Principles of Robot Autonomy I

Multi-sensor perception and sensor fusion

Stanford University

Autonomous Systems Lab

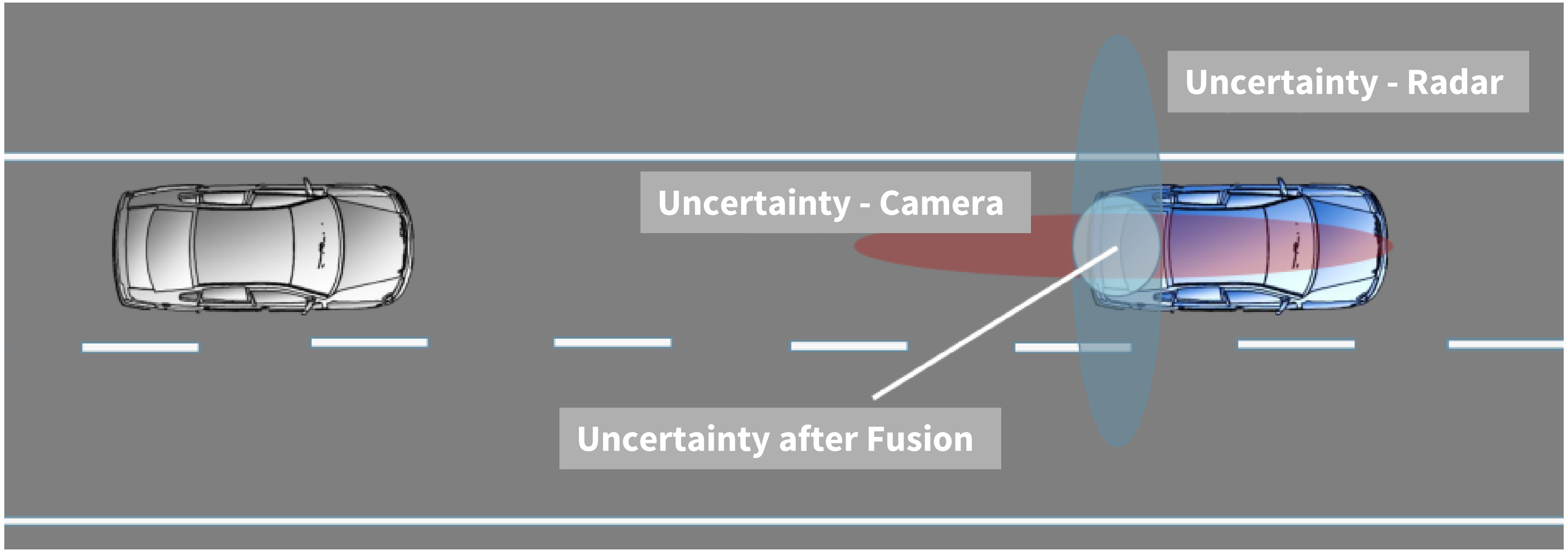
Today’s lecture

- 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

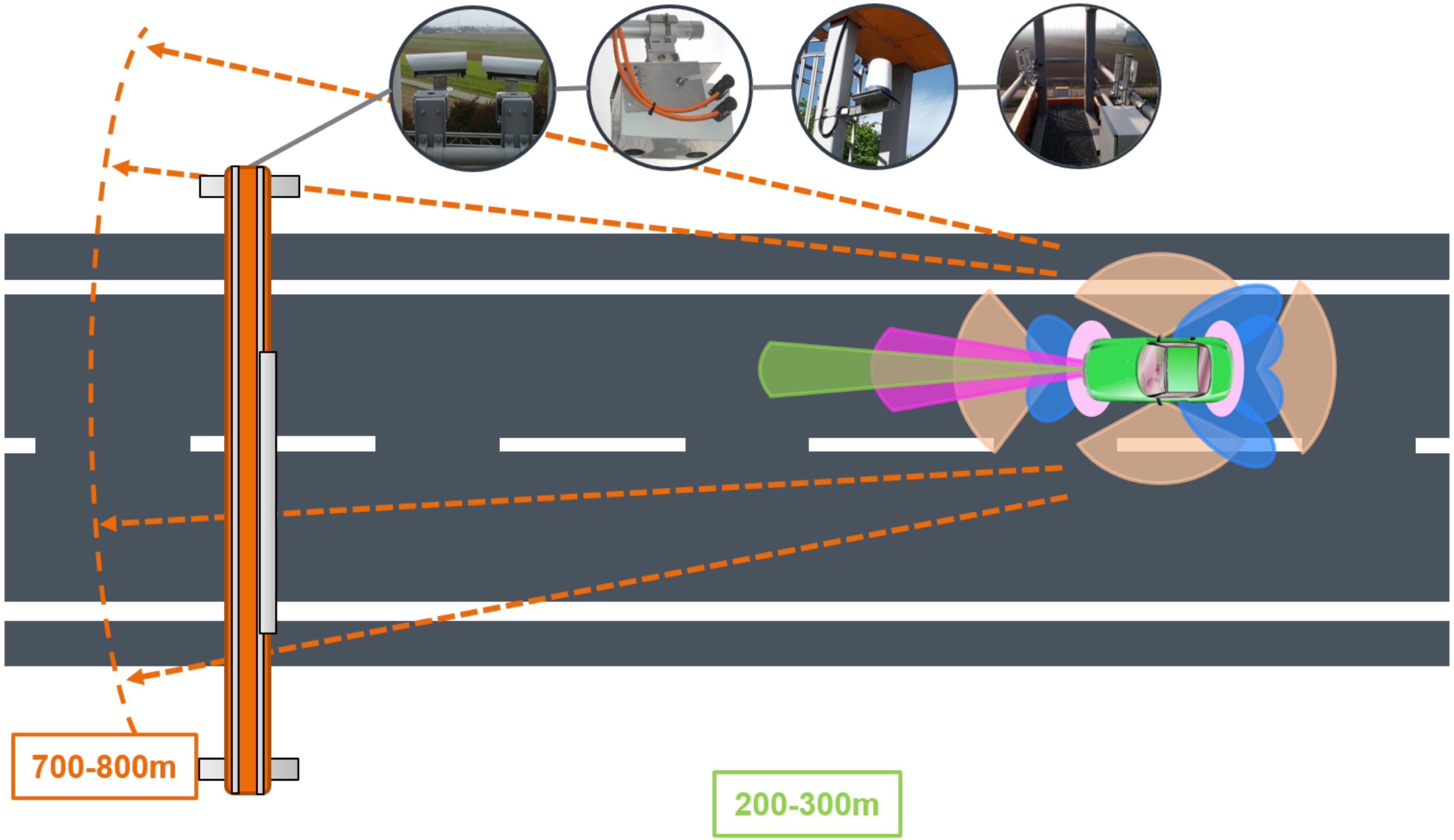
- Readings
  - F. Gustafsson. Statistical Sensor Fusion. 2010.
  - D. Simon. Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches. 2006.

Multi-sensor perception

- Uncertainty reduction



Using stationary sensors



Single-sensor vs multi-sensor perception

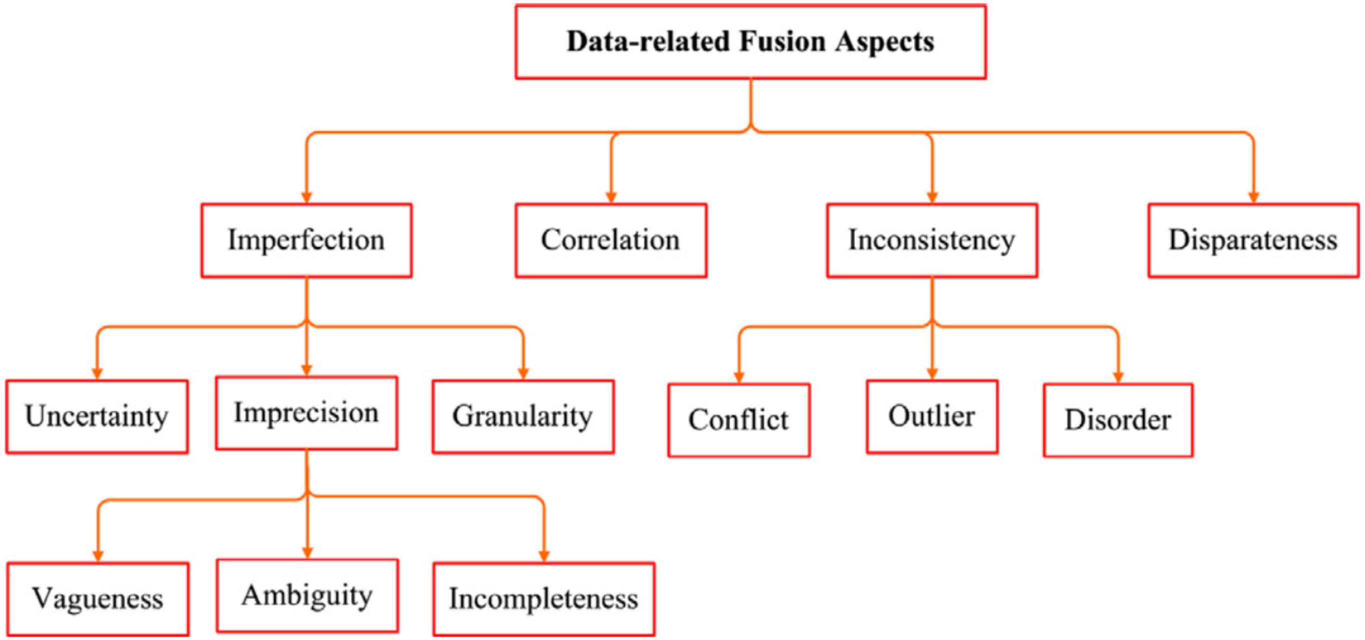
- Drawbacks of single-sensor perception
  - Limited range and field of view
  - Performance is susceptible to common environmental conditions
  - Range determination is not as accurate as required
  - Detection of artefacts, so-called false positives
- Multi-sensor perception might compensate these, 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

Sensor fusion taxonomies

- Data-related taxonomy
- Fusion level taxonomy
- Fusion classes taxonomy
- Architectural taxonomy

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

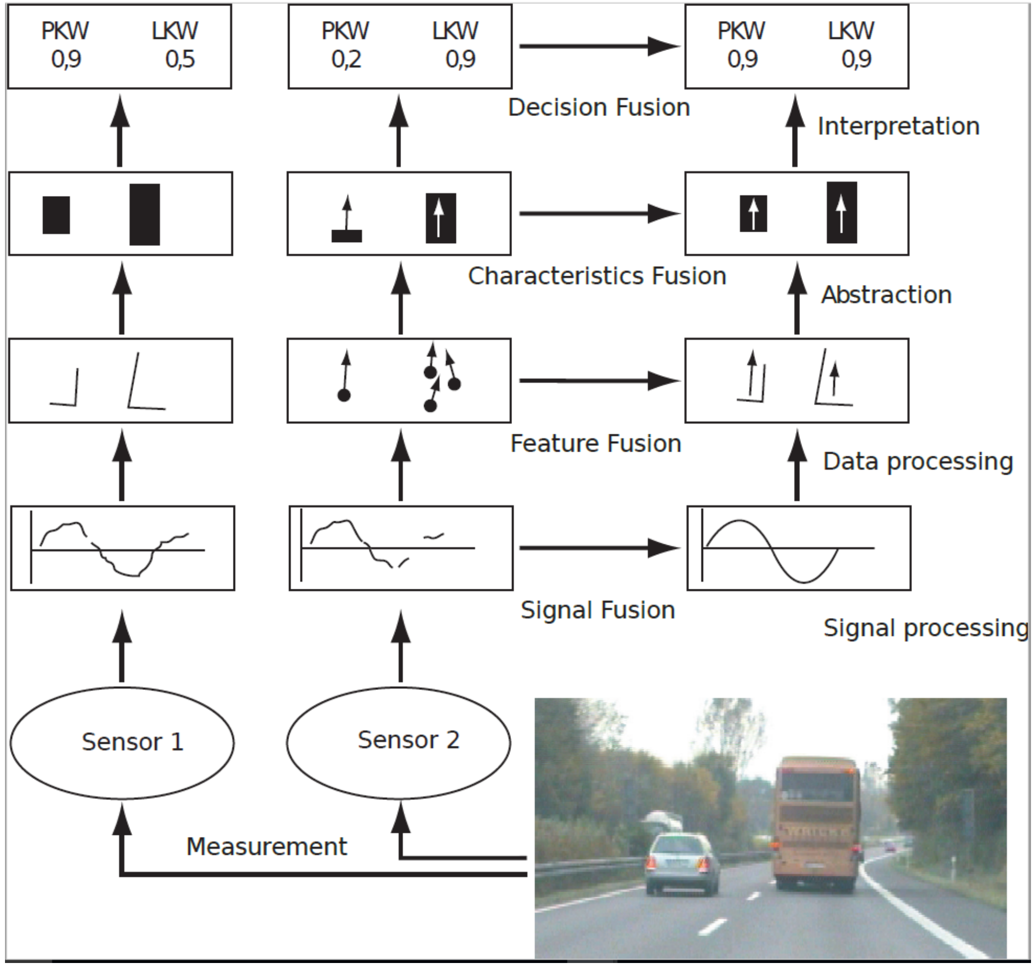


Data-related taxonomy

- 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

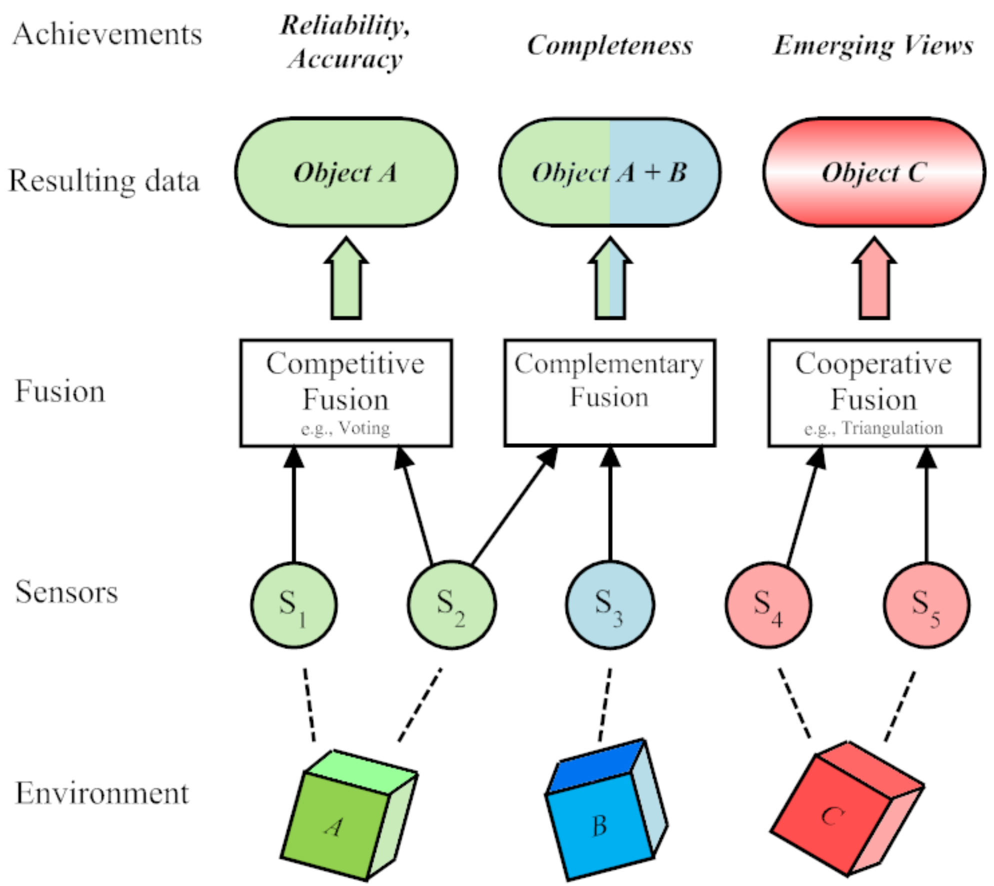
Fusion level taxonomy

- 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



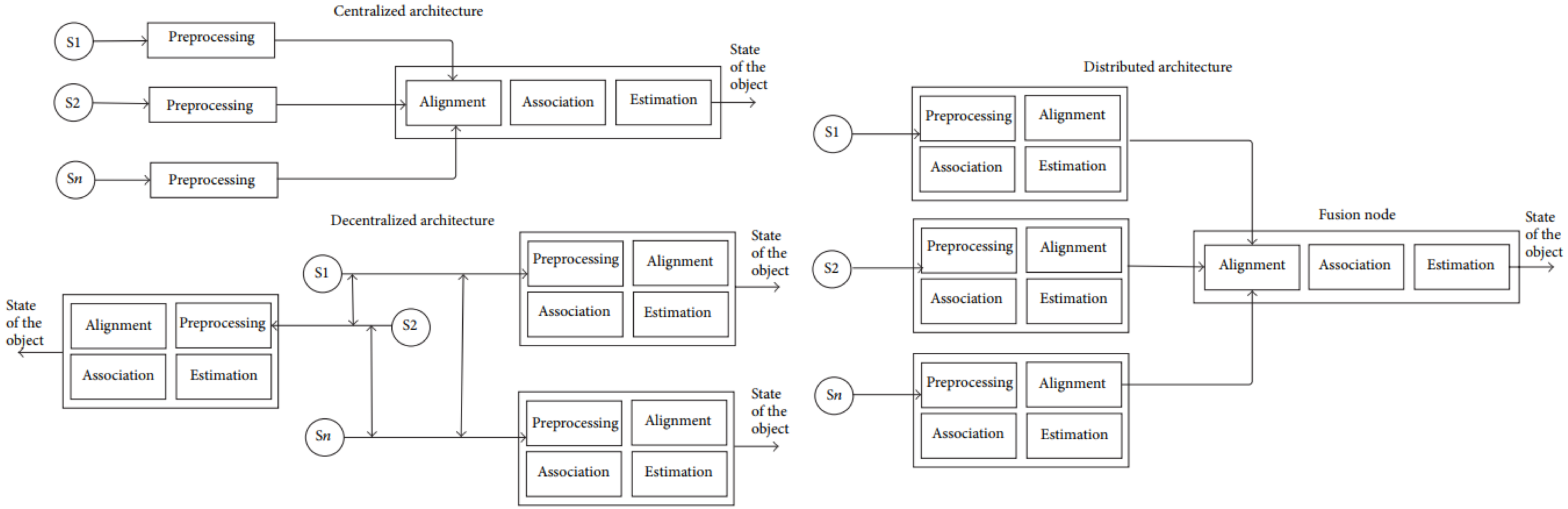
Fusion class taxonomy

- 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



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

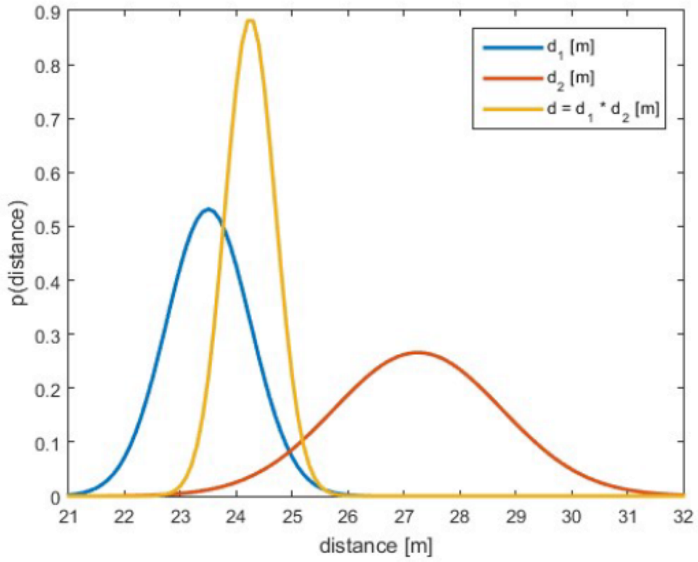
$$
\begin{aligned}
d(x) &= (2\pi\sigma_1\sigma_2)^{-1} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2} + \frac{\mu^2 - s}{2\sigma^2}\right) \\
&= (2\pi\sigma_1\sigma_2)^{-1} \exp\left(\frac{\mu^2 - s}{2\sigma^2}\right) \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right) \\
&= C \cdot \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right)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

Sensor fusion – a simple example

- Mean value and variance are

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

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



- 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

$$p(x_t \mid u_t, x_{t-1}) = \det(2\pi R_t)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t)\right)$$

Kalman filter (KF)

- 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

$$p(z_t \mid x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\right)$$

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

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

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

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

$bel(x_{t-1})$

Data: $(\mu_{t-1}, \Sigma_{t-1}), u_t, z_t$

Result: (μ_t, Σ_t)

Prediction: $\overline{bel}(x_t)$

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \;;$$
$$\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t;$$

Correction: $bel(x_t)$

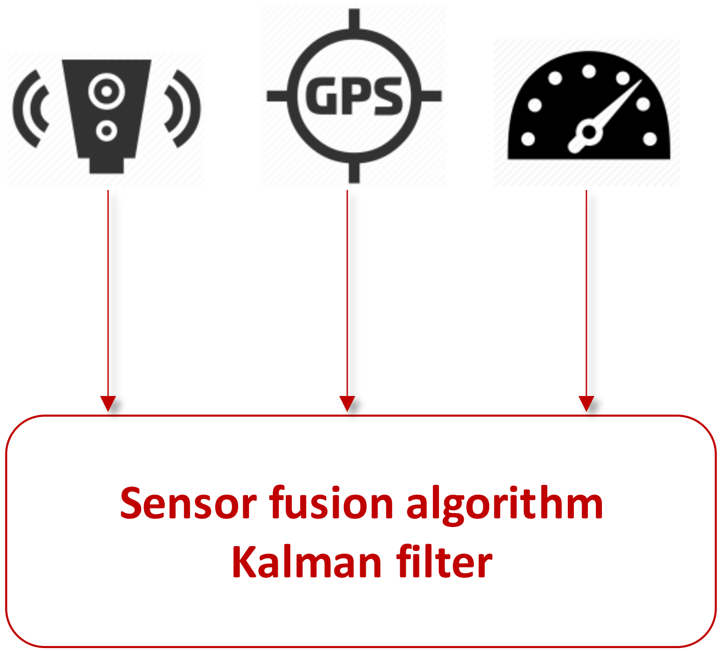
$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1};$$
$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t);$$
$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t;$$

Return (μ_t, Σ_t)

$bel(x_t)$

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

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

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

where ϵ_t is independent 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

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

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

where δ_t is independent 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

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

Sensor fusion example: Algorithm

- Estimation results are obtained using the prediction-correction scheme

Prediction

Project state ahead

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

Project covariance ahead

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

Correction

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

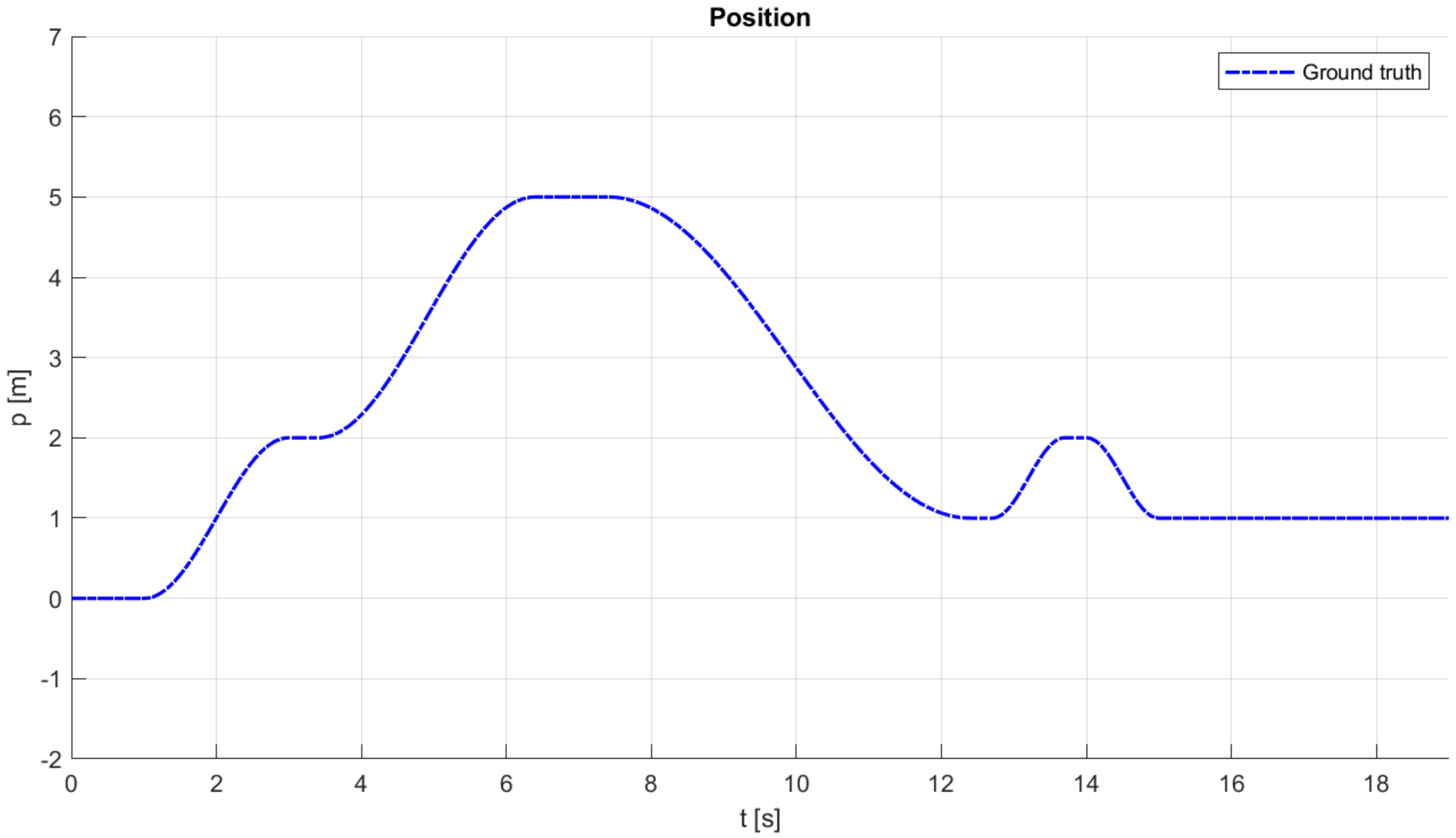
Update estimate with new measurements

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

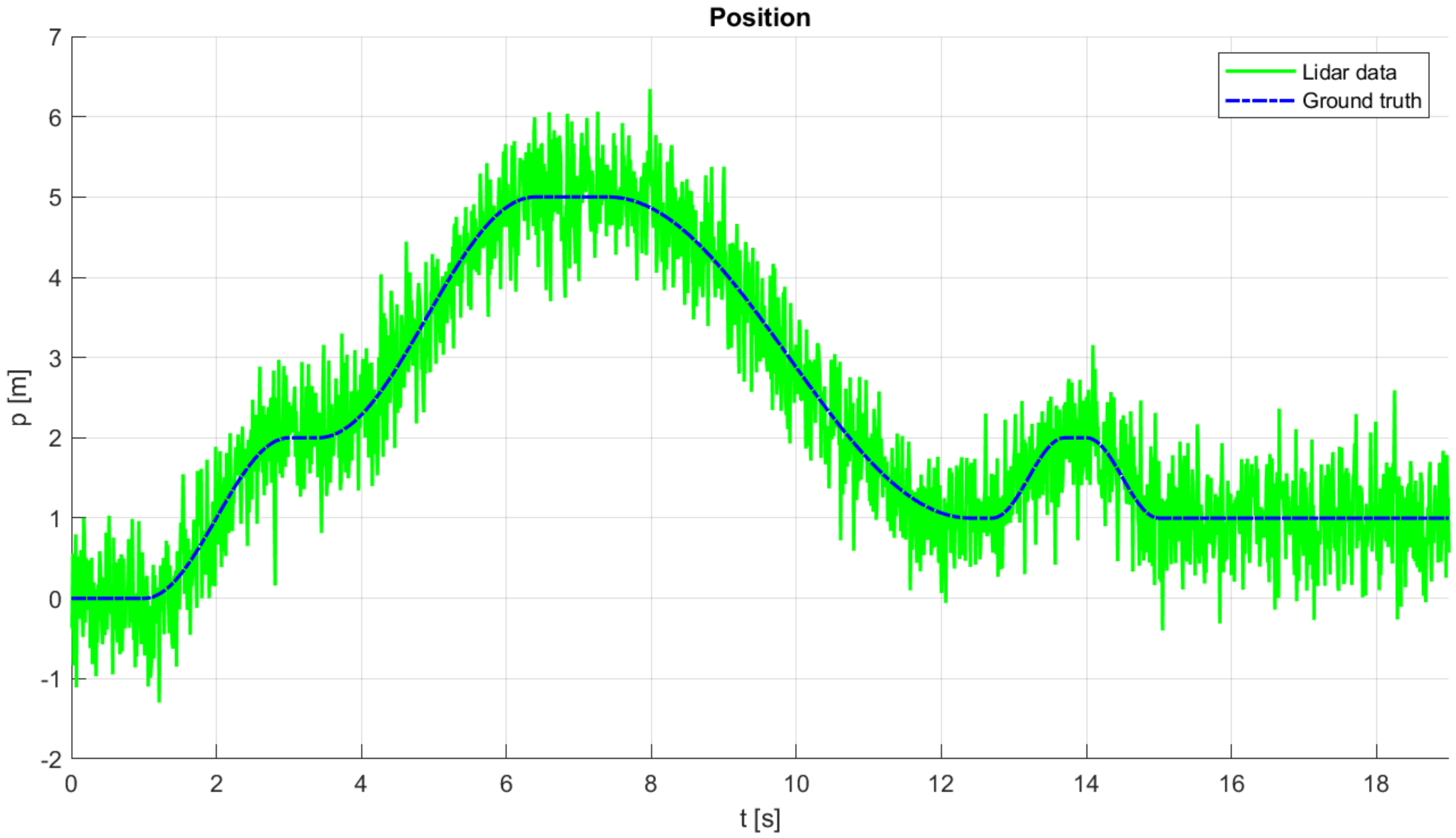
Update covariance

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

Sensor fusion example: Position estimation

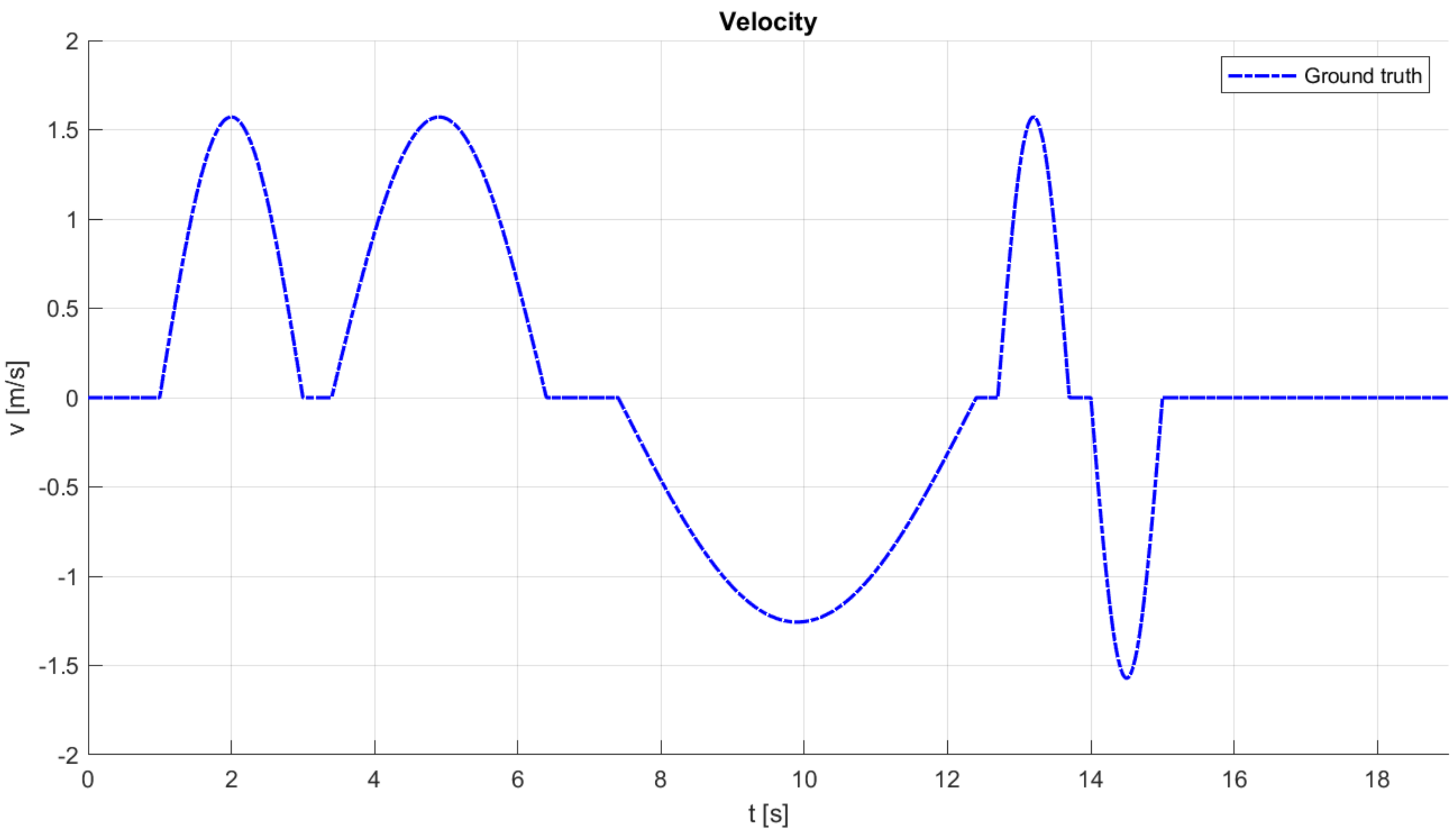


Sensor fusion example: Position estimation

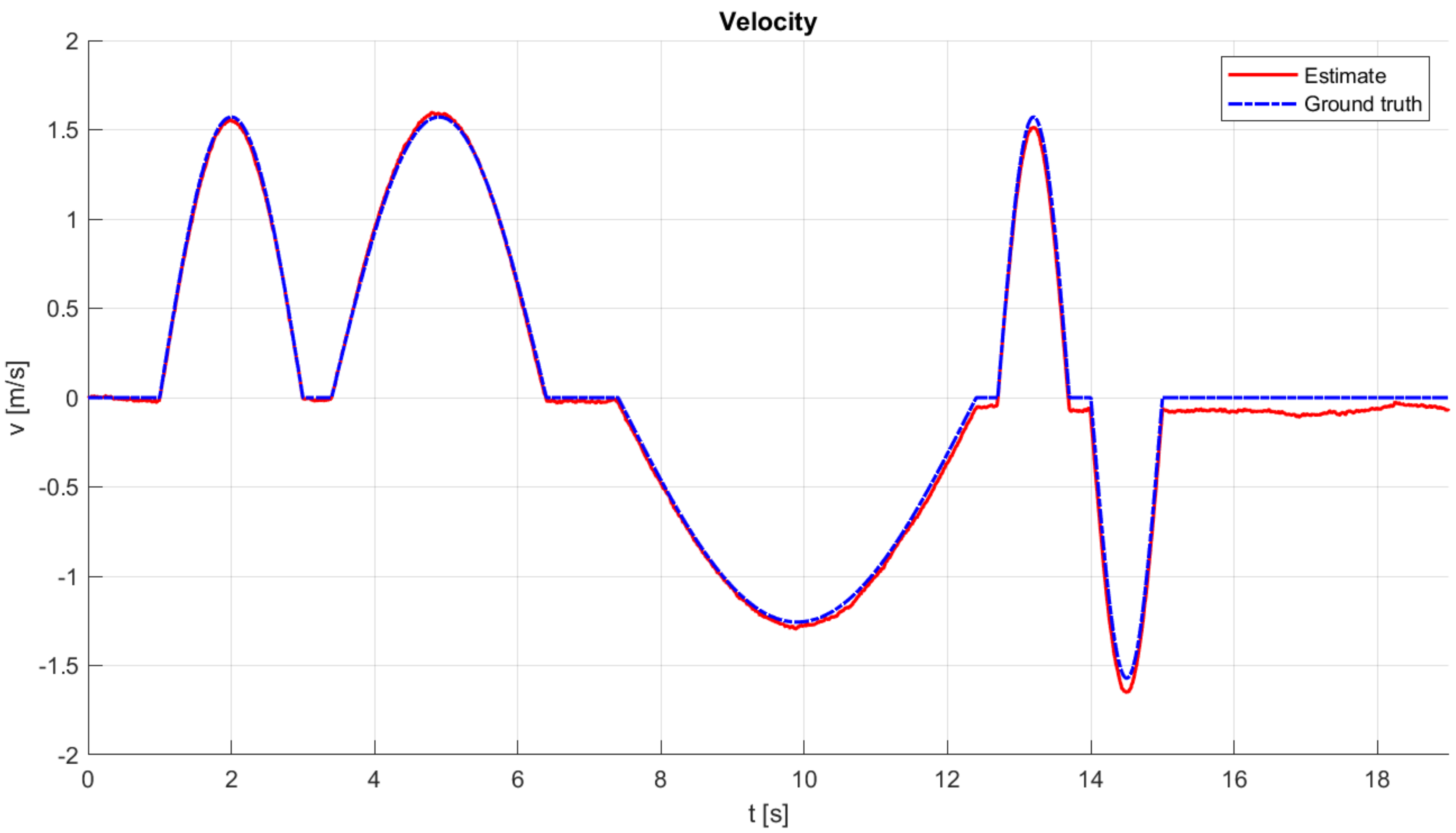


Sensor fusion example: Position estimation

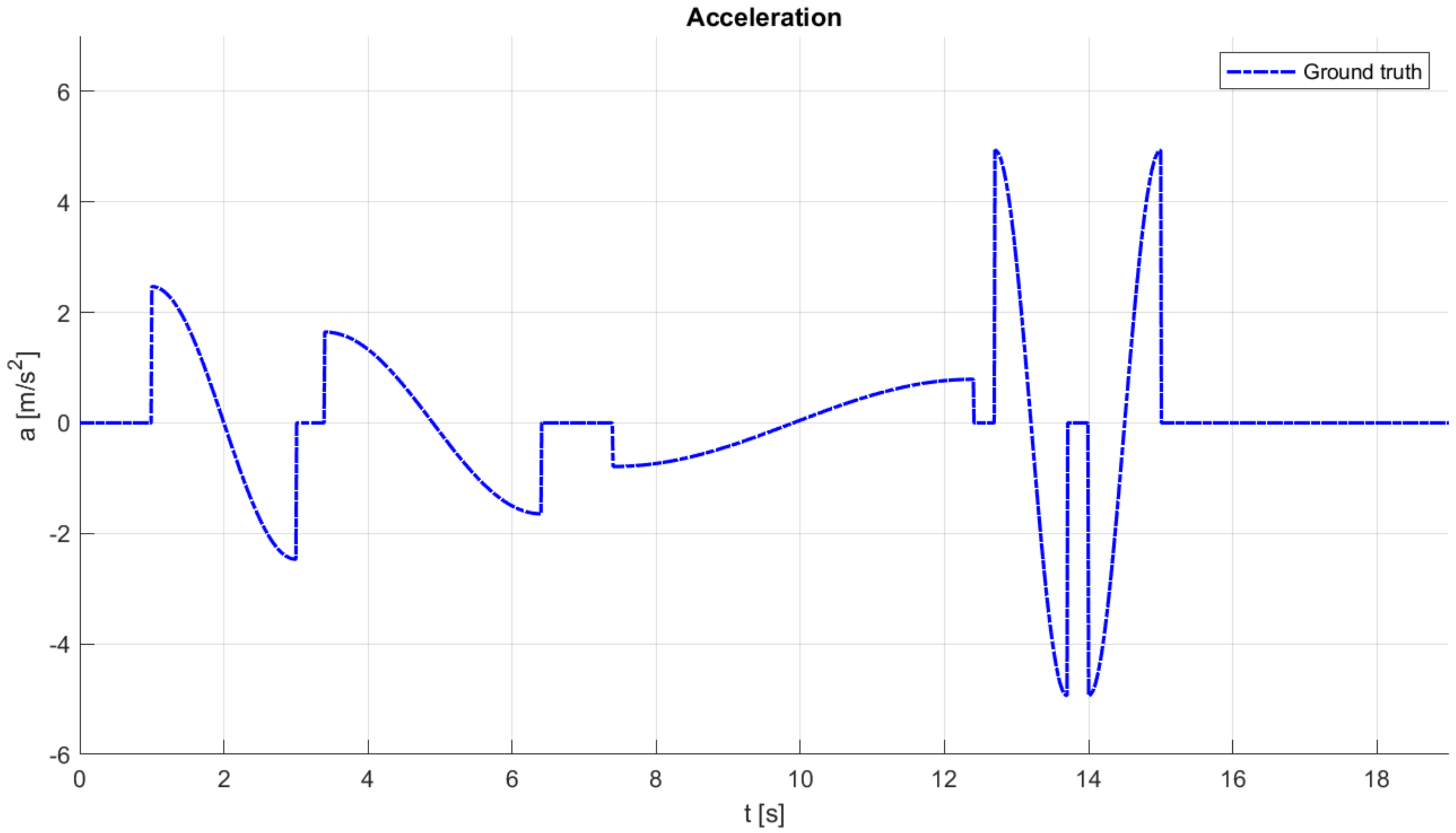
Sensor fusion example: Velocity estimation



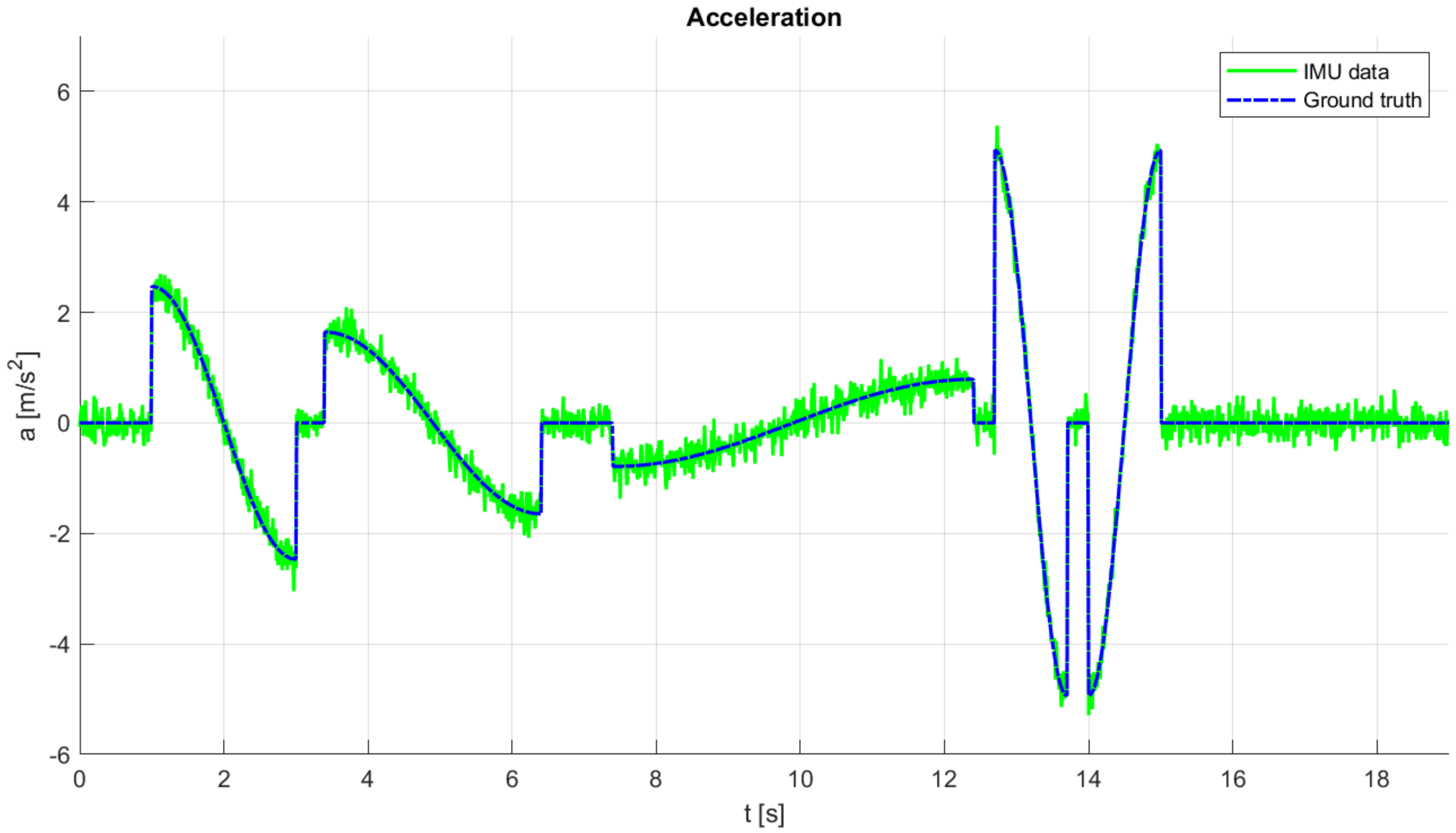
Sensor fusion example: Velocity estimation



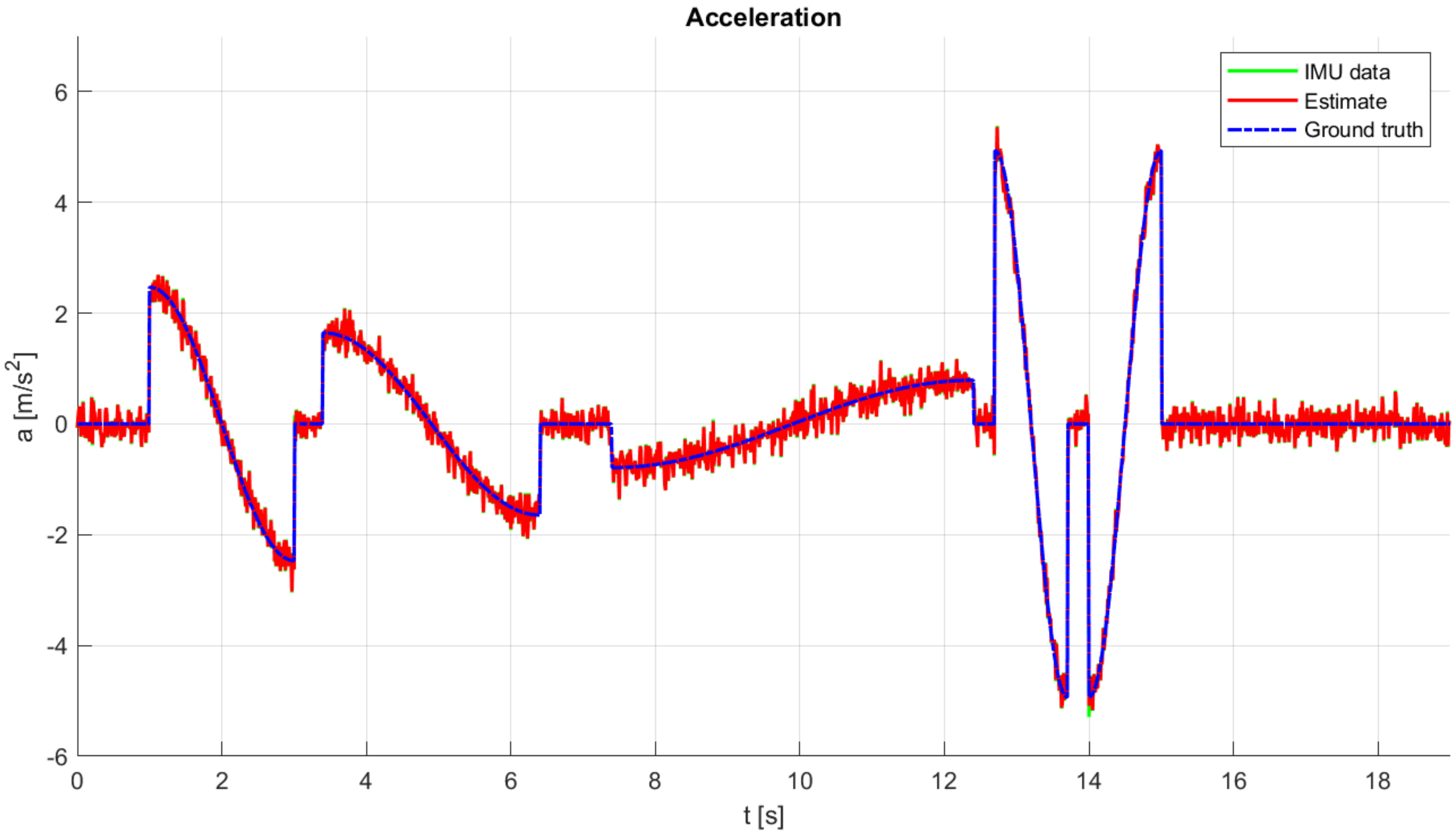
Sensor fusion example: Acceleration estimation



Sensor fusion example: Acceleration estimation



Sensor fusion example: Acceleration estimation



Sensor fusion example: Measurement model

- In the previous scenario – the position estimate is quite noisy (because of the low precision of the Lidar measurements)
- Therefore, in the second scenario, position is measured with Lidar and GPS

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

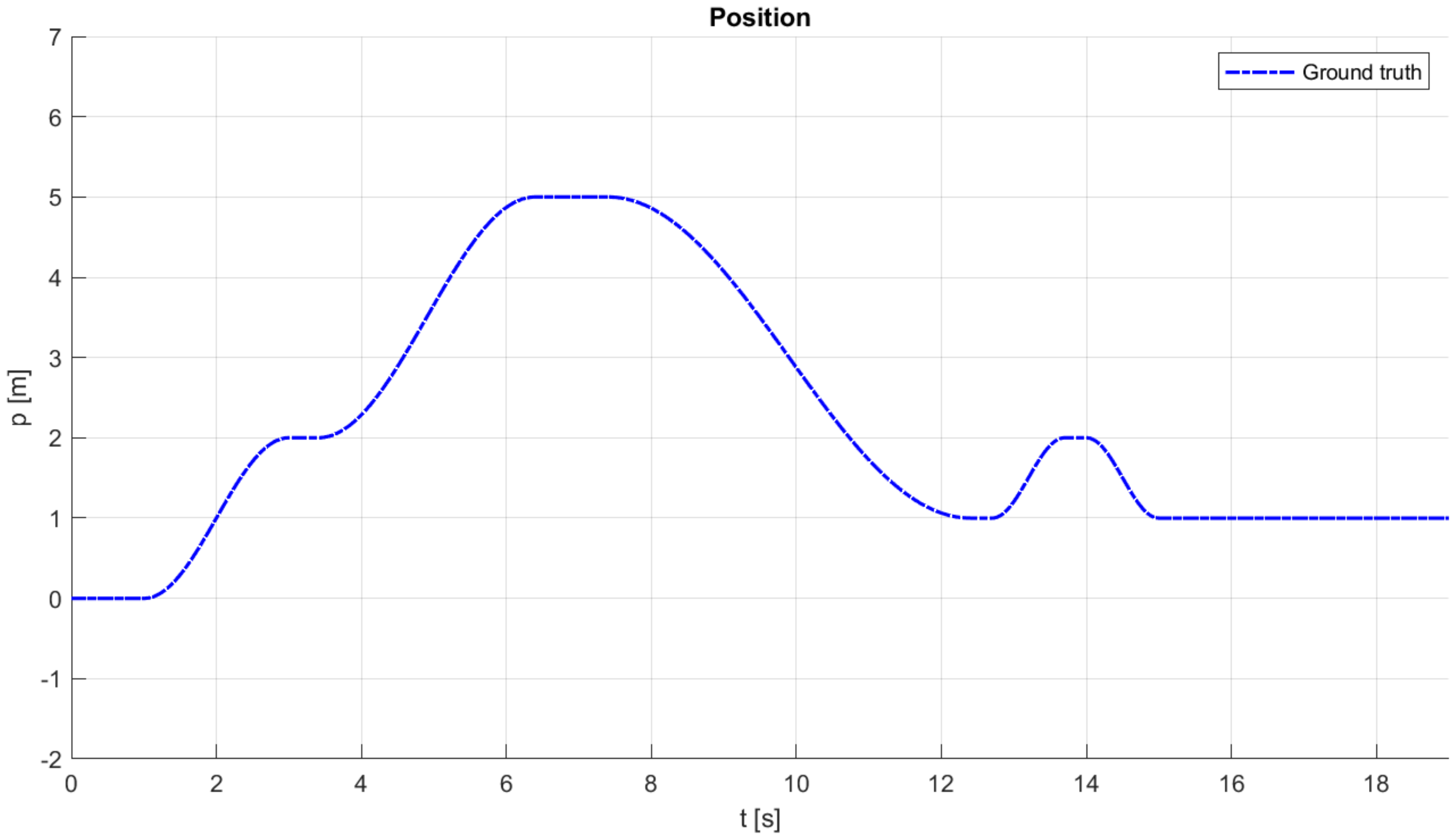
$$z_t = C_t \mu_t + \delta_t$$

Sensor fusion example: Noise model tuning

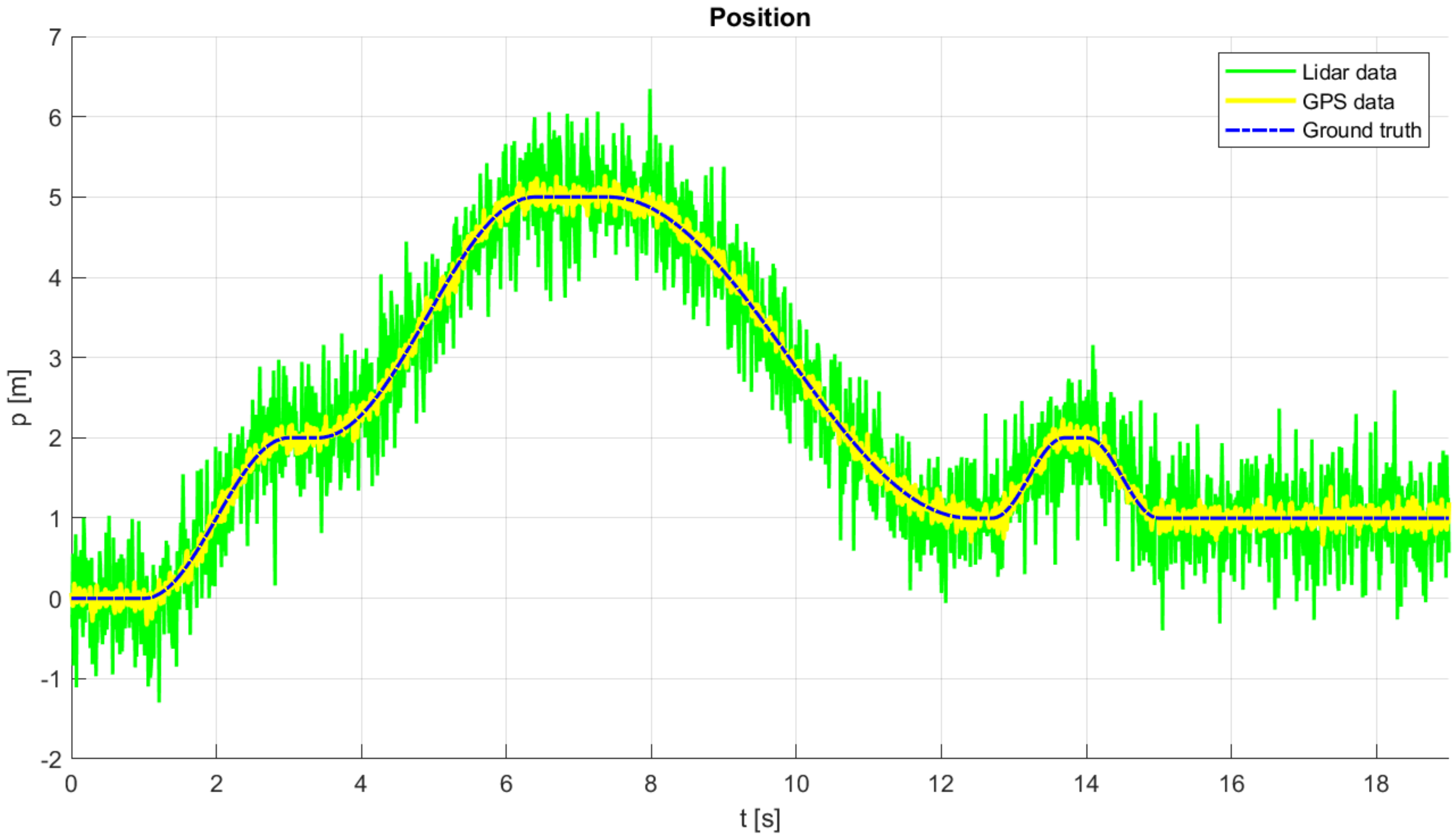
- The measurement noise covariance matrix Q_t for this scenario has an additional GPS variance

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

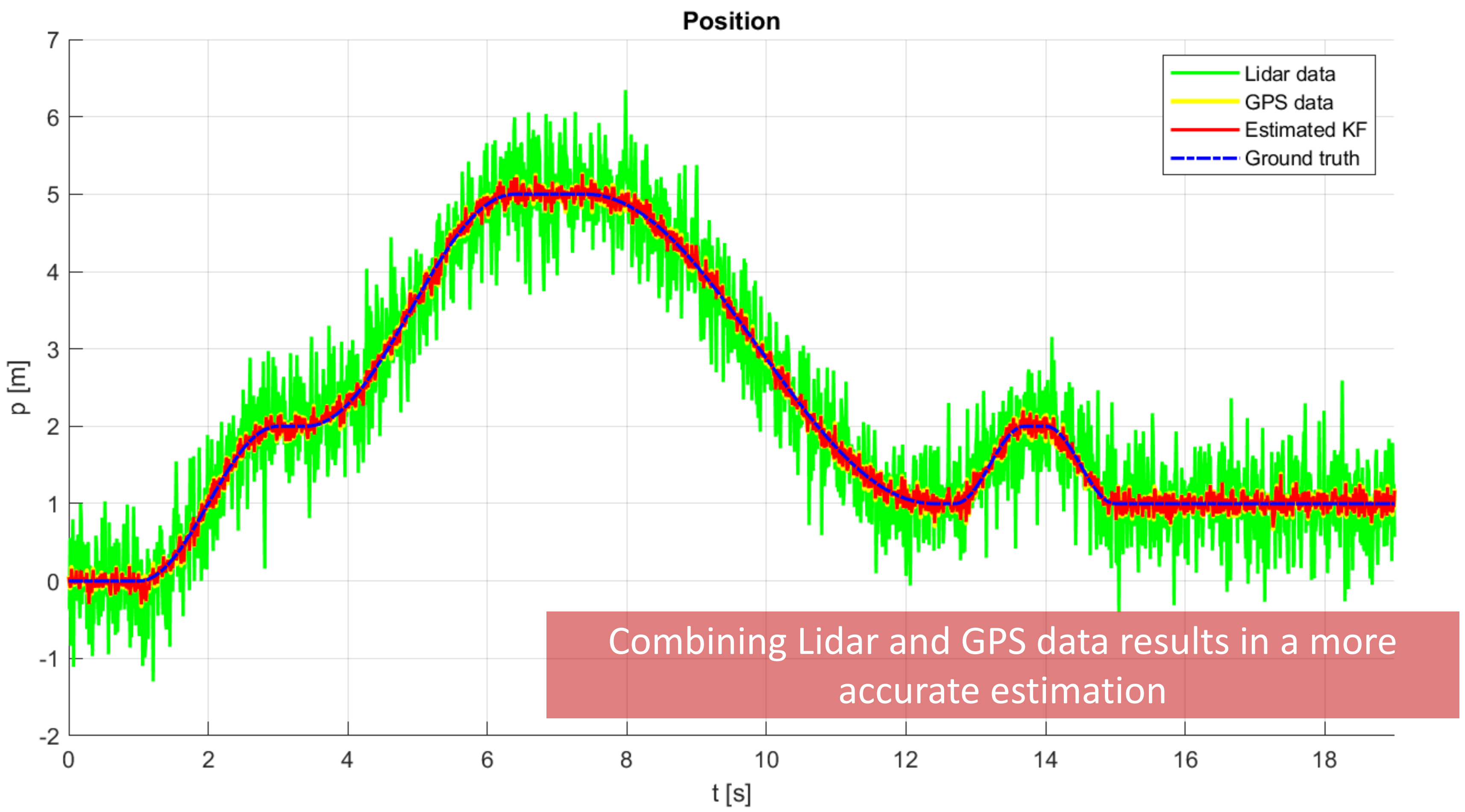
Sensor fusion example: Position estimation



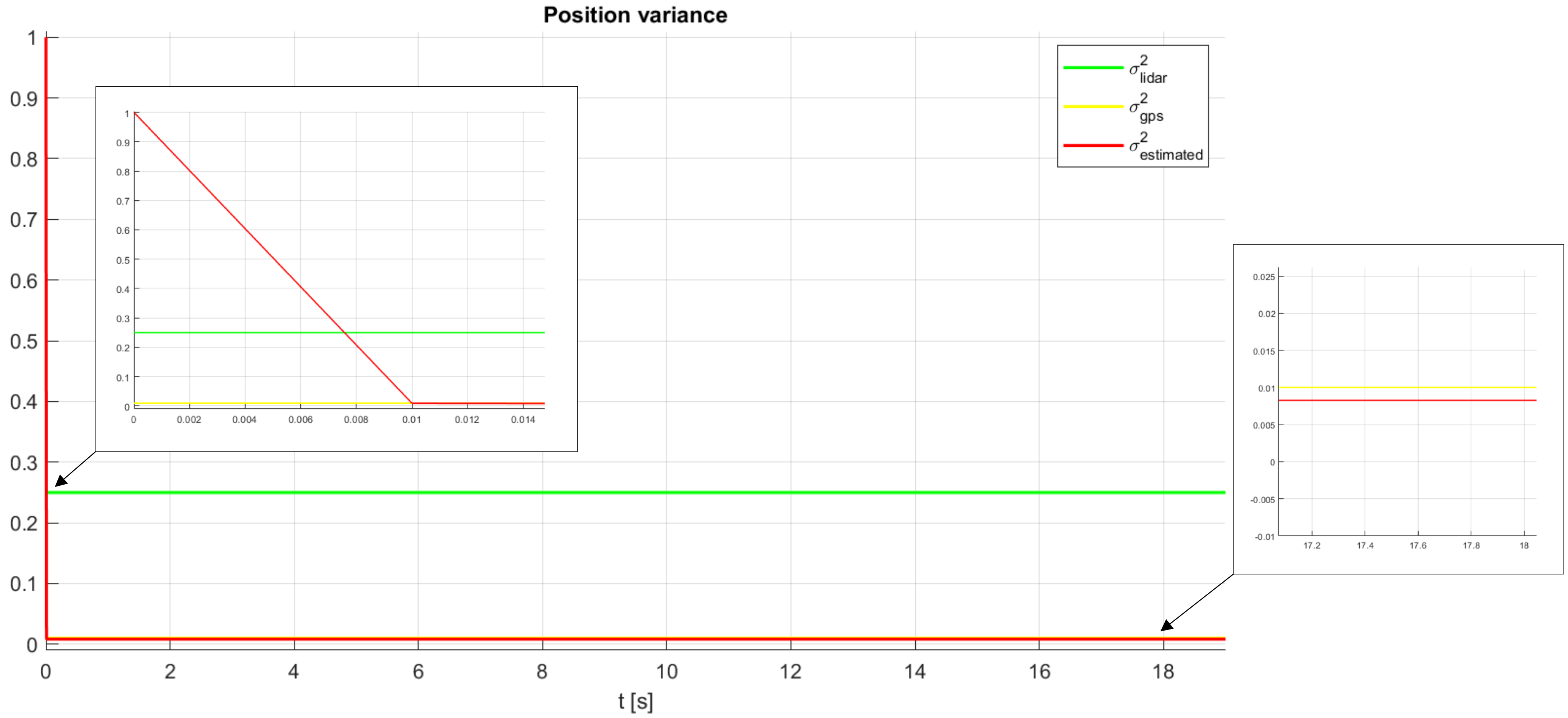
Sensor fusion example: Position estimation



Sensor fusion example: Position estimation



Sensor fusion example: Position 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)

Sensor fusion example: Conclusion

- Problem: Vehicle state estimation using Kalman filter
- 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

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.
- …