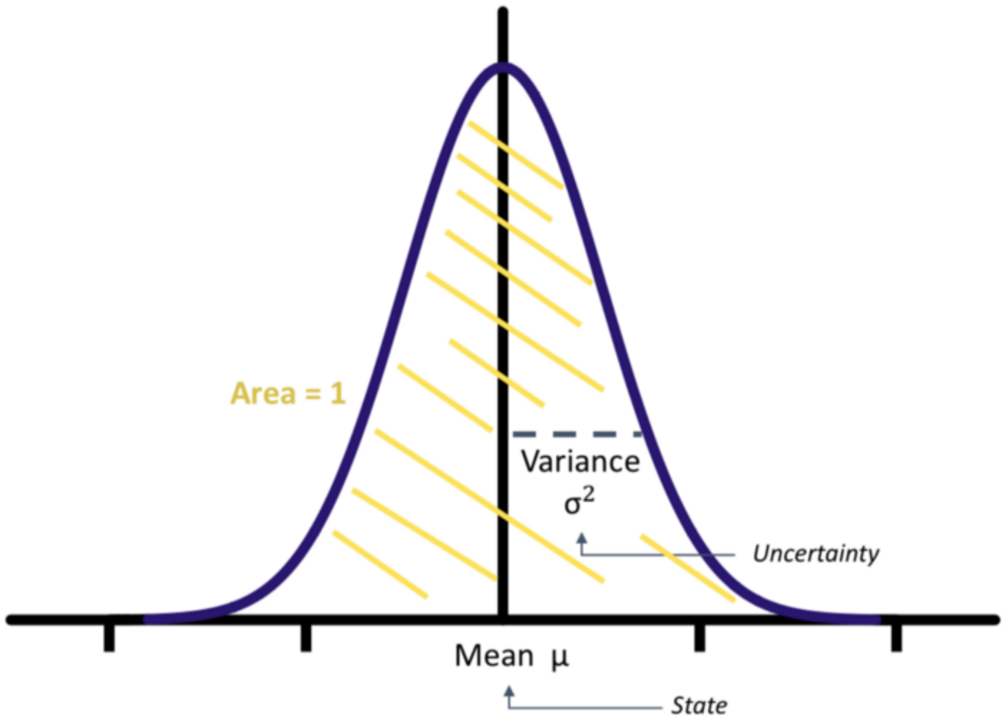


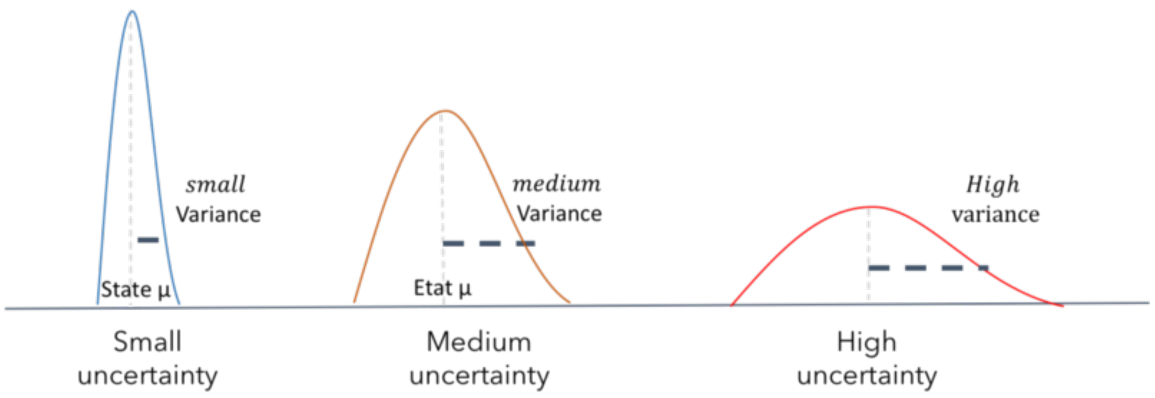
Both sensors equally certain!

Pedestrian → 10.4
probability → 0.98

Gaussians

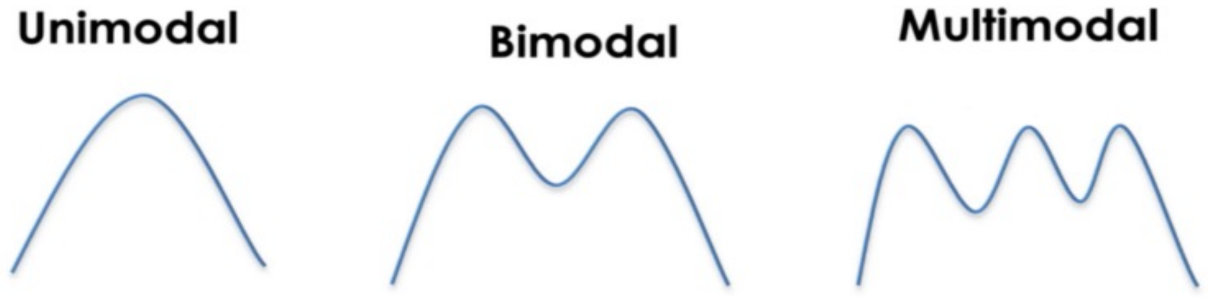
Kalman Filters express state and uncertainty using **Gaussians**





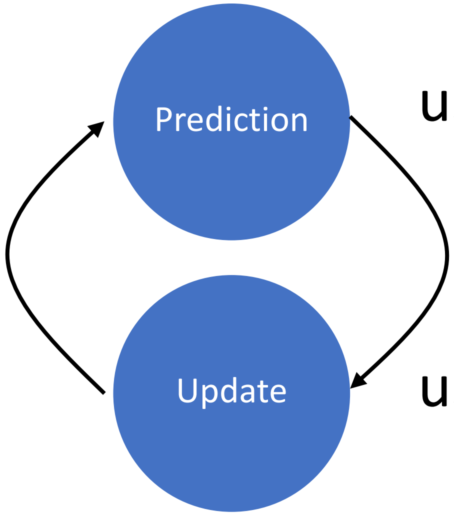
Kalman Filter and Gaussians

- Kalman Filter is unimodal
  - Single peak each time



- An obstacle is **not** both, 10m away with 90% and 8m away with 70% probability
- **9.7m away with 98% or nothing**

Kalman Filter | Under the Hood



use estimated state to **predict future** state+uncertainty

use observations of sensors to **correct** prediction+**improve** accuracy

state of system

position

$$x = \begin{pmatrix} p \\ v \end{pmatrix}$$

velocity

Only needs **current observations** and **previous predictions**

Kalman Filter | Steps

- Imagine sensor fusion between LiDAR and RADAR
- Only their outputs (late fusion)
- Kalman Filter: unimodals and Gaussians to represent state/uncertainty
- Prediction/update cycle

Kalman Filters | Predictions

- estimate state x' and uncertainty P' at time t+1

transition matrix from t to t+1

estimated state of system at t+1

old state of system at time t

$$x' = Fx + u$$

noise

estimated uncertainty of system at t+1

old uncertainty of system at time t

$$P' = FPF^T + Q$$

covariance matrix including noise

Kalman Filter | Predictions [contd.]

- New position = former position (x) times matrix (F)
- F ➔ motion model
  - matrix describing how to move from t to t+1

position (t+1) = position (t) + velocity (t)*time
velocity (t+1) = velocity (t)

constant velocity

- Can consider other motion models for F
  - Constant Turn Rate, Constant Velocity, Constant Acceleration

Kalman Filter | Prediction Matrix Example

$$x' = F \cdot x$$

$$x' = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix}$$

$$x' = \left| \begin{array}{l} \text{position} + \text{velocity} \cdot \Delta t \\ \dot{x} \end{array} \right.$$

Kalman Filter | Update Measurements

- Adjust position, correct how to update next step

difference between
actual measurements and prediction

actual measurement

observation matrix

$$y = z - \underbrace{Hx'}_{\text{prediction}}$$

system error

Kalman Gain [between 0 and 1]

$$S = HP'H^T + R$$

sensor noise

$$K = P'H^TS^{-1}$$

Kalman Filter | Final Update

- Compute new x and P

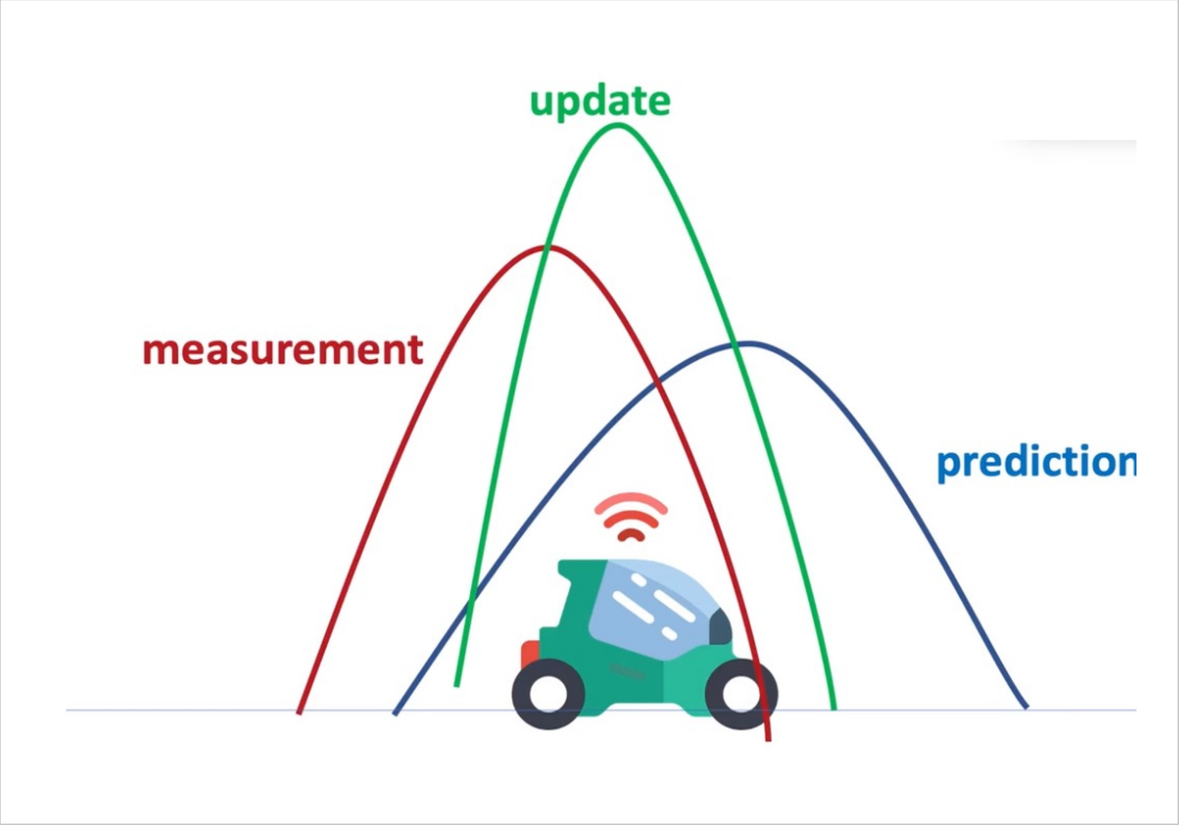
$$x = x' + Ky$$

$$P = (I - KH)P'$$

Kalman Filter | Bayesian Filtering

- We want to **estimate posterior**
  - posterior = prior (prediction) * likelihood (measurement)

$$\underbrace{P(A|B)}_{\text{posterior}} = \frac{\overbrace{P(B|A)}^{\text{likelihood}} * \overbrace{P(A)}^{\text{prior}}}{\underbrace{P(B)}_{\text{normalizer}}}$$



RADAR/LiDAR Sensor Fusion Flow

Each time a sensor value comes in ➔ new prediction+update cycle

RADAR/LiDAR Sensor Fusion Flow

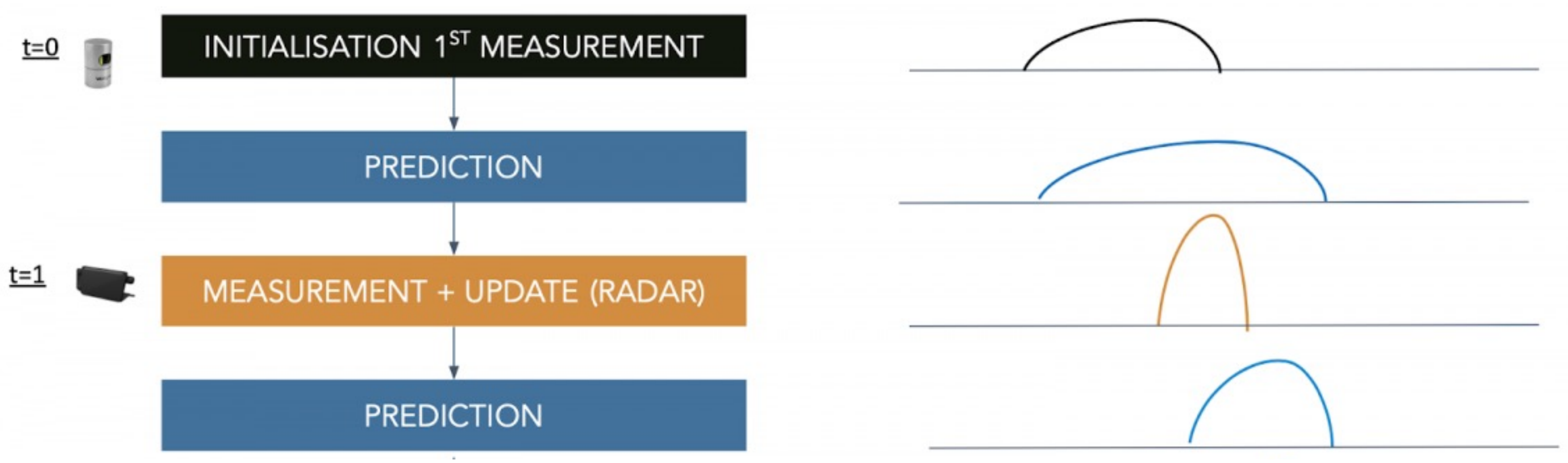
Each time a sensor value comes in → new prediction+update cycle

RADAR/LiDAR Sensor Fusion Flow

Each time a sensor value comes in ➔ new prediction+update cycle

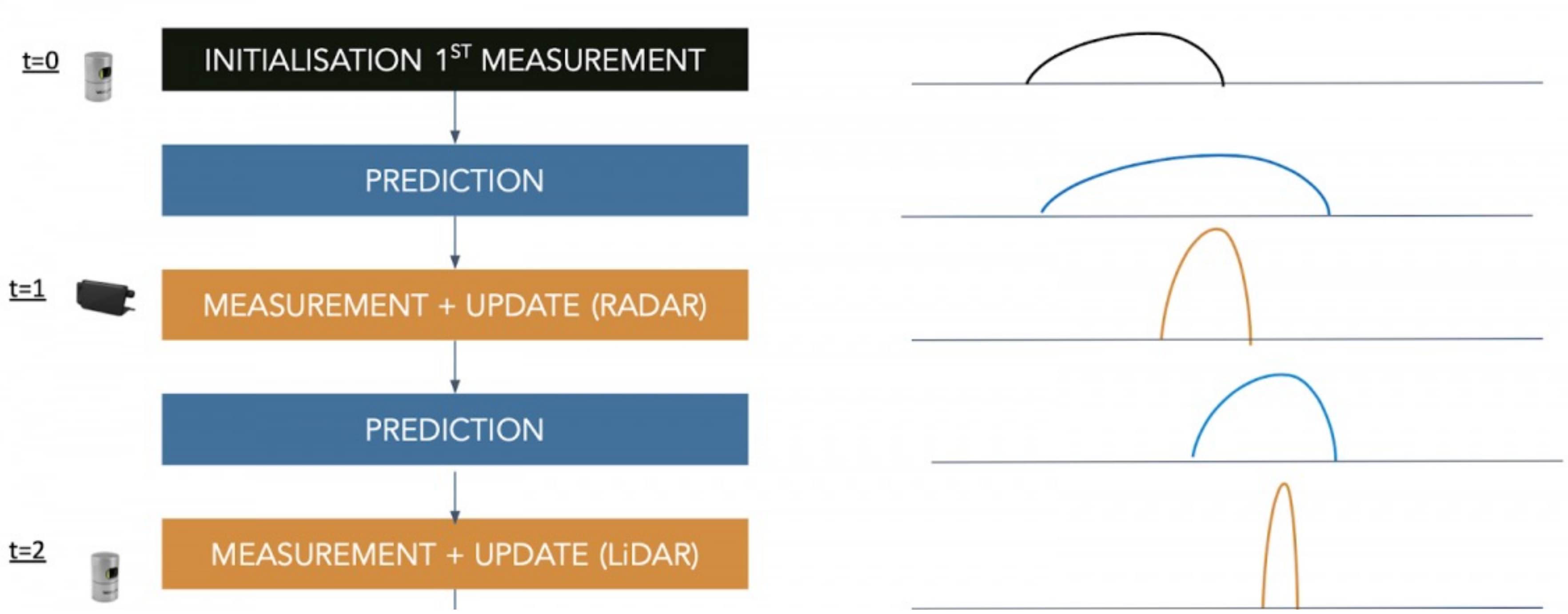
RADAR/LiDAR Sensor Fusion Flow

Each time a sensor value comes in ➔ new prediction+update cycle



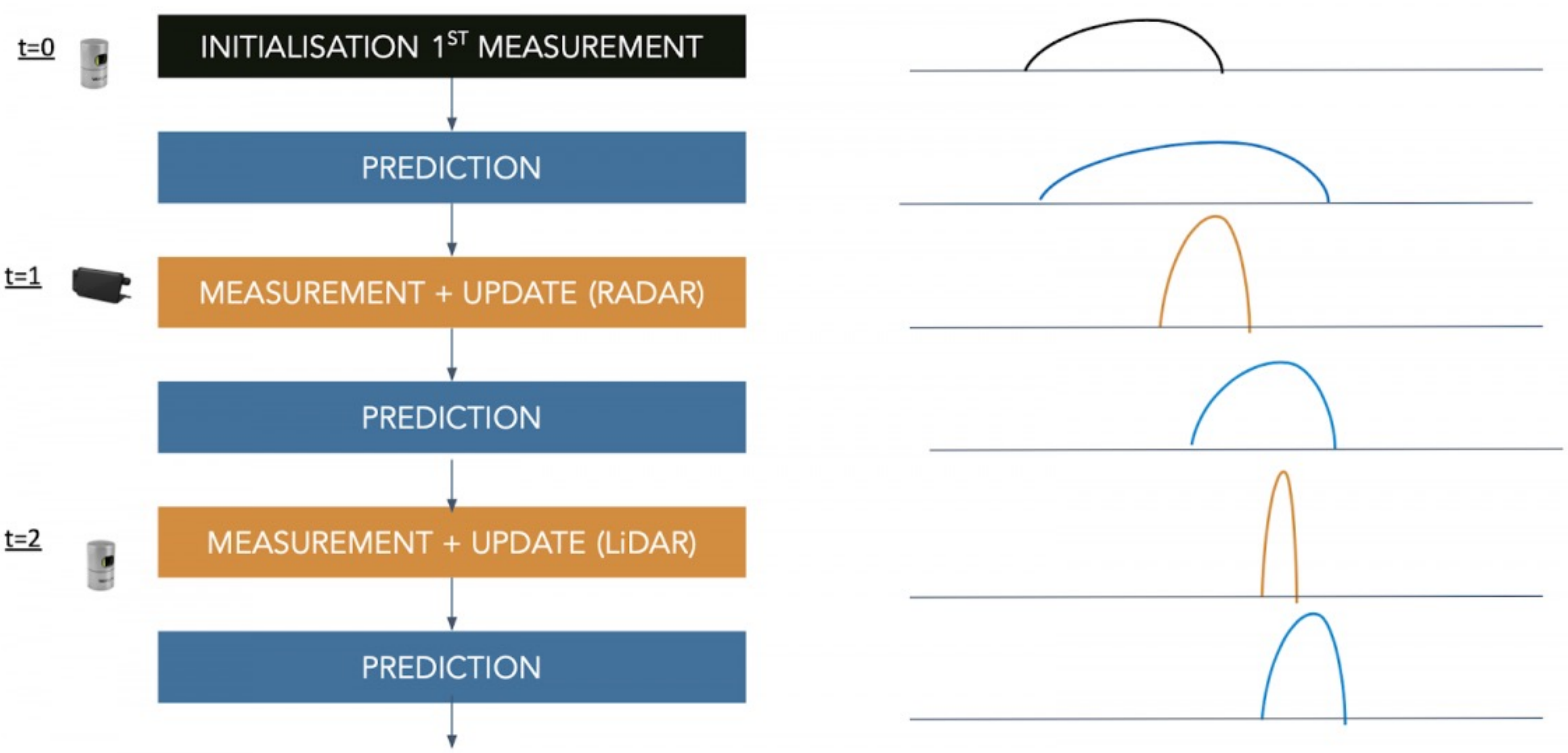
RADAR/LiDAR Sensor Fusion Flow

Each time a sensor value comes in → new prediction+update cycle



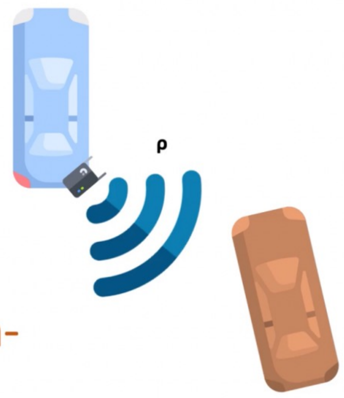
RADAR/LiDAR Sensor Fusion Flow

Each time a sensor value comes in → new prediction+update cycle



Kalman Filter | Hang on a minute…

- LiDAR has cartesian (linear) values of type y=ax+b
- RADAR → **not linear**!



Measurements are non-linear

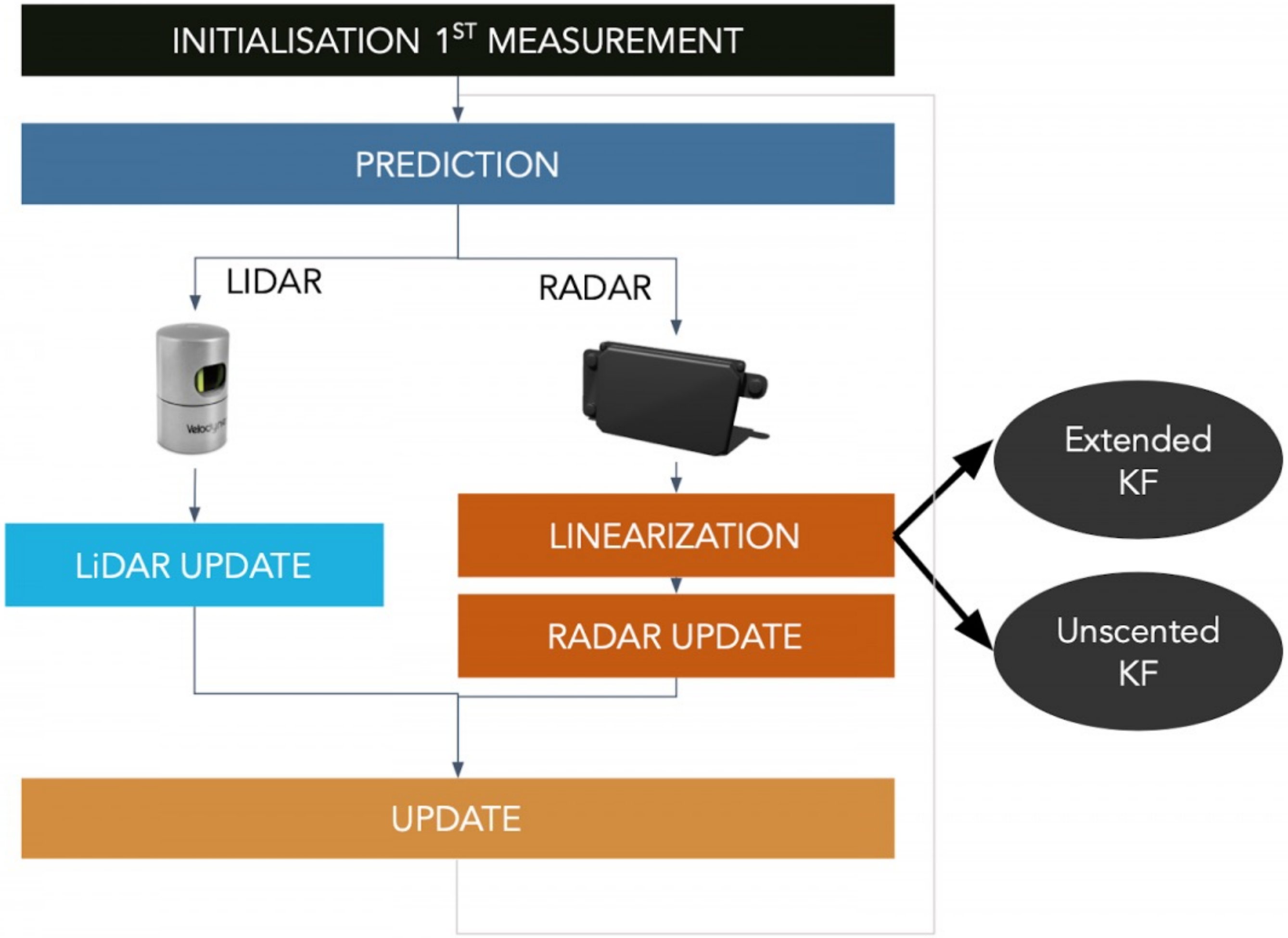
- How to reconcile the two?

Non-Linear Kalman Filters

- The world is **non-linear**
  - Not everything moves in straight lines
  - All sensors work differently

- Two types of Kalman Filters
  - Extended Kalman Filters
  - Unscented Kalman Filters

Final Sensor Fusion Flow



References

- Kalman Filter (with code):

https://towardsdatascience.com/wtf-is-sensor-fusion-part-2-the-good-old-kalman-filter-3642f321440

- A decent primer on the Kalman Filter

https://www.thinkautonomous.ai/blog/?p=sensor-fusion

- Extended Kalman Filter with a nice video

https://kusemanohar.wordpress.com/2020/04/08/sensor-fusion-extended-kalman-filter-ekf/

- Some papers that use EKF
  - https://ieeexplore.ieee.org/document/1338645
  - https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7435659/