Principles of Robot Autonomy I

Multi-sensor perception and sensor fusion

Daniel Watzenig

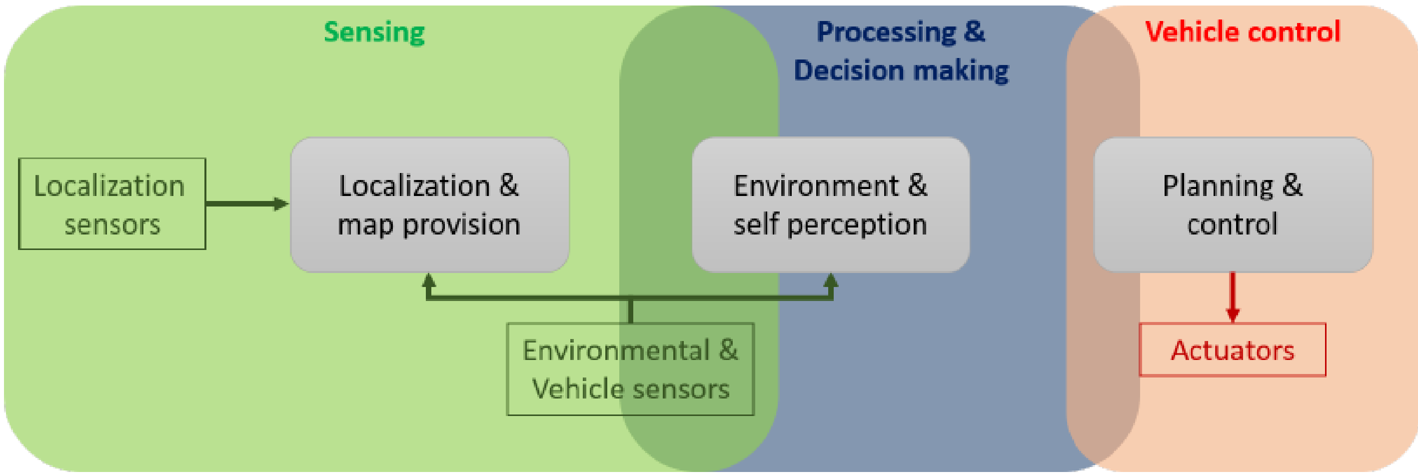
Today’s lecture

- Aim
  - Introduce the topic of multi-sensor perception and sensor fusion
  - Learn about Kalman filtering applied to sensor fusion
  - Devise a sensor fusion algorithm for position estimation (low-level fusion)

- Readings
  - F. Gustafsson. Statistical Sensor Fusion. 2010.
  - C. Lundquist, Z. Sjanic, F. Gustafsson. Statistical Sensor Fusion: Exercises. 2015.
  - D. Simon. Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches. 2006.

Multi-sensor approach in robotics

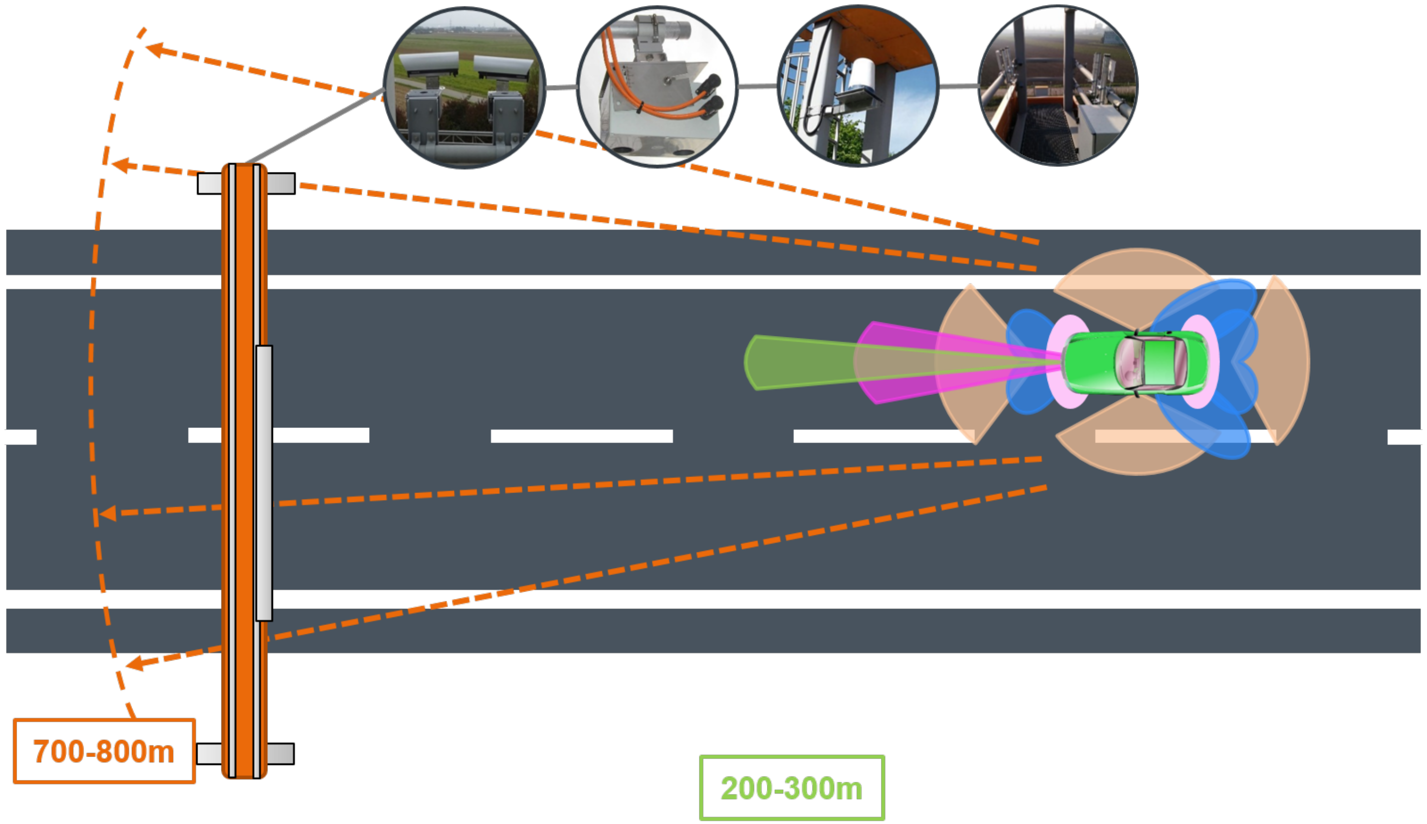
- Localization
- Environment



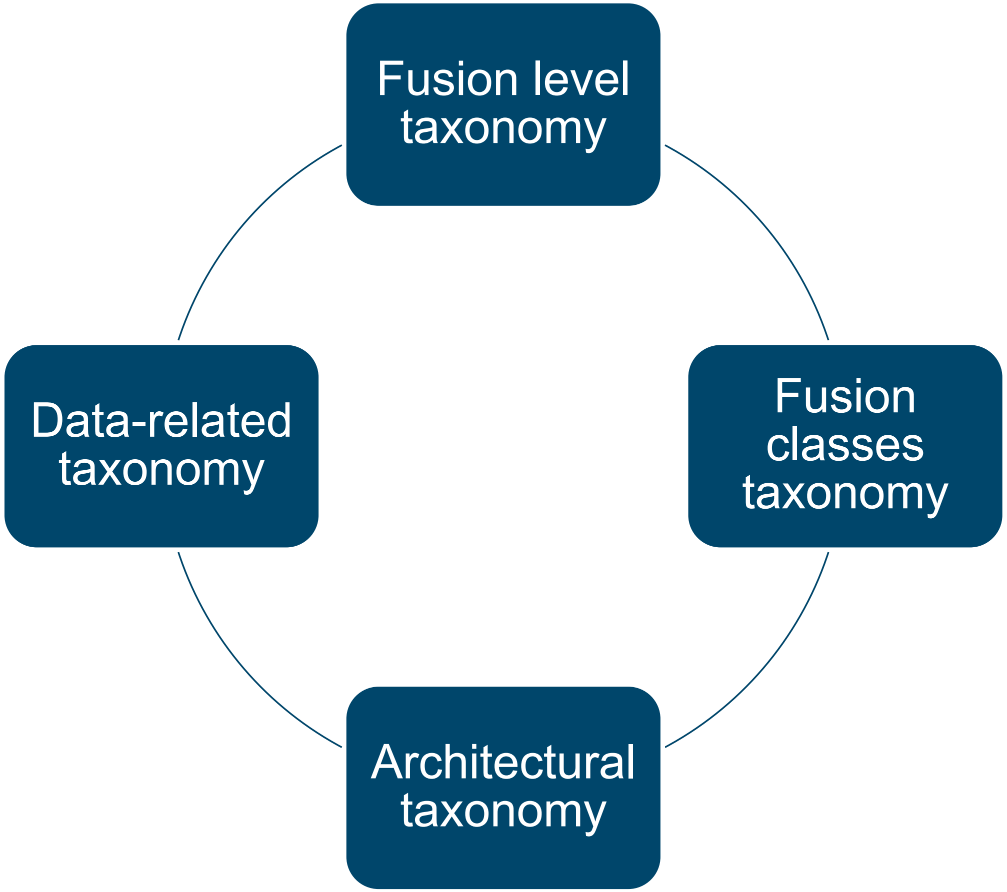
Single-sensor vs multi-sensor perception

- Drawbacks of single-sensor perception
  - Limited range and field of view
  - Performance is susceptible to common environmental conditions
  - Range determination is not as accurate as required
  - Detection of artefacts, so-called false positives
- Multi-sensor perception might compensate these, and provide:
  - Increased classification accuracy of objects
  - Improved state estimation accuracy
  - Improved robustness for instance in adverse weather conditions
  - Increased availability
  - Enlarged field of view

Using stationary sensors

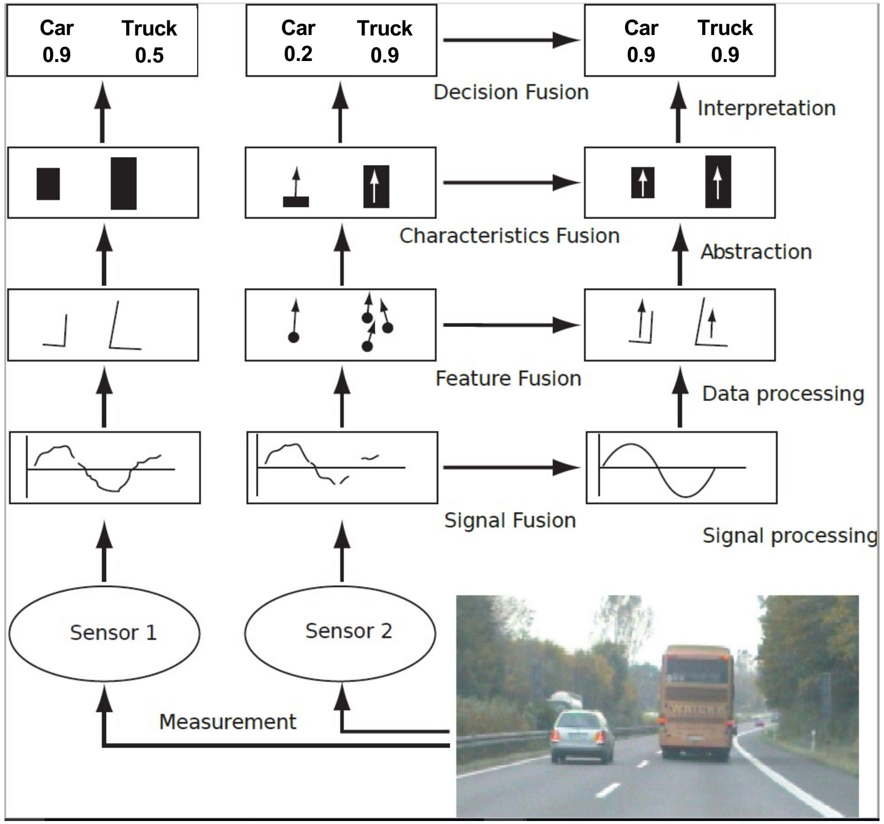


Sensor fusion taxonomies



Fusion level taxonomy

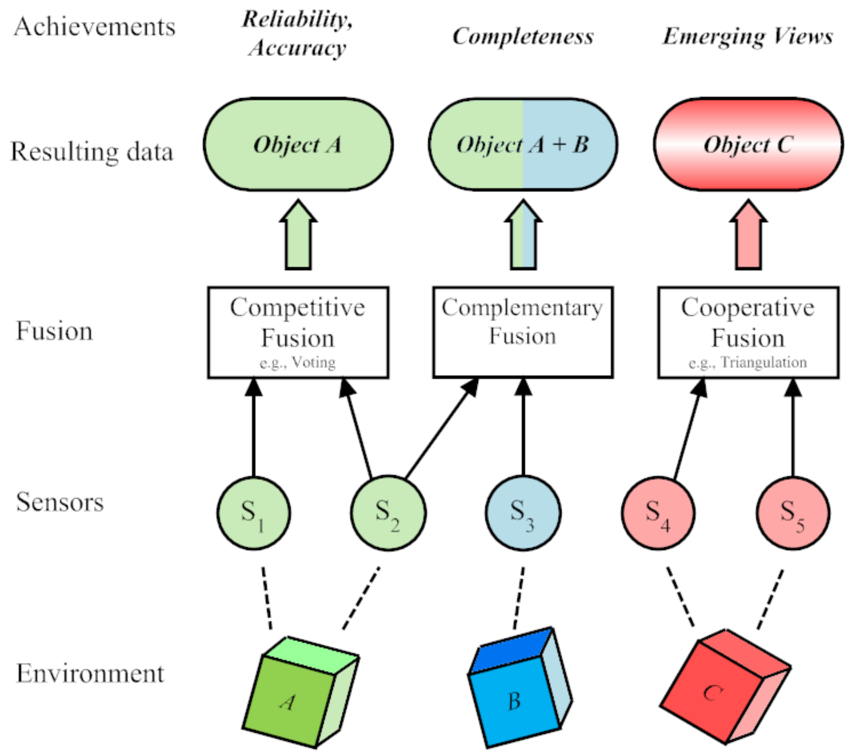
- Fusion is typically divided into three levels of abstraction:
  - Low-level fusion
  - Intermediate-level fusion
  - High-level fusion
- They respectively fuse:
  - Signals
  - Features and characteristics
  - Decisions



Schematic depiction of fusion levels (Stüker, Heterogene Sensordatenfusion zur robusten Objektverfolgung im automobilen Straßenverkehr, 2016)

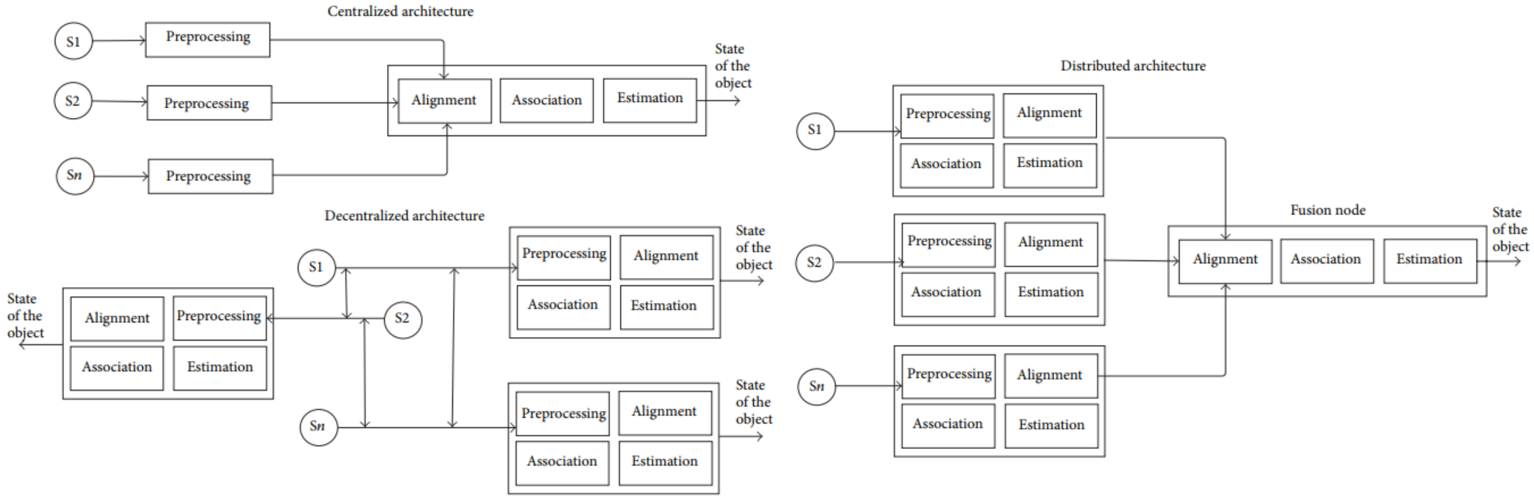
Fusion class taxonomy

- Competitive fusion
  - is used when redundant sensors measure the same quantity, in order to reduce the overall uncertainty
- Complementary fusion
  - is used when sensors provide a complementary information about the environment, for instance distance sensors with different ranges
- Cooperative fusion
  - is used when the required information can not be inferred from a single sensor (e.g. GPS localization and stereo vision)



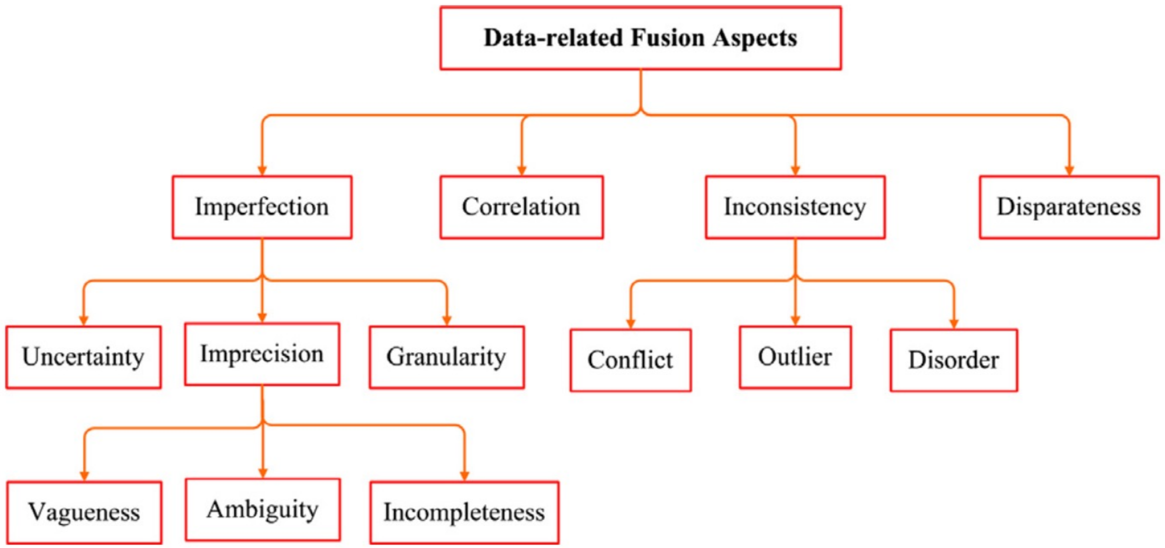
Architectural taxonomy

- The **centralized** architecture is theoretically optimal, but scales badly with respect to communication and processing
- The **decentralized** architecture is a collection of autonomous centralized systems, and has the same scaling issues
- The **distributed** architecture scales better, but can lead to information loss because each sensor processes its information locally



Data-related taxonomy

- The most interesting data-related fusion aspect is the inherent imperfection of the sensory data
- The data-related taxonomy provides us with a checklist of underlying data issues and how to deal with them



Data-related taxonomy

- Sensory data makes a statement about the environment
  - "The distance to the nearest car is 35.12 m”
- Due to the inherent data imprecision, we have to deal with:
  - **Uncertainty:** The distance to the nearest car is more than 20 m with 80% probability
  - **Vagueness:** The distance to the nearest car is more than 20 m with 80% probability, and we are 90% confident in this statement
  - **Ambiguity**
  - **Incompleteness**
- The underlying data can contain multiple imperfections at once

Bayesian statistics in multi-sensor data fusion

- **Basic premise**: all unknowns are treated as random variables and the knowledge of these quantities is summarized via a probability distribution
  - This includes the observed data, any missing data, noise, unknown parameters, and models
- Bayesian statistics provides
  - a framework for quantifying objective and subjective uncertainties
  - principled methods for model estimation and comparison and the classification of new observations
  - a natural way to combine different sensor observations
  - principle methods for dealing with missing information

Sensor fusion – a simple example

- Problem: determine the distance to n objects using measurements from two sensors
- Assumptions:
  - Both sensors have the same field of view
  - First sensor has a higher precision than the second sensor
  - Consider the simplest case (n=1)

- How to fuse these measurements properly?

Sensor fusion – a simple example

- Sensors provide redundant measurements of the same physical quantity (distance)
- To incorporate the precision information → measurements are assumed to be normally distributed random variables
- Specifically, the univariate Gaussian distributions are:

$$d_1(x) = (2\pi\sigma_1^2)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\frac{(x-\mu_1)^2}{\sigma_1^2}\right) \quad \sim \mathcal{N}(\mu_1, \sigma_1^2)$$

$$d_2(x) = (2\pi\sigma_2^2)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\frac{(x-\mu_2)^2}{\sigma_2^2}\right) \quad \sim \mathcal{N}(\mu_2, \sigma_2^2)$$

Sensor fusion – a simple example

- Assumption from before:
  - First sensor has a higher precision than the second sensor
- This can be captured as: $\sigma_1^2 < \sigma_2^2$
- Problem is to find $d(x) \sim \mathcal{N}(\mu, \sigma^2)$
- The idea is to combine the previous Gaussian distributions

$$d(x) = d_1(x) \cdot d_2(x) = (4\pi^2\sigma_1^2\sigma_2^2)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}\left(\frac{(x-\mu_1)^2}{\sigma_1^2} + \frac{(x-\mu_2)^2}{\sigma_2^2}\right)\right)$$

Sensor fusion – a simple example

- Re-arranging the expression in the exponent and dividing the numerator and denominator by $(\sigma_1^2 + \sigma_2^2)$:

$$-\frac{1}{2}\left(\frac{(x-\mu_1)^2}{\sigma_1^2}+\frac{(x-\mu_2)^2}{\sigma_2^2}\right) = -\frac{1}{2}\frac{(\sigma_1^2+\sigma_2^2)x^2-2(\sigma_2^2\mu_1+\sigma_1^2\mu_2)x+(\sigma_2^2\mu_1^2+\sigma_1^2\mu_2^2)}{\sigma_1^2\sigma_2^2}$$

$$= -\frac{1}{2}\frac{x^2-2\frac{\mu_1\sigma_2^2+\mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}x+\frac{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2}{\sigma_1^2+\sigma_2^2}}{\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2}}$$

- To obtain an expression of form $x^2 - 2\mu x + \mu^2 = (x-\mu)^2$ in the numerator, it is necessary to add and subtract the square of the second term

Sensor fusion – a simple example

$$-\frac{1}{2}\frac{x^2 - 2\frac{\mu_1\sigma_2^2+\mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}x + \left(\frac{\mu_1\sigma_2^2+\mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)^2 - \left(\frac{\mu_1\sigma_2^2+\mu_2\sigma_1^2}{\sigma_1^2+\sigma_2^2}\right)^2 + \frac{\mu_1^2\sigma_2^2+\mu_2^2\sigma_1^2}{\sigma_1^2+\sigma_2^2}}{\frac{\sigma_1^2\sigma_2^2}{\sigma_1^2+\sigma_2^2}}$$

- The expression in the exponent becomes

$$-\frac{1}{2}\frac{(x-\mu)^2 - \mu^2 + s}{\sigma^2} = -\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2} + \frac{\mu^2 - s}{2\sigma^2}$$

Sensor fusion – a simple example

- Putting everything together leads to the final distribution which represents the fused information

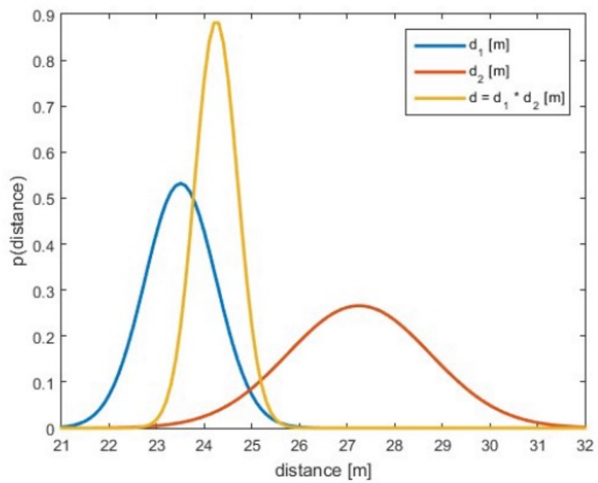
$$
\begin{aligned}
d(x) &= (2\pi\sigma_1\sigma_2)^{-1} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2} + \frac{\mu^2 - s}{2\sigma^2}\right) \\
&= (2\pi\sigma_1\sigma_2)^{-1} \exp\left(\frac{\mu^2 - s}{2\sigma^2}\right) \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right) \\
&= C \cdot \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right)
\end{aligned}
$$

Sensor fusion – a simple example

- Mean value and variance are

$$\mu = \frac{\mu_1 \sigma_2^2 + \mu_2 \sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

$$\sigma^2 = \frac{\sigma_1^2 \cdot \sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$



- The fused value is the weighted average of the measurements
- The weighting favors the sensor with higher precision
- The overall uncertainty decreases

Kalman filter (KF) – again

- Assumption #1: linear dynamics $x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$
  - i.i.d. process noise ϵ_t is $\mathcal{N}(0, R_t)$
  - Assumption #1 implies that the probabilistic generative model is Gaussian
- Assumption #2: linear measurement model $z_t = C_t x_t + \delta_t$
  - i.i.d. measurement noise δ_t is $\mathcal{N}(0, Q_t)$
  - Assumption #2 implies that the measurement probability is Gaussian

Kalman filter (KF)

- Assumption #3: the initial belief is Gaussian

$$bel(x_0) = p(x_0) = \det(2\pi\Sigma_0)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x_0 - \mu_0)^T \Sigma_0^{-1} (x_0 - \mu_0)\right)$$

- Key fact: These three assumptions ensure that the posterior $bel(x_t)$ is Gaussian for all *t*, i.e., $bel(x_t) = \mathcal{N}(\mu_t, \Sigma_t)$
- Note:
  - KF implements a belief computation for continuous states
  - Gaussians are unimodal → commitment to single-hypothesis filtering

Kalman filter: algorithm revisited

Prediction

Project state ahead

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$

Project covariance ahead

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

Uncertainty in prediction

Correction

Compute Kalman gain

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

Update estimate with new measurement

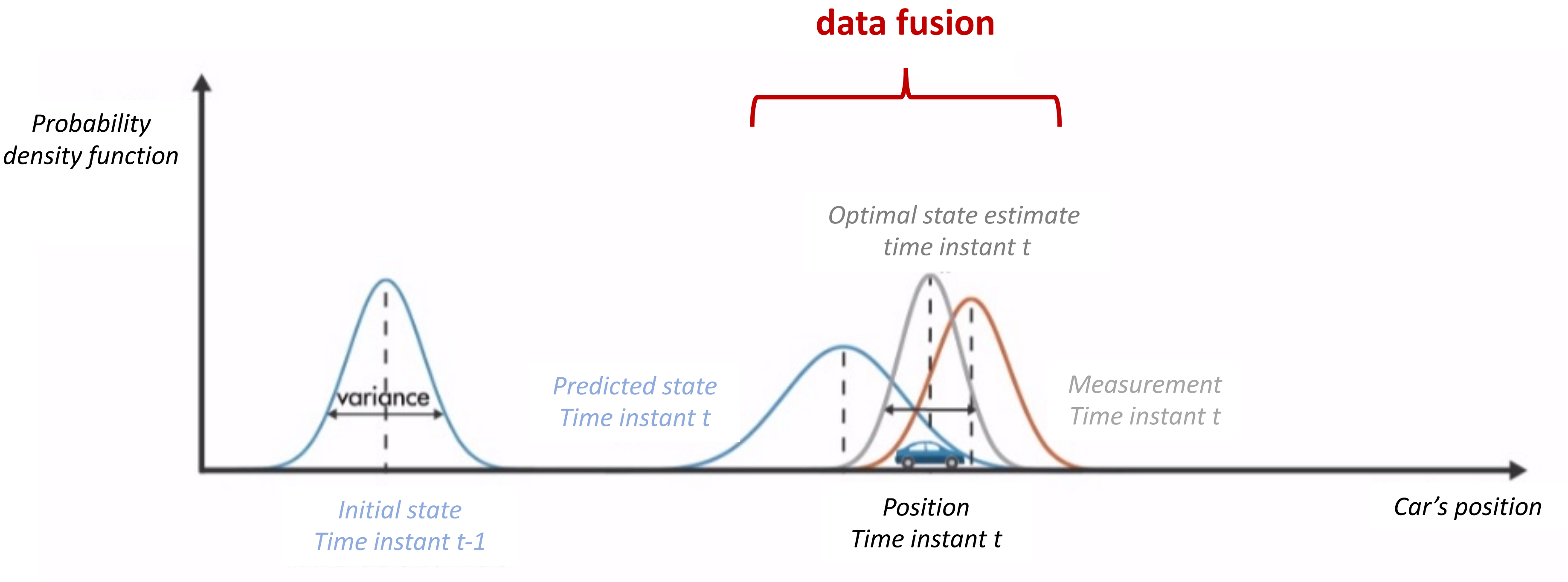
$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t)$$

Update covariance

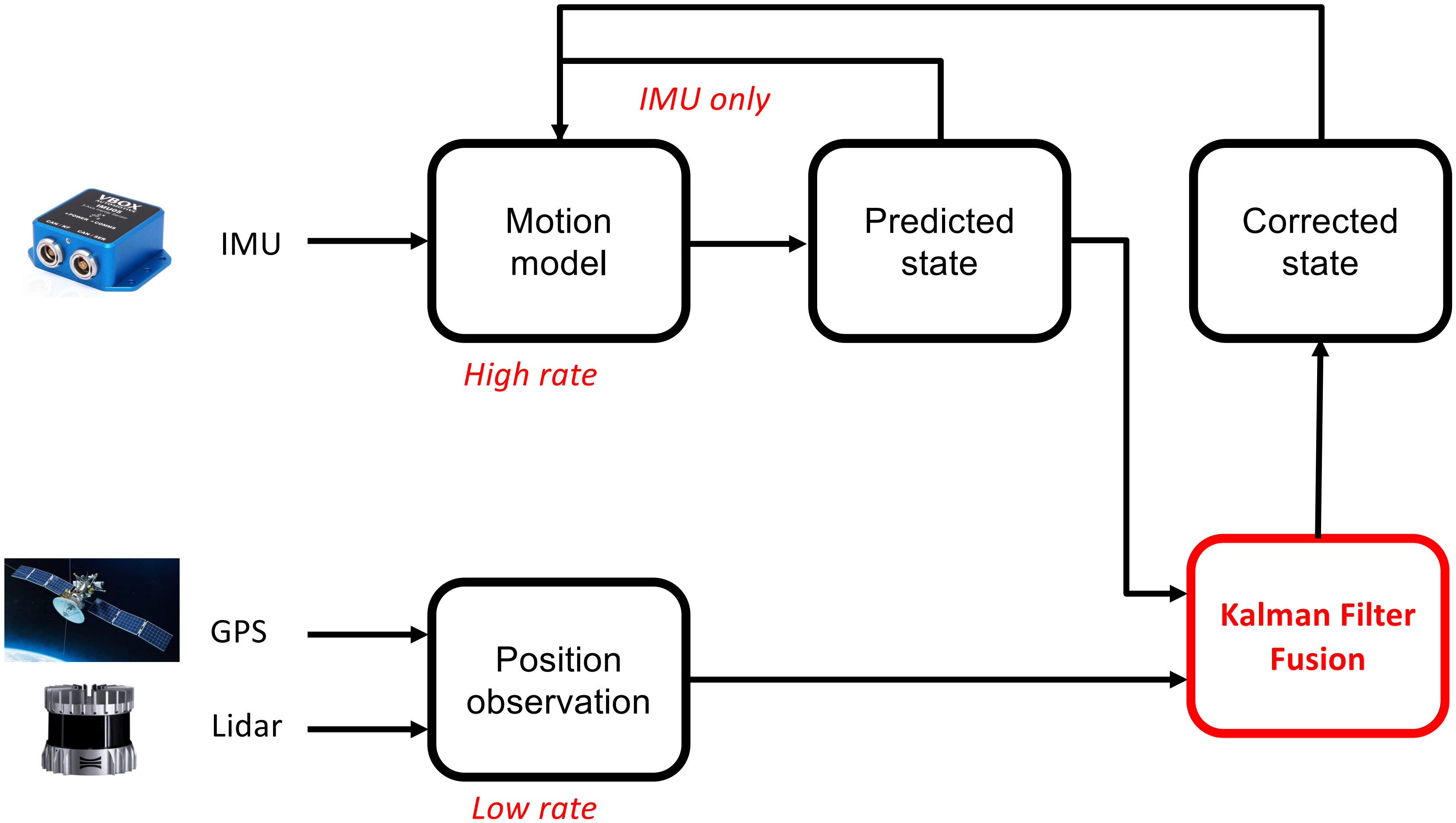
$$\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$$

Uncertainty in correction

Kalman filter (KF) – pose estimation

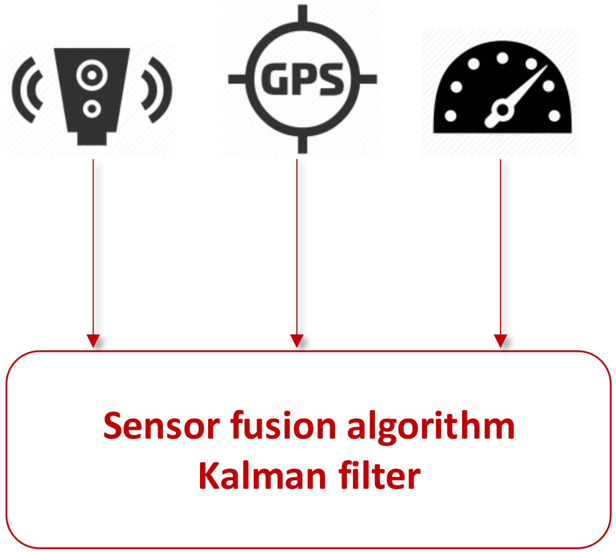


Kalman filter (KF) – multi-sensor fusion



Sensor fusion example

- Problem: Estimate position, velocity, and acceleration of a vehicle from noisy position and acceleration measurements
- Assumptions:
  - Single track model for the vehicle
  - Lidar provides position measurements with low precision
  - GPS provides position measurements with high precision
  - IMU provides acceleration measurements
- Sensor fusion is done using the Kalman filter



Sensor fusion example: Motion model

- State vector: $\mu_t = \begin{bmatrix} p & v & a \end{bmatrix}^T$
- Change of the state over time is captured by the motion model

$$
\begin{aligned}
p_t &= p_{t-1} + T_s v_{t-1} + \frac{T_s^2}{2} a_{t-1} + \epsilon_{pt} \\
v_t &= v_{t-1} + T_s a_{t-1} + \epsilon_{vt} \\
a_t &= a_{t-1} + \epsilon_{at}
\end{aligned}
$$

- T_s represents sampling time

Sensor fusion example: Motion model

- The motion model can be represented in matrix form

$$\underbrace{\begin{bmatrix} p \\ v \\ a \end{bmatrix}_t}_{\text{State vector}} = \underbrace{\begin{bmatrix} 1 & T_s & \frac{T_s^2}{2} \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix}}_{\text{State transition matrix}} \begin{bmatrix} p \\ v \\ a \end{bmatrix}_{t-1} + \underbrace{\begin{bmatrix} \epsilon_p \\ \epsilon_v \\ \epsilon_a \end{bmatrix}_t}_{\text{Process noise}}$$

State vector | *State transition matrix* | *Process noise*

$$\mu = A_t \mu_{t-1} + \epsilon_t$$

where ϵ_t is i.i.d. process noise distributed as $\mathcal{N}(0, R_t)$

Sensor fusion example: Measurement model

- The measurement model defines a mapping from the state space to the measurement space
- For this example, two possible fusion scenarios will be considered:
  1. Lidar + IMU
  2. Lidar + GPS + IMU
- In the first scenario, only measurements from Lidar and IMU are available
  - Assumption: Lidar provides low precision measurements (noisy data)
- In the second scenario, high precision GPS measurements are also available

Sensor fusion example: Measurement model

- First scenario – measurement model is given by

$$\underbrace{\begin{bmatrix} p_{lidar} \\ a_{imu} \end{bmatrix}_t}_{\text{Measurement vector}} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Measurement matrix}} \underbrace{\begin{bmatrix} p \\ v \\ a \end{bmatrix}_t}_{\text{State vector}} + \underbrace{\begin{bmatrix} \delta_{lidar} \\ \delta_{imu} \end{bmatrix}_t}_{\text{Measurement noise}}$$

$$z_t = C_t \mu_t + \delta_t$$

where δ_t is i.i.d. measurement noise distributed as $\mathcal{N}(0, Q_t)$

Sensor fusion example: Initialization

- Choosing the initial state vector μ_0 - depends on available information
  - If there is *a-priori* knowledge – initialization is done with known values
  - If there is a lack of information – initial state is chosen to be zero
  - For this example the initial state vector is set to zero

$$\mu_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

- Choosing the initial covariance matrix Σ_0 - should be defined based on the initialization error
  - If the initial state is not very close to the correct state - Σ_0 will have large values
  - If the initial state is close to the correct state - Σ_0 will have small values

$$\Sigma_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Sensor fusion example: Noise model tuning

- The process noise covariance matrix R_t - describes the confidence in the system model
  - Small values indicate higher confidence – predicted values are more weighted
  - Large values indicate lower confidence – measurements become dominant
- The measurement noise covariance matrix Q_t - describes the confidence in the measurements
  - Has a similar interpretation as R_t
- Both matrices need to be symmetric and positive definite

$$R_t = \begin{bmatrix} 0.05 & 0 & 0 \\ 0 & 0.001 & 0 \\ 0 & 0 & 0.05 \end{bmatrix} \qquad Q_t = \begin{bmatrix} \sigma_{lidar}^2 & 0 \\ 0 & \sigma_{imu}^2 \end{bmatrix}$$

Sensor fusion example: Algorithm

- Estimation results are obtained using the prediction-correction scheme

Prediction

Project state ahead

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$

Project covariance ahead

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

Correction

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

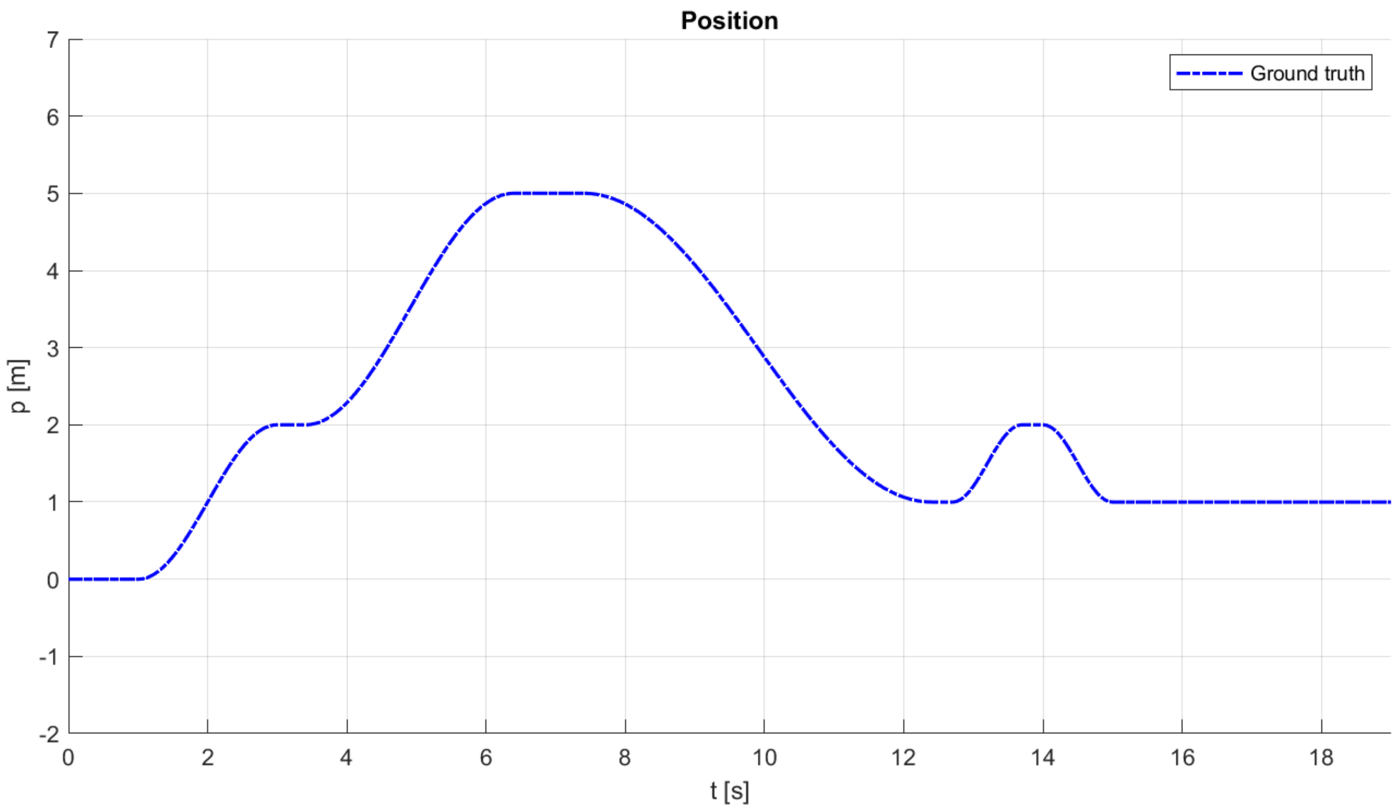
Update estimate with new measurements

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

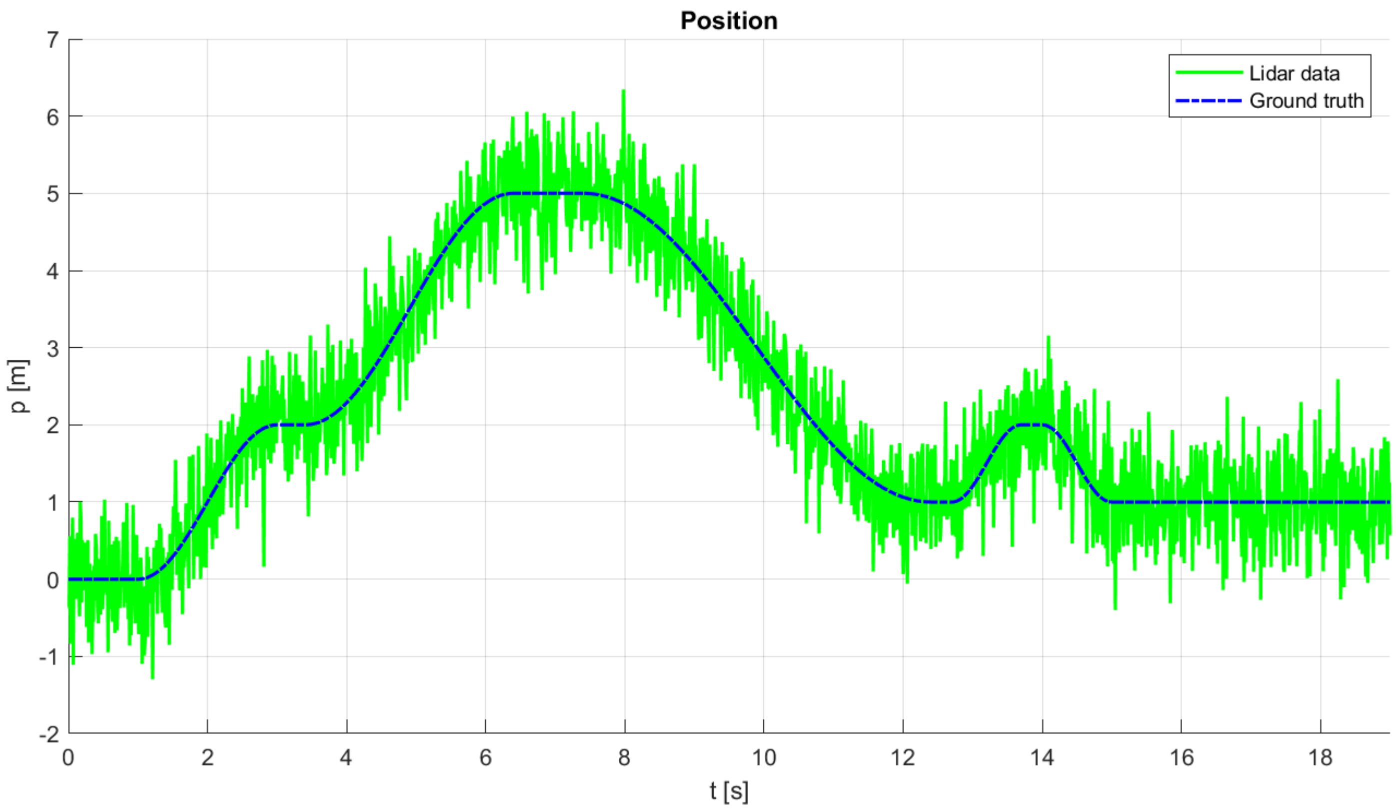
Update covariance

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

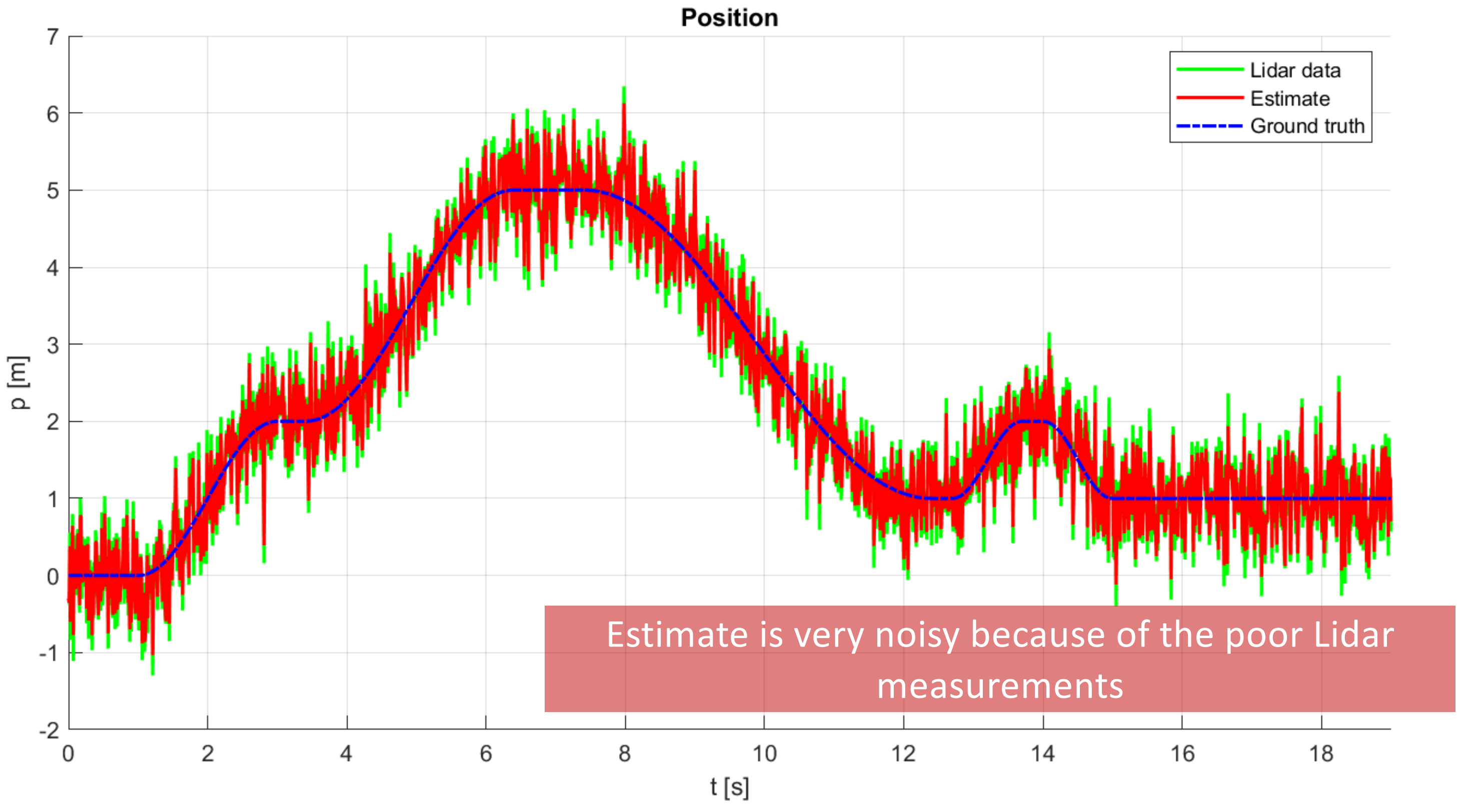
Sensor fusion example: Position estimation



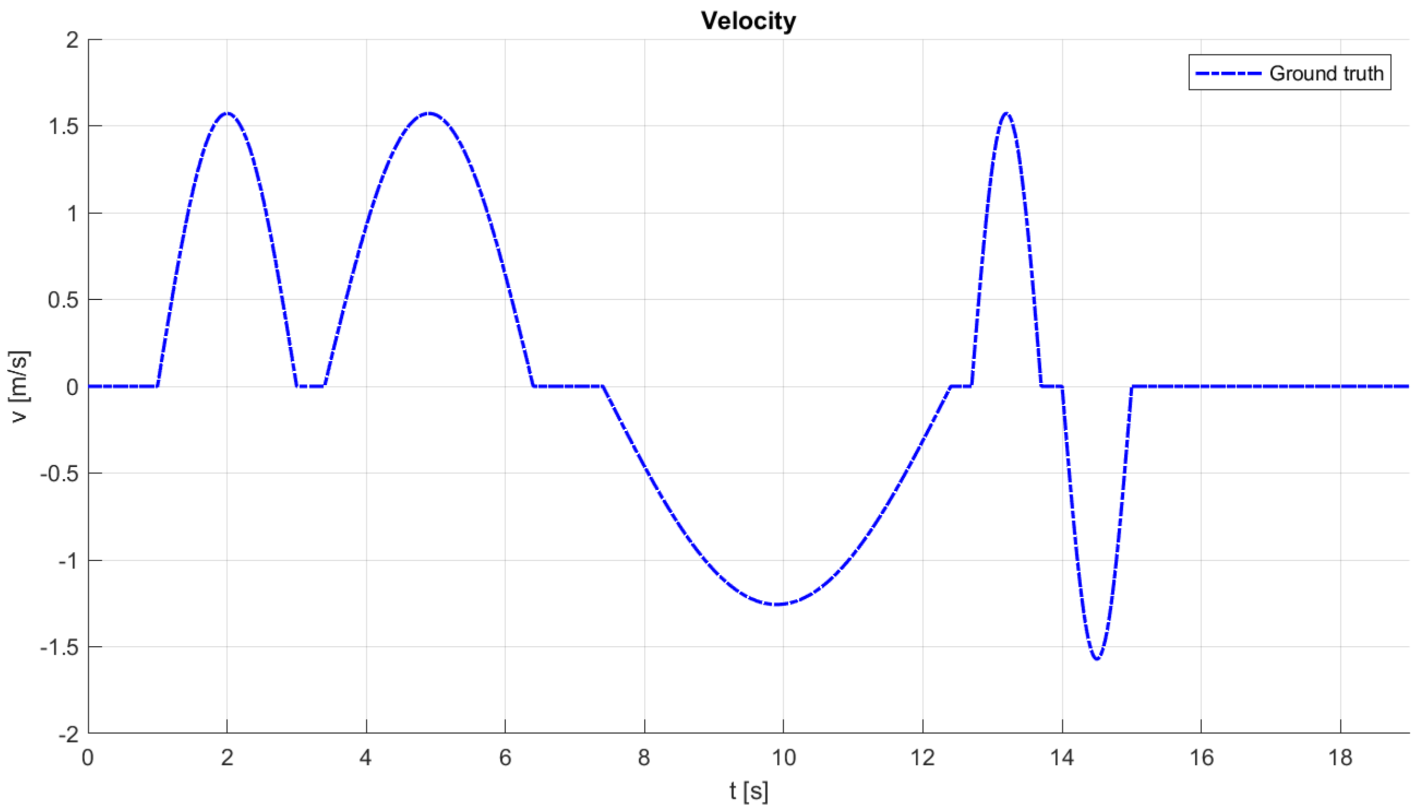
Sensor fusion example: Position estimation



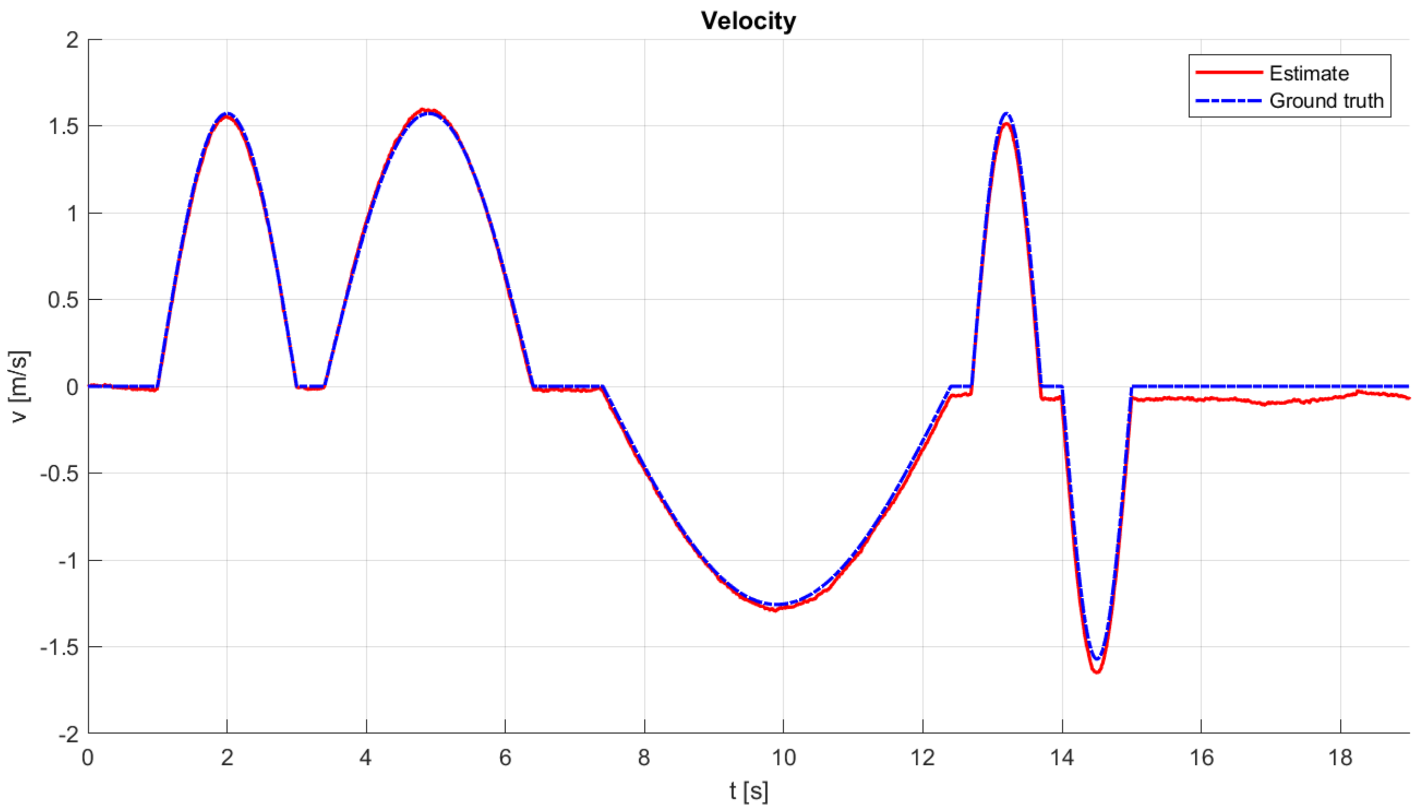
Sensor fusion example: Position estimation



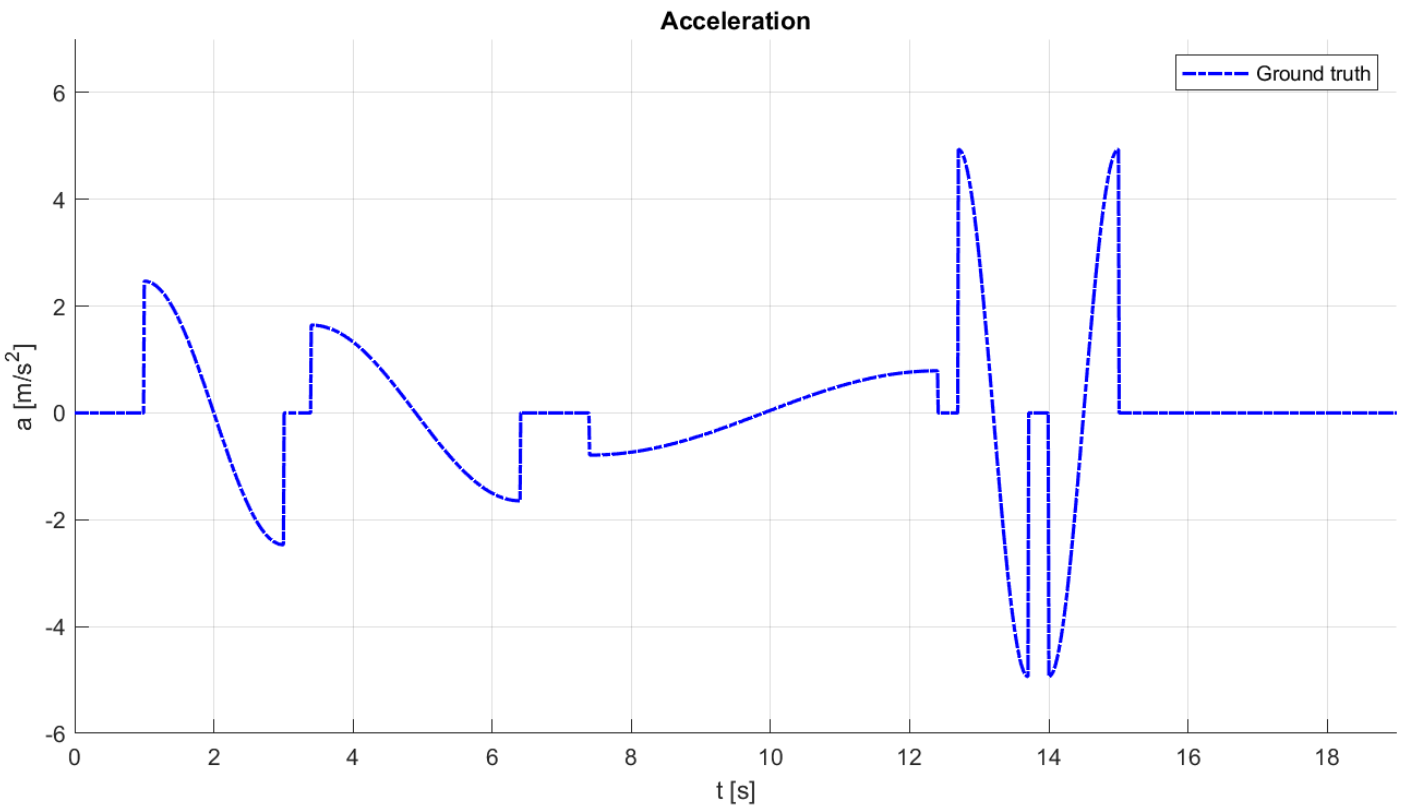
Sensor fusion example: Velocity estimation



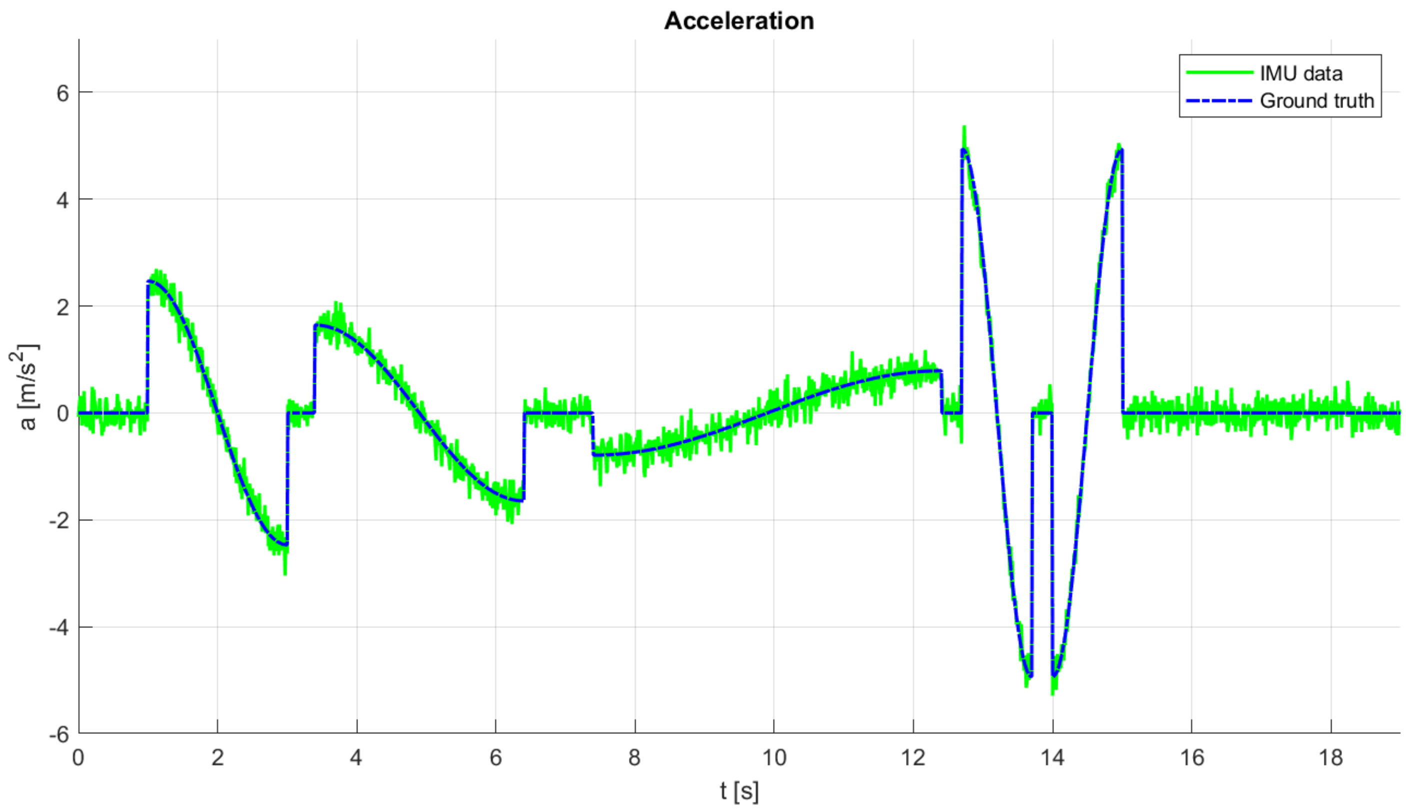
Sensor fusion example: Velocity estimation



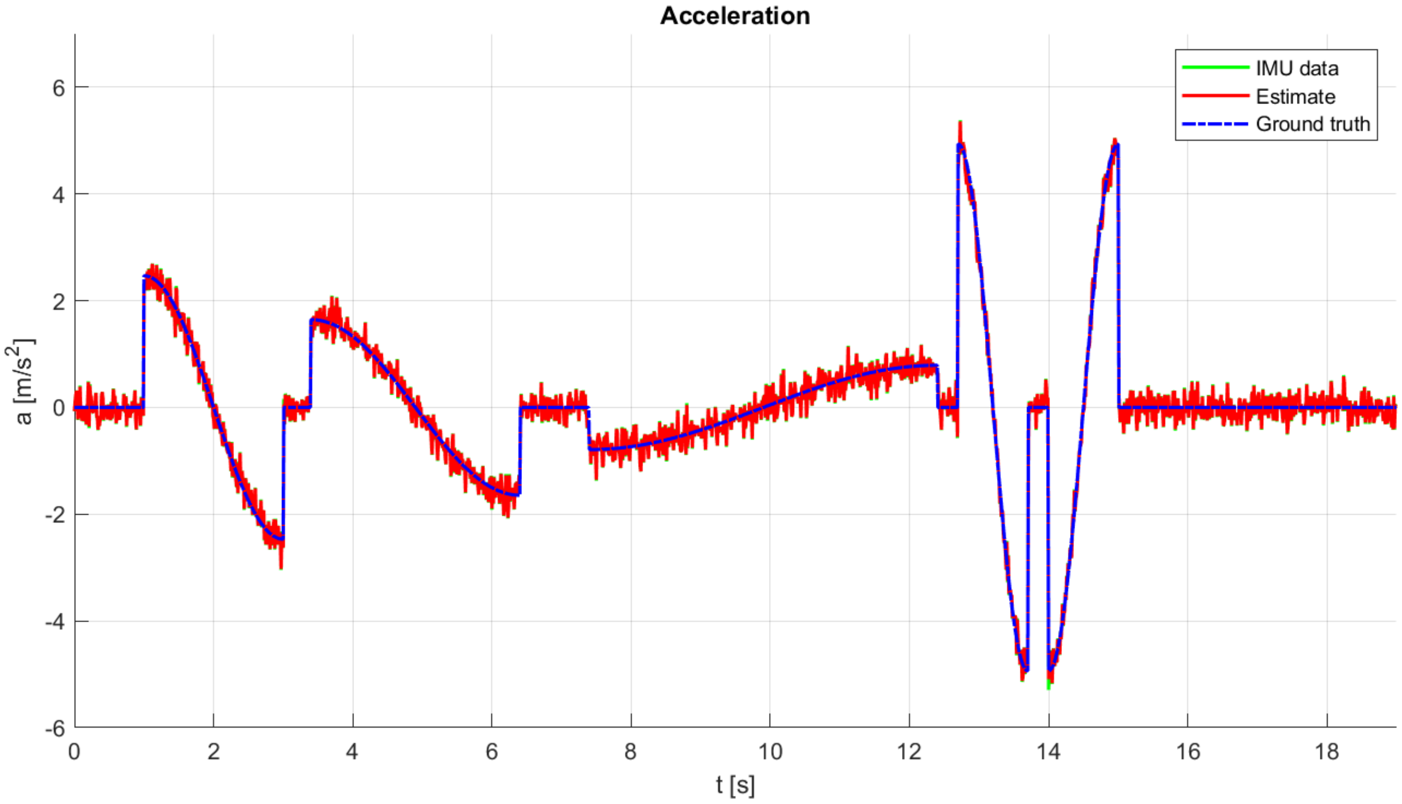
Sensor fusion example: Acceleration estimation



Sensor fusion example: Acceleration estimation



Sensor fusion example: Acceleration estimation



Sensor fusion example: New measurement model

- In the previous scenario – the position estimate is quite noisy (because of the low precision of the Lidar measurements)
- Now: position is measured with Lidar <u>and</u> GPS

$$\begin{bmatrix} p_{lidar} \\ p_{gps} \\ a_{imu} \end{bmatrix}_t = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ v \\ a \end{bmatrix}_t + \begin{bmatrix} \delta_{lidar} \\ \delta_{gps} \\ \delta_{imu} \end{bmatrix}_t$$

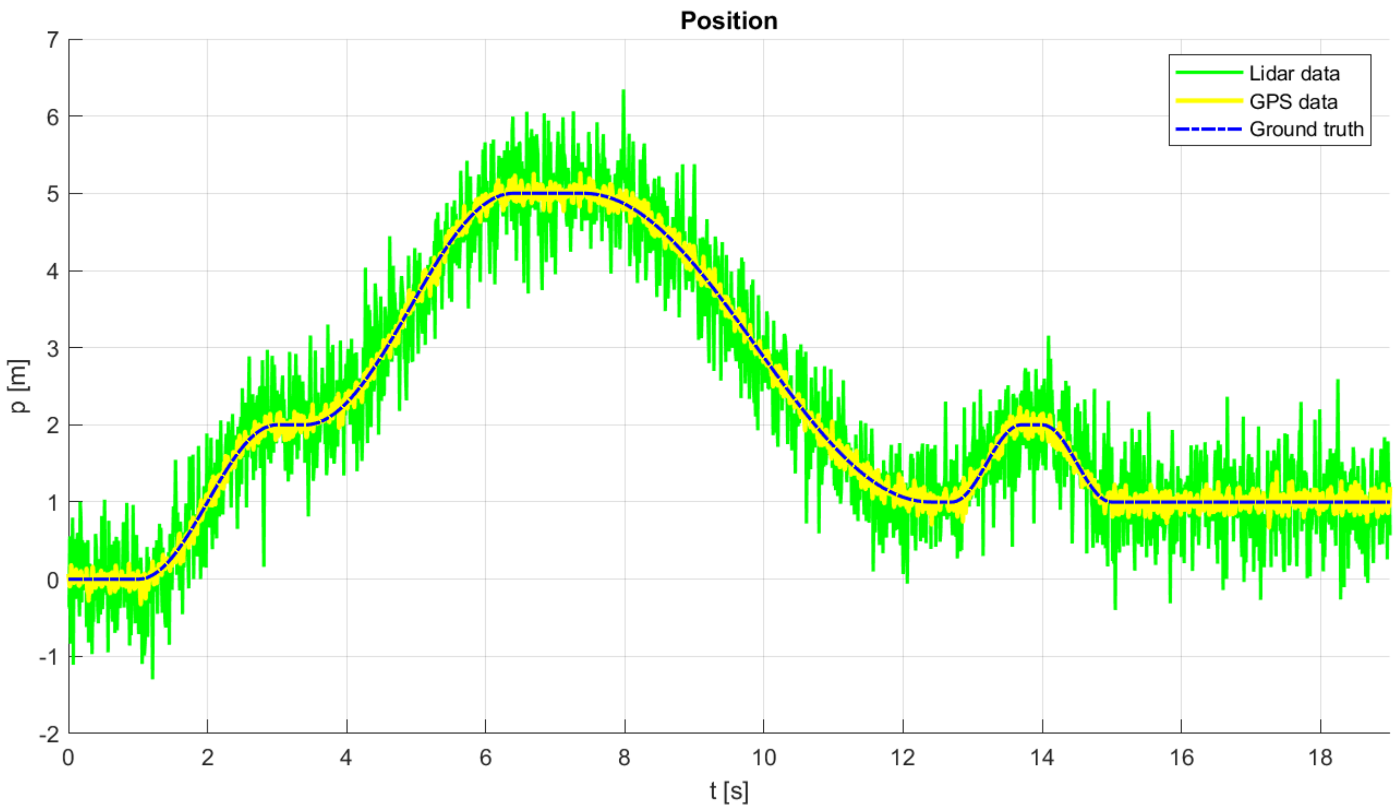
$$z_t = C_t \mu_t + \delta_t$$

Sensor fusion example: Noise model tuning

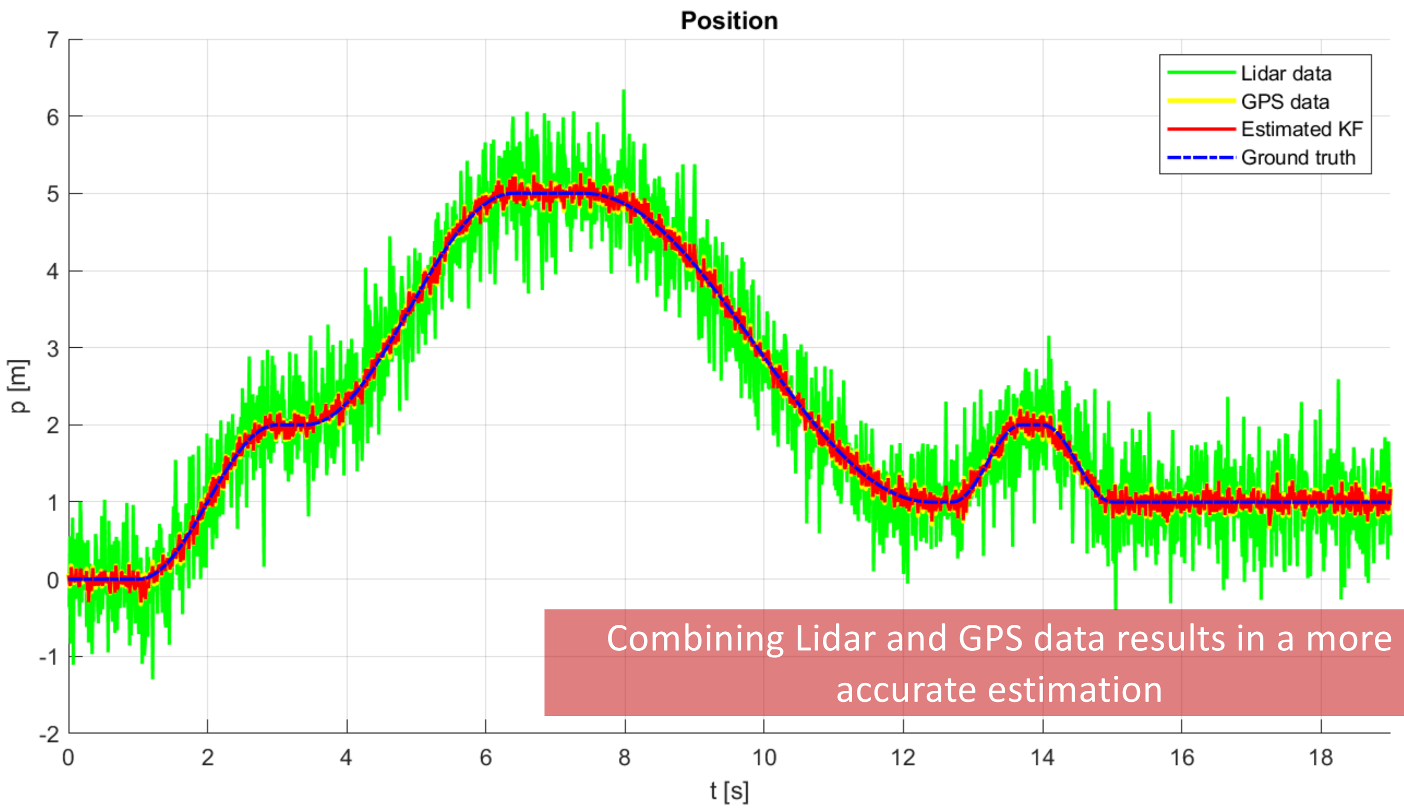
- The measurement noise covariance matrix Q_t for this scenario has an additional GPS variance

$$Q_t = \begin{bmatrix} \sigma_{lidar}^2 & 0 & 0 \\ 0 & \sigma_{gps}^2 & 0 \\ 0 & 0 & \sigma_{imu}^2 \end{bmatrix} = \begin{bmatrix} 0.5^2 & 0 & 0 \\ 0 & 0.1^2 & 0 \\ 0 & 0 & 0.2^2 \end{bmatrix}$$

Sensor fusion example: Position estimation

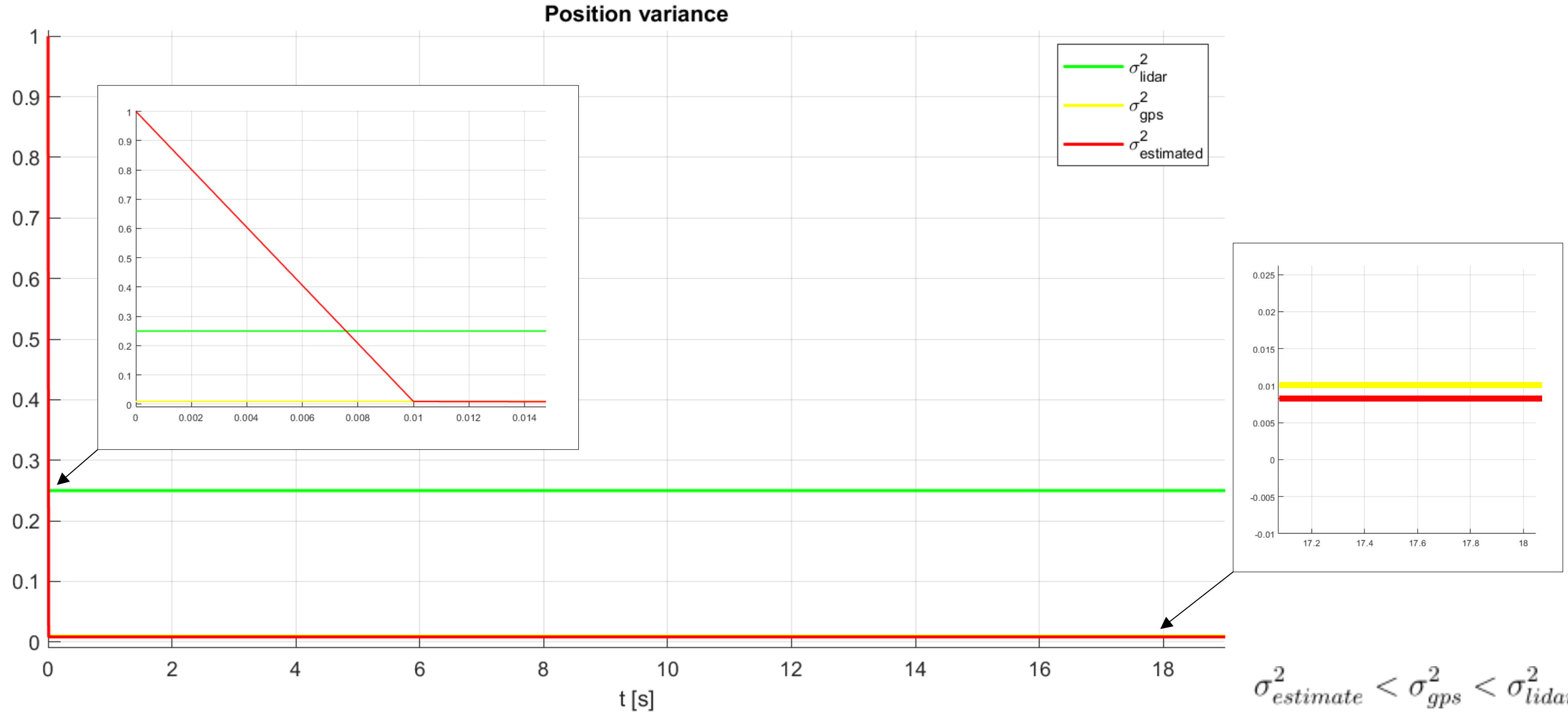


Sensor fusion example: Position estimation



Combining Lidar and GPS data results in a more accurate estimation

Sensor fusion example: Position variance



$$\sigma^2_{estimate} < \sigma^2_{gps} < \sigma^2_{lidar}$$

Sensor fusion example: Conclusion

- Problem: Vehicle state estimation using EKF
- The example pointed out:
  - How to create a motion model and a measurement model
  - How to fuse the data from different types of sensors
  - How to set the initial state vector and the initial covariance matrix
  - How to chose appropriate values for process noise and measurement noise covariance matrices
  - How to achieve a more accurate state estimation by adding more sensors
  - How fusion of data decreases the overall estimation variance

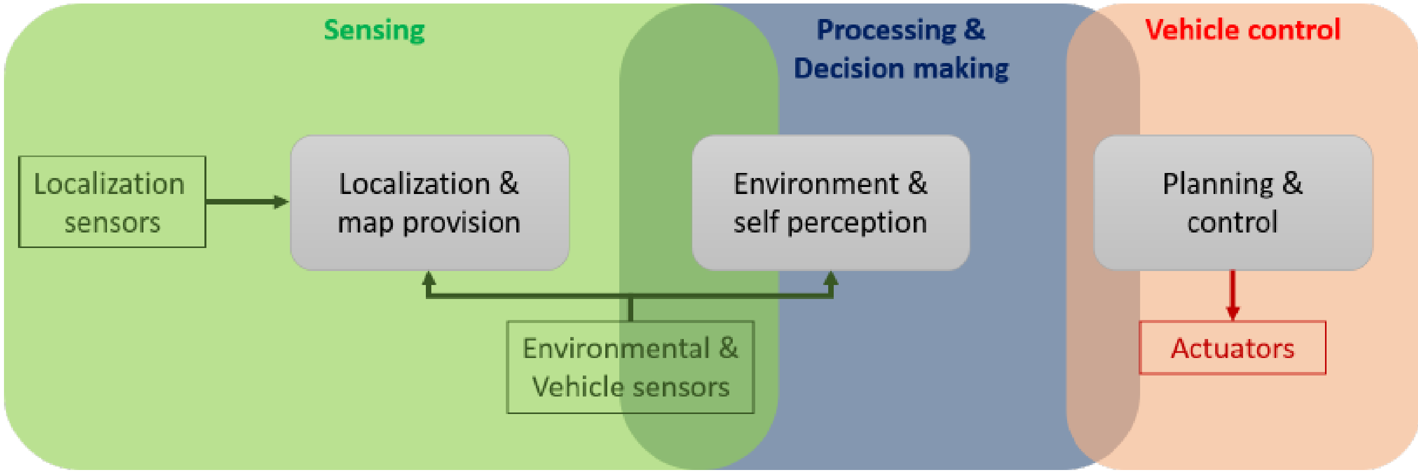
Useful trick

- Augment the state vector with some auxiliary states and then apply the KF to the augmented state space model

- What can we handle?
  - Colored state noise
  - Colored measurement noise
  - Sensor offset and drifts
  - Sensor faults (sudden offset)
  - Actuator fault (sudden offset)

Multi-sensor approach in robotics

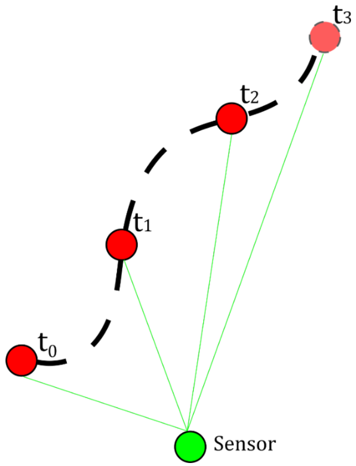
- Localization
- **Environment**

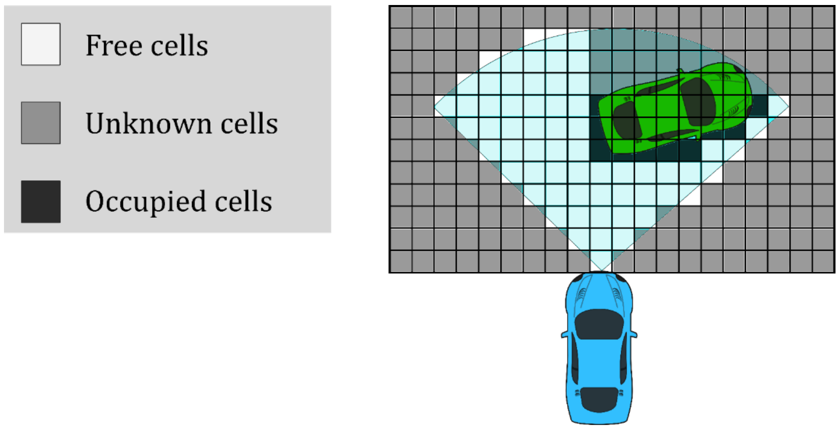


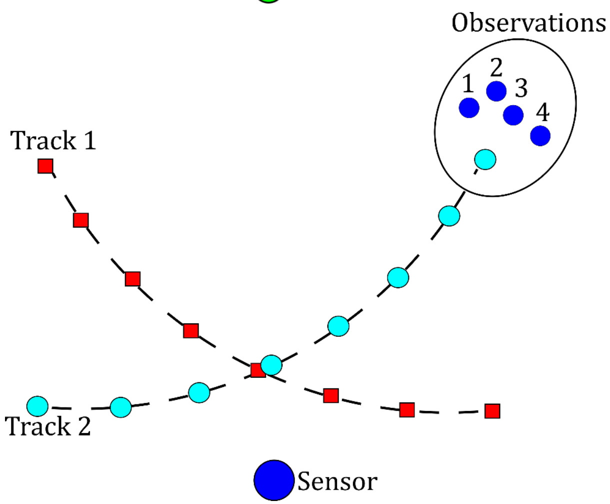
Modeling the environment

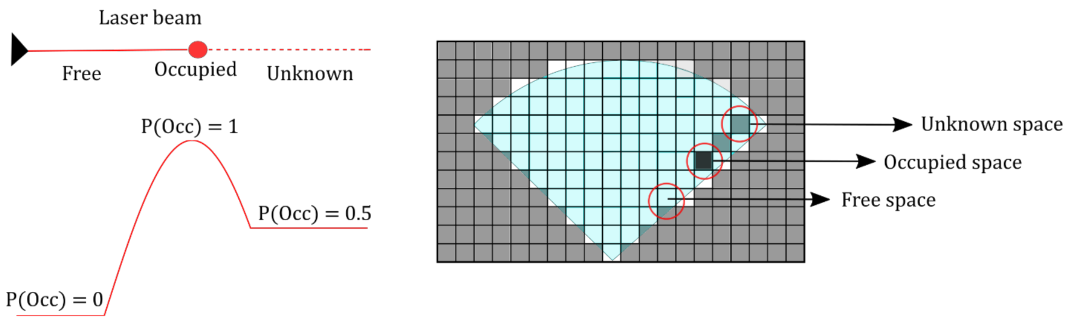
- Two types of algorithms are typically used (multiple sensors)
  - Object tracking algorithms
  - Occupancy grid algorithms
- Goal of object tracking algorithms
  - to determine the list of objects, which are currently present in the environment
  - to estimate their state variables
- Occupany grid approach
  - we describe the environment in a form of a discrete grid with certain height and width of the cells (fixed resolution step size)
  - each cell has a probability that it is occupied (or not), defined by sensor observations

Tracking vs. occupancy grids



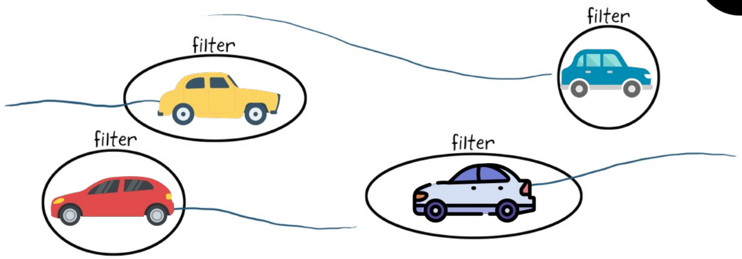






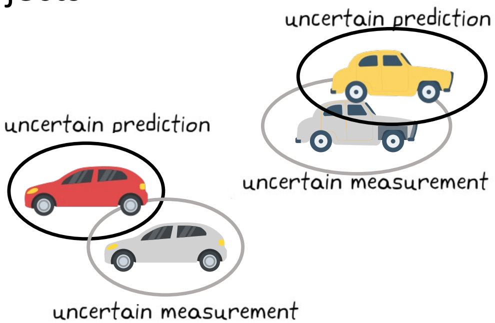
Multi-object tracking

Is the tracking of multiple objects at the same time tougher than tracking a single object?

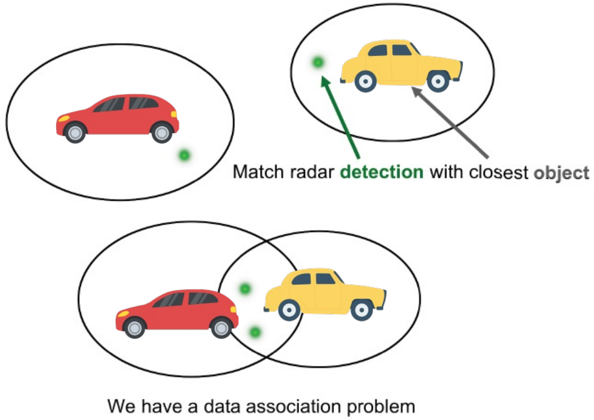


The difficulty of multi-object tracking

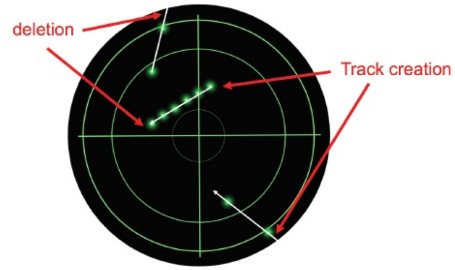
1. Uncertainties in predictions and in measurements of the objects



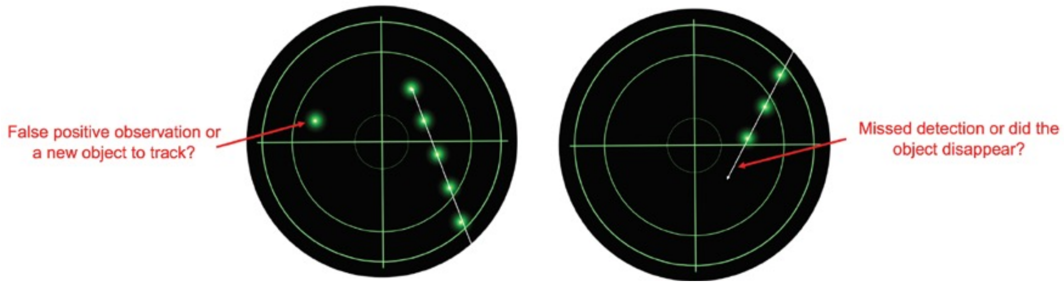
2. Data association problem



3. Track maintenance



4. Track maintenance due to uncertainties



Multi-object tracking

When tracking multiple objects:

- What are the ways to approach the data association problem?
- What are the ways to address the track maintenance problem?

Multi-object tracking flow chart

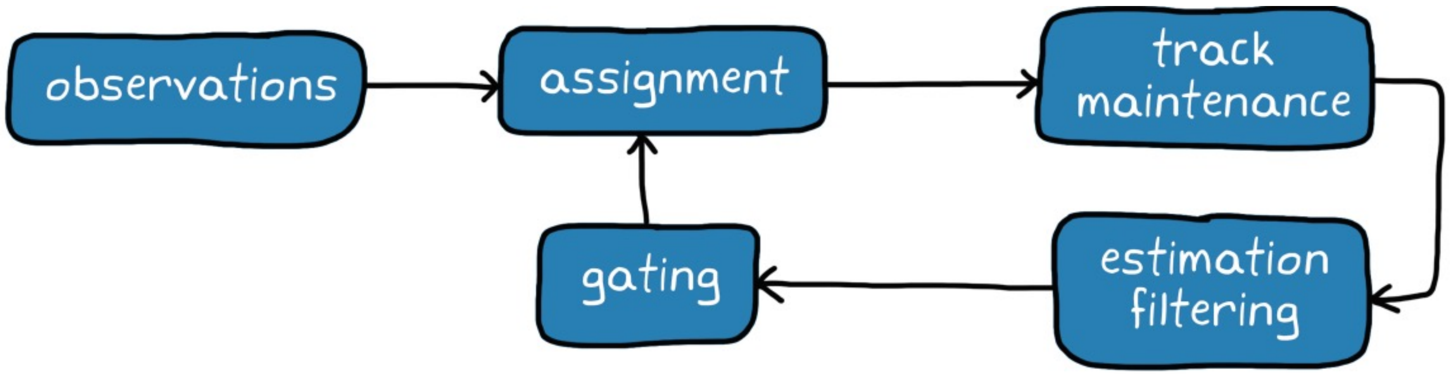


Figure adapted from Design and Analysis of Modern Tracking Systems by Samuel Blackman and Robert Popoli (Artech House Radar Library).

Multi-object tracking using EKF

Recall: Kalman Filter from previous lectures

Description of the system and the measurement models:

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$$

$$z_t = C_t x_t + \delta_t$$

- Process noise ϵ_t is $\mathcal{N}(0, R_t)$
- Measurement noise δ_t is $\mathcal{N}(0, Q_t)$

Kalman filter equations

Prediction

Project state ahead

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$

Project covariance ahead

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

Correction

$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

Update estimate with new measurement

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

Update covariance

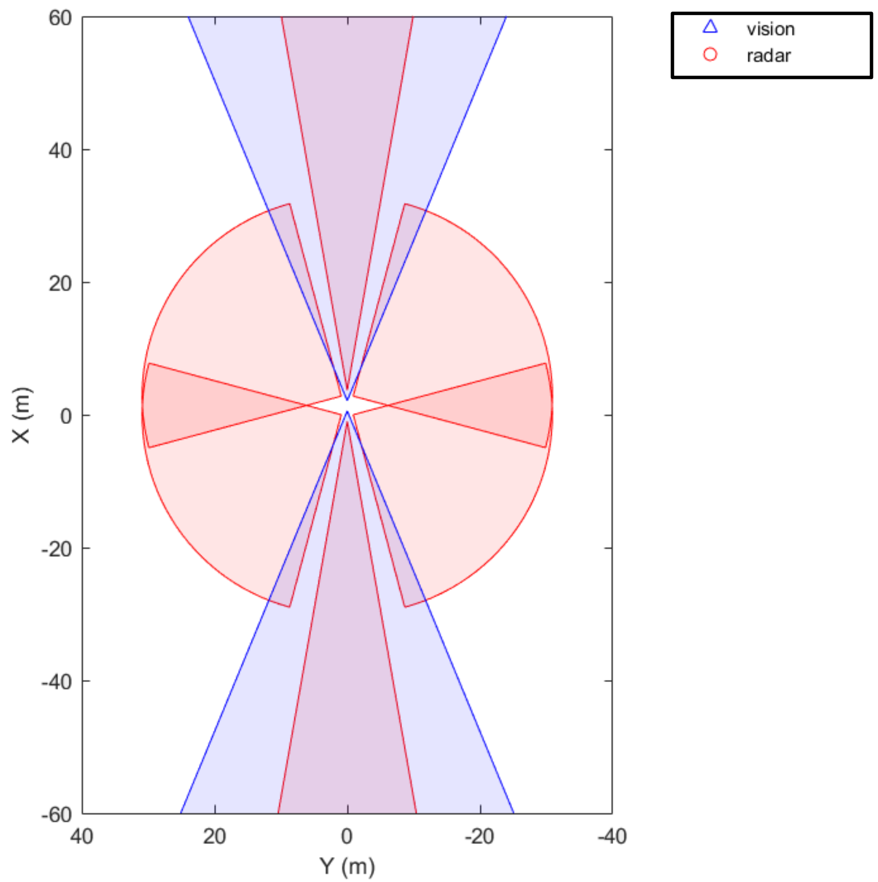
$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$$

Sensor fusion using radar and vision data

Define Radar and Vision Sensors

Sensors defined / ego vehicle:

- 6 radar sensors:
  - 2 long-range radar sensors covering 20 degrees (in front and back),
  - 4 short-range radar sensors covering 90 degrees (two per side),
- 2 vision sensors:
  - Front-facing camera located at front windshield,
  - Rear-facing camera located at rear windshield,
- sensors have some overlap and some coverage gap.



Sensor fusion using radar and vision data

Create a Tracker

- to track the vehicles that are close to the ego vehicle,
- Note:
  1. initialize a constant velocity motion model
  2. initialize the Kalman filter that works with position and velocity

It is responsible for the following:

A. Assigning detections to tracks.

B. Initializing new tracks based on unassigned detections. All tracks are initialized as 'Tentative', accounting for the possibility that they resulted from a false detection.

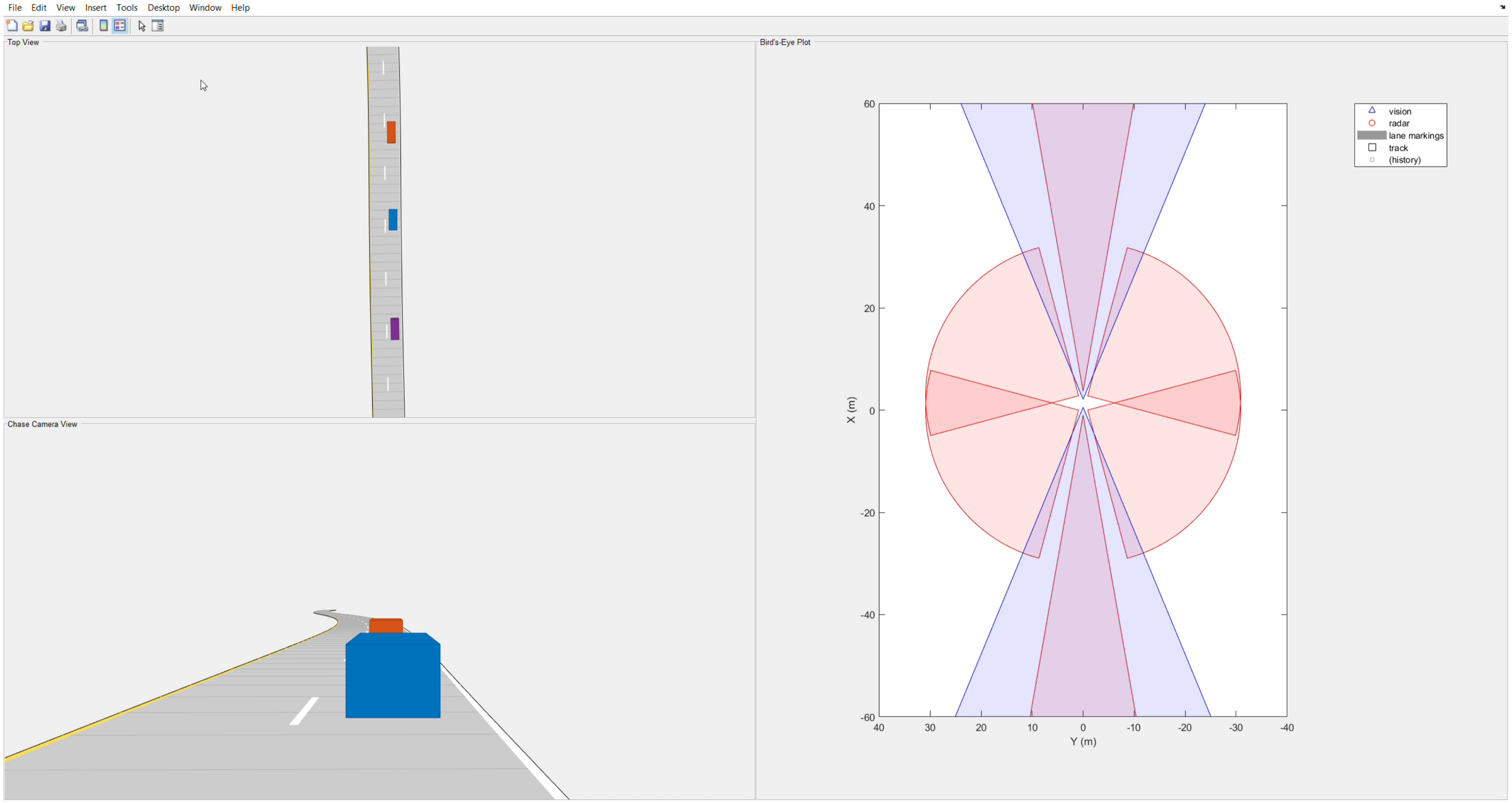
C. Confirming tracks if they have more than *M* assigned detections in *N* frames.

D. Updating existing tracks based on assigned detections.

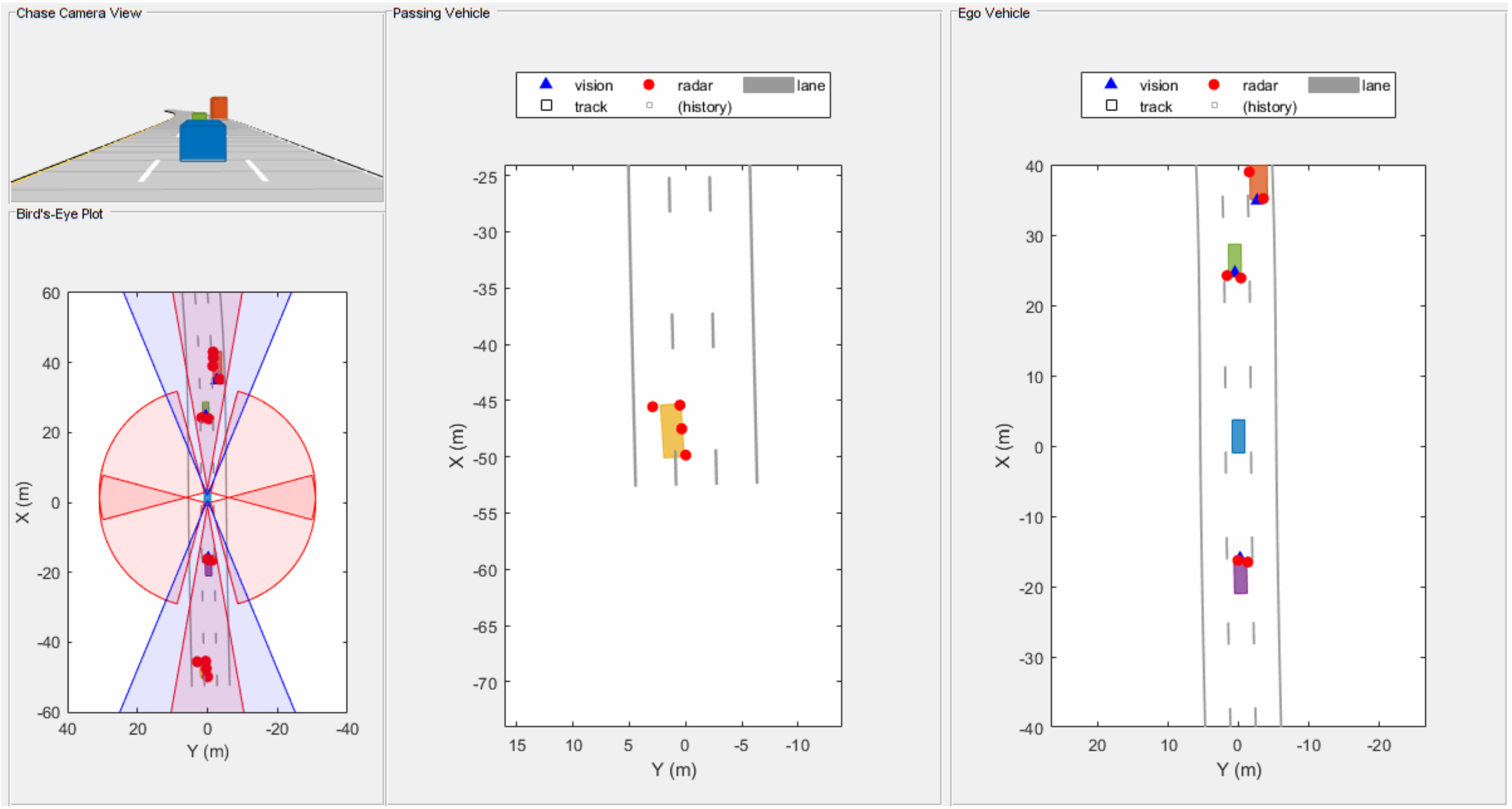
E. Coasting (predicting) existing unassigned tracks.

F. Deleting tracks if they have remained unassigned (coasted) for too long

Sensor fusion using radar and vision data



Sensor fusion using radar and vision data



- Gaussian mixture phd tracker (here: MATLAB implementation)
- Can handle multiple detections per object per sensor (here: 6 radars, 2 cameras)
- It estimates the size and orientation of the object (along with pose and velocity)

Takeaways

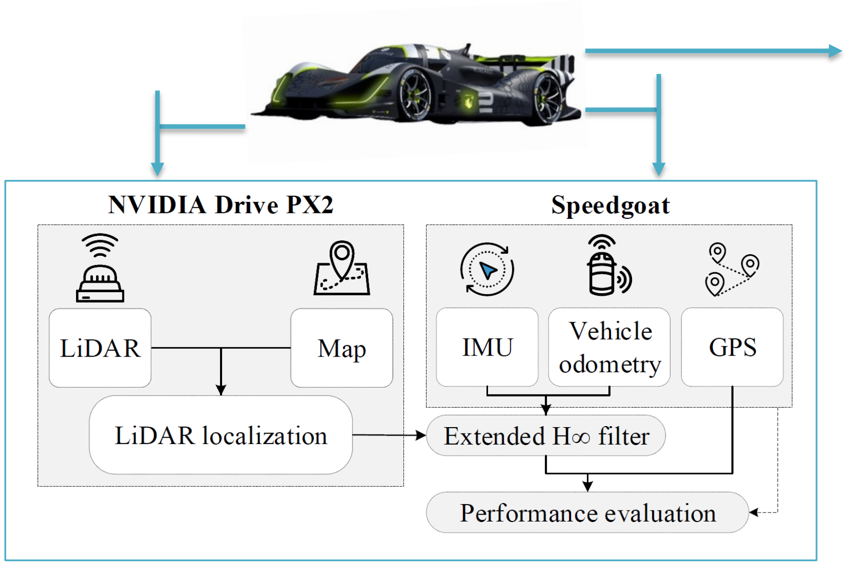
- **Multiple target tracking (MTT), multi-object tracking (MOT)**
  - multiple detections from multiple targets,
  - use of one or more sensors,
  - one or more tracks are used to estimate the states of the targets.

- **Note: Extended object tracking**
  - high-resolution radar/lidar sensors,
  - can handle multiple detections per object.

Common problems in multi-sensor data fusion

- **Registration:** Coordinates (both time and space) of different sensors or fusion agents must be aligned.
- **Bias:** Even if the coordinate axis are aligned, due to the transformations, biases can result. These have to be compensated.
- **Correlation:** Even if the sensors are independently collecting data, processed information to be fused can be correlated.
- **Data association:** multi-target tracking problems introduce a major complexity to the fusion system.
- **Out-of-sequence measurements:** Due to delayed communications between local agents, measurements belonging to a target whose more recent measurement has already been processed, might arrive to a fusion center.
- …

Example: Asynchronous measurements



Sensor	Type
GPS	OXTS 4002
LiDAR	Ouster OS1-64 and OS1-16
Gyroscope	McLaren Applied
Accelerometer	McLaren Applied
Speed sensor	Kistler Correvit SFII

Asynchronous measurements incorporation

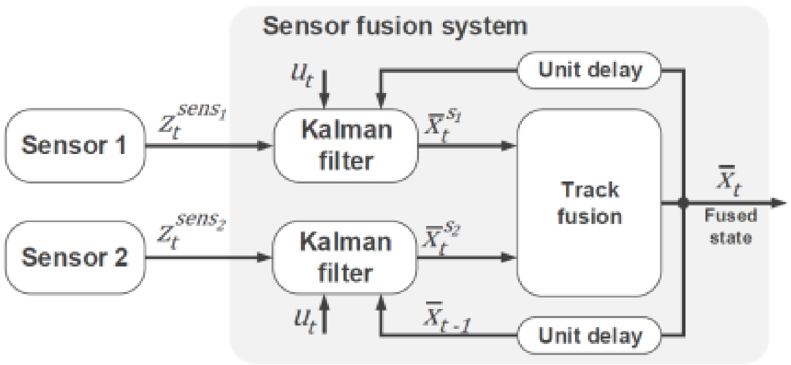
$$z = \begin{cases} [x_{\mathrm{L}}, y_{\mathrm{L}}, \psi_{\mathrm{L}}]^T \,, & \text{LiDAR } (\sim 20 \text{ Hz}) \\ [x_{\mathrm{v}}, y_{\mathrm{v}}, \psi_{\mathrm{v}}]^T \,, & \text{Vehicle pose}(\sim 250 \text{ Hz}) \\ [\dot{\psi}_{\mathrm{v}}]^T, & \text{Vehicle twist}(\sim 250 \text{ Hz}) \\ [\psi_{\mathrm{IMU}}, \dot{\psi}_{\mathrm{IMU}}]^T \,, & \text{IMU}(\sim 240 \text{ Hz}) \end{cases}$$

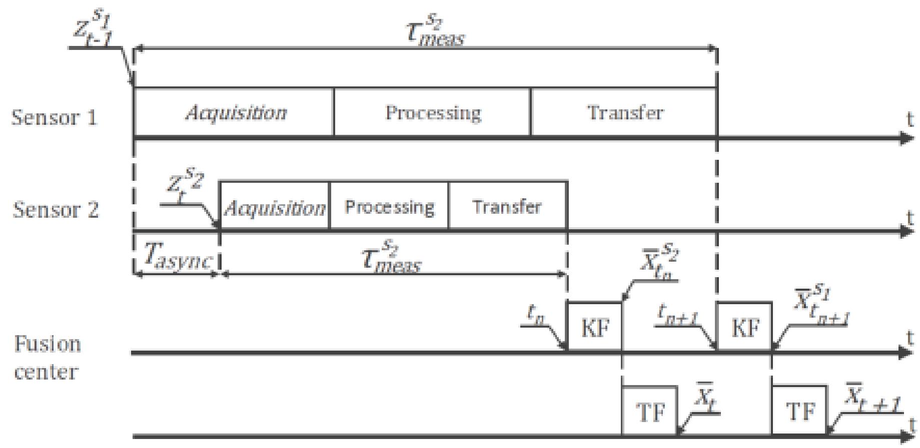
Vehicle motion model: explicit dependence on the sampling time Δt

Allows to incorporate sensors with different update rates correctly.

Example: Out-of-sequence measurements

- Might lead to incorrect temporal order, which in turn causes a negative time measurement update (NTMU) in the fusion algorithm (e.g., EKF).
- As a result, the process of sensor fusion is not performed correctly.
- A wrong representation of the environment is created!





[Source: A. Mehmed, Runtime monitoring of automated driving systems, 2019]

Example: Out-of-sequence measurements

- Timestamping data at arrival (Centralized Method)
  - Measurement cycle time T_c=1/fps
- Timestamping at the time of acquisition (Distributed Method)
  - Global time is needed
- Triggering method (by external source)

Live demo / Autoware

1. Localization with odometry only (IMU)
2. Localization with GNSS without noise
3. Localization with GNSS with noisy data
4. Localization with GNSS with noise and bias
5. Localization with Lidar