BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

# 5 – Kalman Filter and EKF

## Advanced Methods for Mapping and Self-localization in Robotics (MPC-MAP)

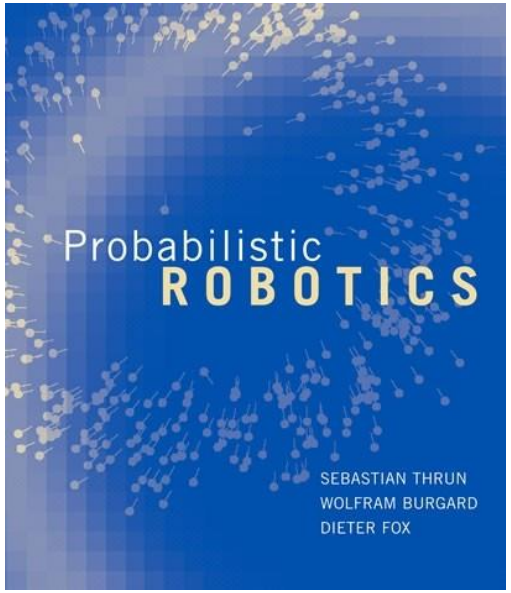
Petr Gabrlik

Brno University of Technology

2021

The materials presented herein are mainly based on the ***Probabilistic Robotics*** book by *Sebastian Thrun* et al. [PR]

This presentation contains equations and graphics from the book; some images are adopted from the slides available at [probabilistic-robotics.org](probabilistic-robotics.org). Such materials are marked with [PR].

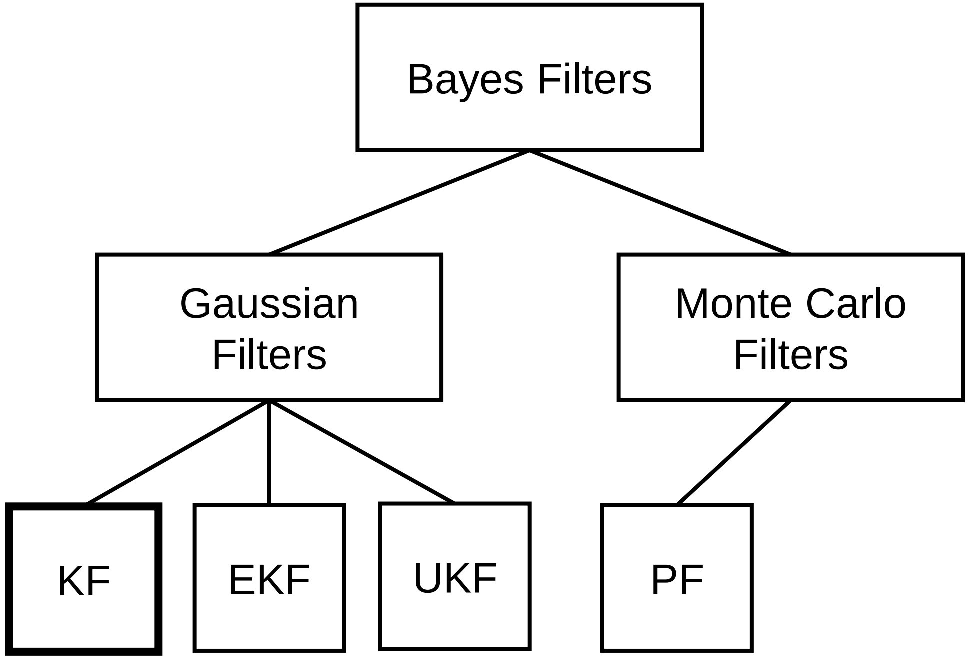



[PR]

[PR] THRUN, Sebastian, BURGARD, Wolfram and FOX, Dieter, 2005. Probabilistic Robotics. 1st edition. Cambridge, Mass: The MIT Press. ISBN 978-0-262-20162-9.

**Kalman Filter**

- An algorithm for ***filtering*** and ***prediction*** in linear systems / ***estimating*** unknown variables.
- Gaussian filter, an early implementation of Bayes filter for continuous space.
- The best studied technique for Bayes filters.
- Widely used and popular technique to date.



**Kalman filter**

- Developed and introduced in ~1950s.
- Named after *Rudolf E. Kálmán*, Hungarian-American engineer/mathematician.
- Similar algorithm developer by other researchers that time.
- First described by technical papers by Swerling (1958), Kalman (1960) [1] and Kalman and Bucy (1961).

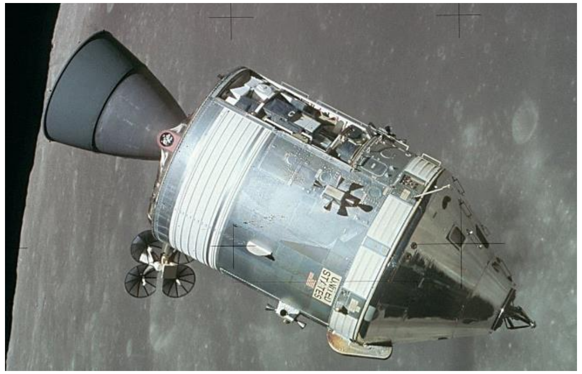
Rudolf E. Kálmán (1930 – 2016)

[2]

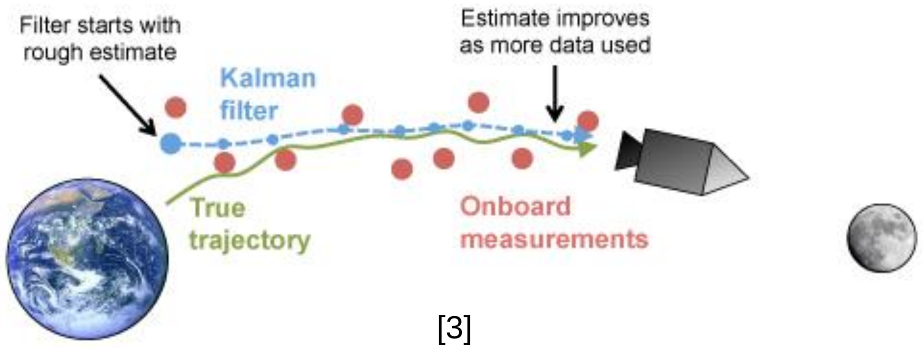
[1] KALMAN, Rudolph Emil, 1960. A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME – Journal of Basic Engineering. 1960. Vol. 82, no. Series D, p. 35–45. Available from: https://www.cs.unc.edu/~welch/kalman/media/pdf/Kalman1960.pdf
[2] Rudolf E. Kálmán, 2021. Wikipedia [online]. [Accessed 4 March 2021]. Available from: https://en.wikipedia.org/w/index.php?title=Rudolf_E._K%C3%A1lm%C3%A1n&oldid=1001450066

# Applications – Apollo

- Used for ***trajectory estimation*** for the ***Apollo*** program in the ~1960s [1].
- One of the very first applications of the Kalman filter.
- EKF due to system nonlinearities.
- Sensors:
  - Accelerometers for thrusting periods.
  - Optical sextant (sparse measuremets)
- Implemented at onboard computer:
  - 2k of magnetic core RAM,
  - 36k wire rope (ROM) memory,
  - CPU built from ICs, clock <100 kHz.



[2]



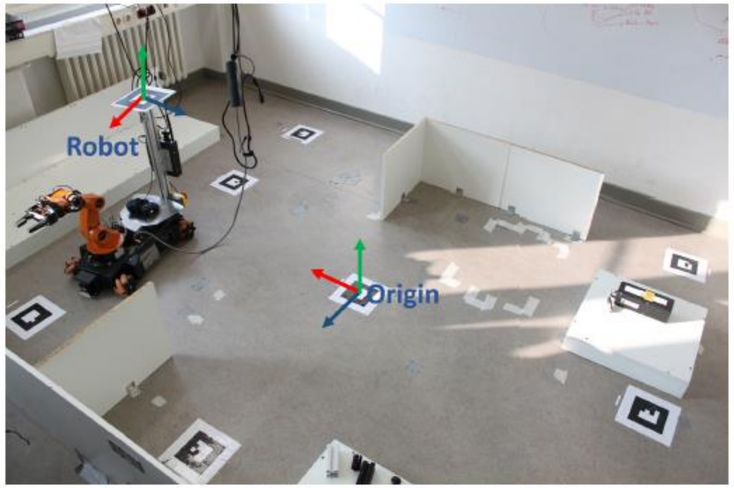

[3]

[1] GREWAL, M. S. and ANDREWS, A. P., 2010. Applications of Kalman Filtering in Aerospace 1960 to the Present [Historical Perspectives]. *IEEE Control Systems Magazine*. June 2010. Vol. 30, no. 3, p. 69–78. DOI 10.1109/MCS.2010.936465.
[2] Apollo command and service module, 2021. Wikipedia [online]. [Accessed 4 March 2021]. Available from: https://en.wikipedia.org/wiki/Apollo_command_and_service_module
[3] Implementations of Kalman Filter From Aerospace to Industry. P2 SMTP LIPI [online]. 2018 [cit. 2021-01-18]. Available at: http://smtp.lipi.go.id/berita633-Implementations-of-Kalman-Filter-From-Aerospace-to-Industry.html

# Applications – Robotics

- ***Tracking problems*** - the belief represents the estimate of the true state with a small uncertainty (unimodal).
- Data fusion:
  - AHRS/INS – accelerometer, gyrocsopes, magnetometers, barometer, GNSS.
  - Robot local localization – odometry + fiducial markers.


[1]




[2]

[1] https://www.sbg-systems.com/
[2] HITZMANN, Arne, WENTSCHER, Philipp, GABEL, Alexander and GERNDT, Reinhard, 2014. Automated Testing of Workshop Robot Behavior. International Journal of Mechanical and Materials Engineering. 3 April 2014. Vol. 8, no. 5, p. 732–735.

# KF and PF Comparison

**KF (~1950s)**

- Gaussian filter
- Implementation of Bayes filters
- Recursive
- Parametric
- Unimodal
- Continuous space
- Discrete time
- Linear systems (*EKF for nonlinear*)
- Analytic method
- Optimal for linear Gaussian systems
- Belief represented by multivariate norm. distr.
- Effective on high-dimensional systems

# KF and PF Comparison

**KF (~1950s)**

- Gaussian filter
- Implementation of Bayes filters
- Recursive
- Parametric
- Unimodal
- Continuous space
- Discrete time
- Linear systems (*EKF for nonlinear*)
- Analytic method
- Optimal for linear Gaussian systems
- Belief represented by multivariate norm. distr.
- Effective on high-dimensional systems

**PF (mid-1990s)**

- Non-Gaussian filter
- Implementation of Bayes filters
- Recursive
- Nonparametric
- Multimodal
- Continuous space
- Discrete time
- Linear and nonlinear systems
- Numerical method (Monte Carlo)
- Suboptimal for linear Gaussian systems
- Belief represented by weighted set of particles
- Less effective on high-dimensional systems

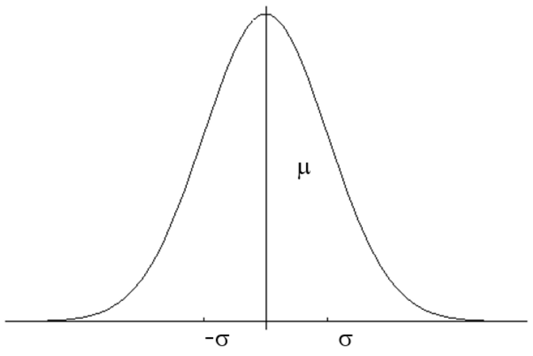
# Bayes Filter

- Random variables possess ***probability density functions*** (PDFs).
- ***Belief*** (*bel*)
  - Momentary state estimate (robot position, actual speed etc.).
  - Represented by PDF over the state space.
- Prior belief – before observations.
- Posterior belief – after observations.

# Normal Distribution

- ***Kalman Filter*** works in „Gaussian world“ – it solely uses ***normal distribution*** as PDF.
- One-dimensional normal distribution ($x$ is a scalar value) is defined by ***Gaussian function*** with the ***mean*** $\mu$ and ***variance*** $\sigma^2$.

$$p(x) \sim \mathcal{N}(x; \mu, \sigma^2): \qquad p(x) = (2\pi\sigma^2)^{-\frac{1}{2}} \exp\{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\} \tag{1}$$

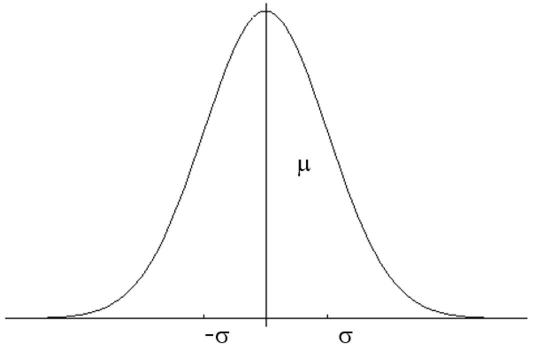



[PR]

# Normal Distribution

- ***Kalman Filter*** works in „Gaussian world“ – it solely uses ***normal distribution*** as PDF.
- One-dimensional normal distribution ($x$ is a scalar value) is defined by ***Gaussian function*** with the ***mean*** $\mu$ and ***variance*** $\sigma^2$.

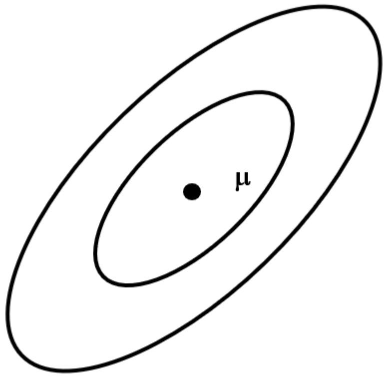
$$p(x) \sim \mathcal{N}(x; \mu, \sigma^2): \qquad p(x) = (2\pi\sigma^2)^{-\frac{1}{2}} \exp\{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\} \tag{1}$$



[PR]

- Normal distribution over vectors is called ***multivariate*** ($x$ is a vector); it is characterized by the ***mean vector*** $\mu$ and ***covariance matrix*** $\Sigma$.

$$p(x) \sim \mathcal{N}(x; \mu, \Sigma): \qquad p(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\} \tag{2}$$



[PR]

- PDF always integrate to 1: $\int p(x)dx = 1$ (3)

# Covariance Matrix

- ***Covariance*** is a measure of the joint variability of two random variables.

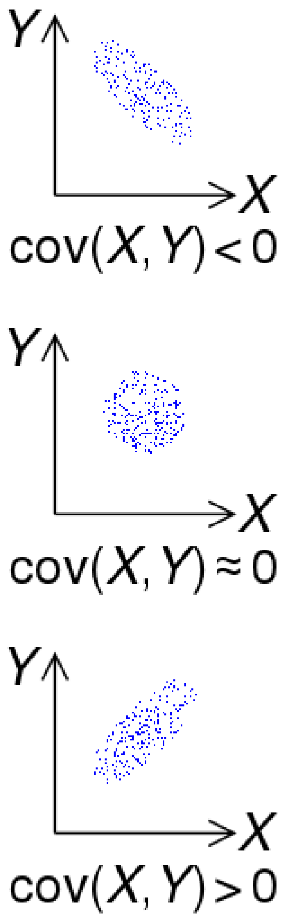
$$cov(X,Y) = E[(X - E[X])(Y - E[Y])] \qquad E[X] \approx mean(X)$$

- ***Covariance matrix*** contains covariance between each pair of elements.
- Matrix dimension: dimensionality of the state $x$ squared.
- Key matrix properties: square, symmetric, quadratic.

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \cdots & \sigma_{1n} \\ \vdots & \ddots & \vdots \\ \sigma_{n1} & \cdots & \sigma_n^2 \end{pmatrix}$$

- Main diagonal contains variances (the covariance of each element with itself).

$$cov(X,X) = var(X) \equiv \sigma^2(X) \equiv \sigma_X^2$$




[1]

[1] Covariance, 2021. Wikipedia [online]. [Accessed 4 March 2021]. Available from: https://en.wikipedia.org/wiki/Covariance

# Kalman Filter (KF)

State estimator for linear systems

Linear discrete-time system (process):

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \tag{4}$$

$$z_t = C_t x_t + \delta_t \tag{5}$$

Linear discrete-time system (process):

$$\boxed{\begin{aligned} x_t &= A_t x_{t-1} + B_t u_t + \varepsilon_t \qquad (4) \\ z_t &= C_t x_t + \delta_t \qquad (5) \end{aligned}}$$

$\longrightarrow$ defines the state transition probability $p(x_t | u_t, x_{t-1})$

| | |
|---|---|
| $x_t, x_{t-1}$ | (n × 1) state vectors |
| $A_t$ | (n × n) state transition matrix |
| $u_t$ | (m × 1) control vector |
| $B_t$ | (n × m) (control-state) matrix |
| $\varepsilon_t$ | (n × 1) process noise (state transition randomness);<br>Gaussian with zero mean and covariance $R_t$ |

# KF | Linear System Description

Linear discrete-time system (process):

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \quad (4)$$

→ defines the state transition probability $p(x_t|u_t, x_{t-1})$

$$z_t = C_t x_t + \delta_t \quad (5)$$

→ defines the measurement probability $p(z_t|x_t)$

| | |
|---|---|
| $x_t, x_{t-1}$ | (n × 1) state vectors |
| $A_t$ | (n × n) state transition matrix |
| $u_t$ | (m × 1) control vector |
| $B_t$ | (n × m) (control-state) matrix |
| $\varepsilon_t$ | (n × 1) process noise (state transition randomness); Gaussian with zero mean and covariance $R_t$ |
| $z_t$ | (k × 1) measurement vector |
| $C_t$ | (k × n) (measurement-state) matrix |
| $\delta_t$ | (k × 1) measurement noise; Gaussian with zero mean and covariance $Q_t$ |

Linear discrete-time system (process):

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t \qquad (4)$$

$$z_t = C_t x_t + \delta_t \qquad (5)$$

(4) → defines the state transition probability $p(x_t | u_t, x_{t-1})$

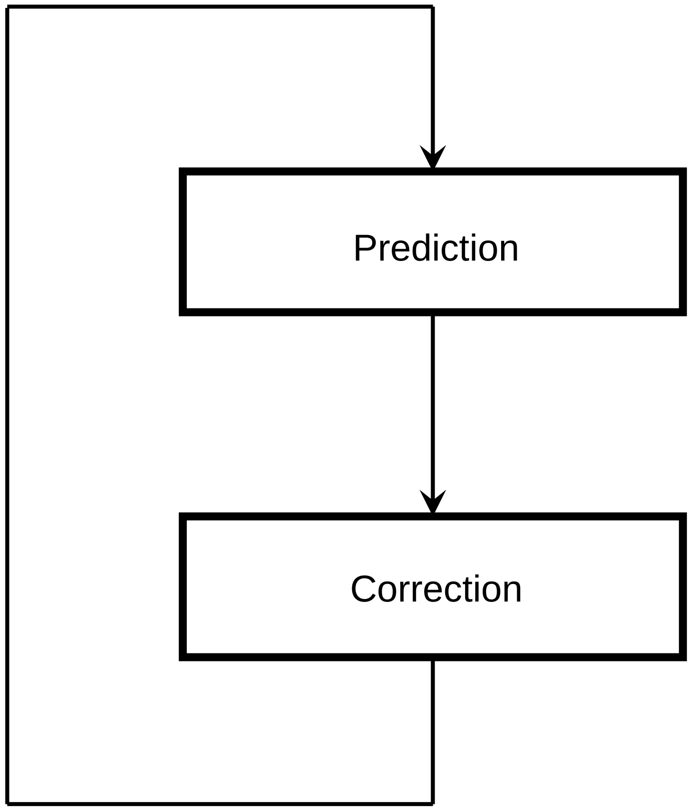
(5) → defines the measurement probability $p(z_t | x_t)$

- *State transition probability* is obtained by substituting (4) into multivariate norm. distr. (2), whereas $\mu_t = A_t x_{t-1} + B_t u_t$ and $\Sigma_t = R_t$:

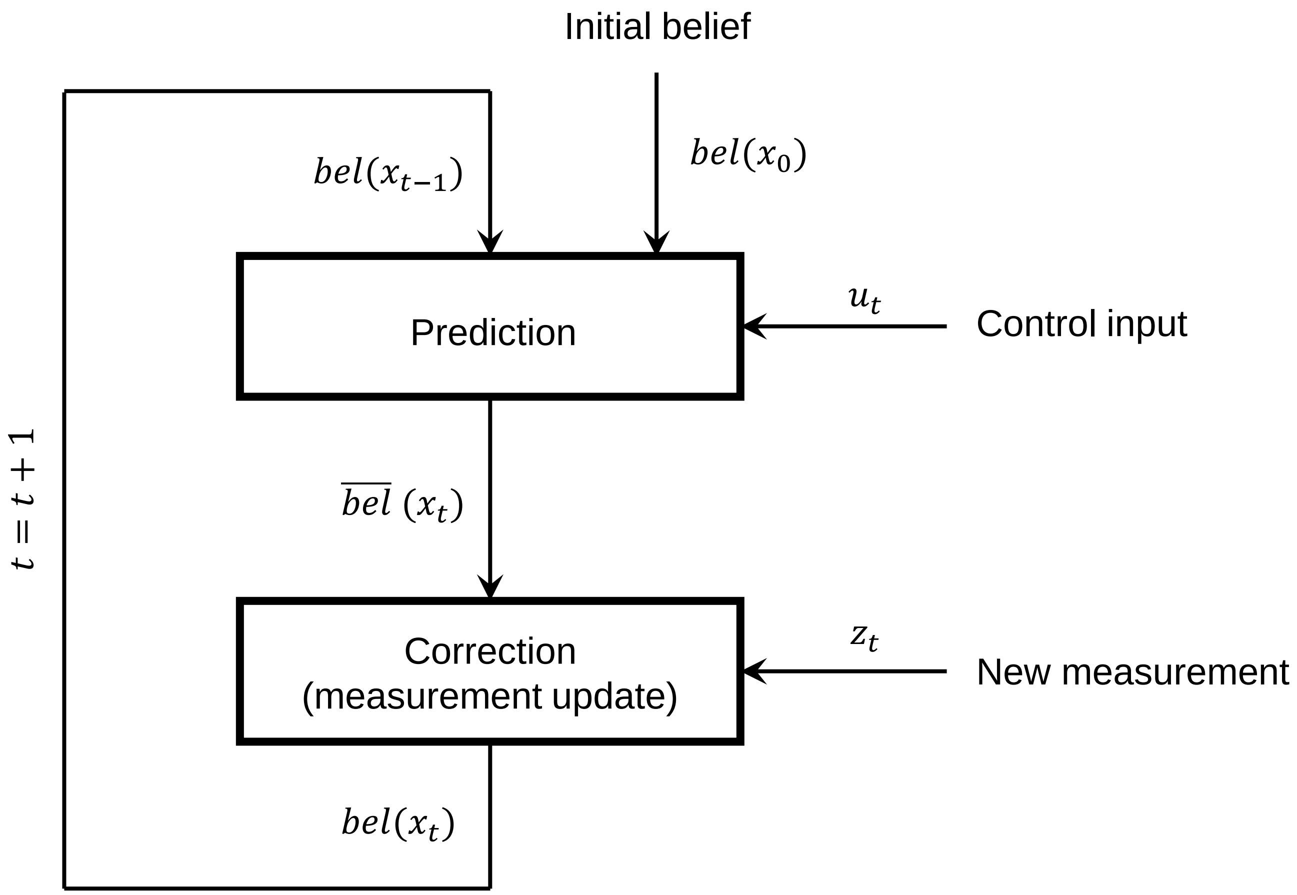
$$p(x_t | u_t, x_{t-1}) = \det(2\pi R_t)^{-\frac{1}{2}} \exp\{-\frac{1}{2}(x_t - A_t x_{t-1} + B_t u_t)^T {R_t}^{-1} (x_t - A_t x_{t-1} + B_t u_t)\} \qquad (6)$$

- *Measurement probability* is obtained by substituting (5) into multivariate norm. distr. (2), whereas $\mu_t = C_t x_t$ and $\Sigma_t = Q_t$:

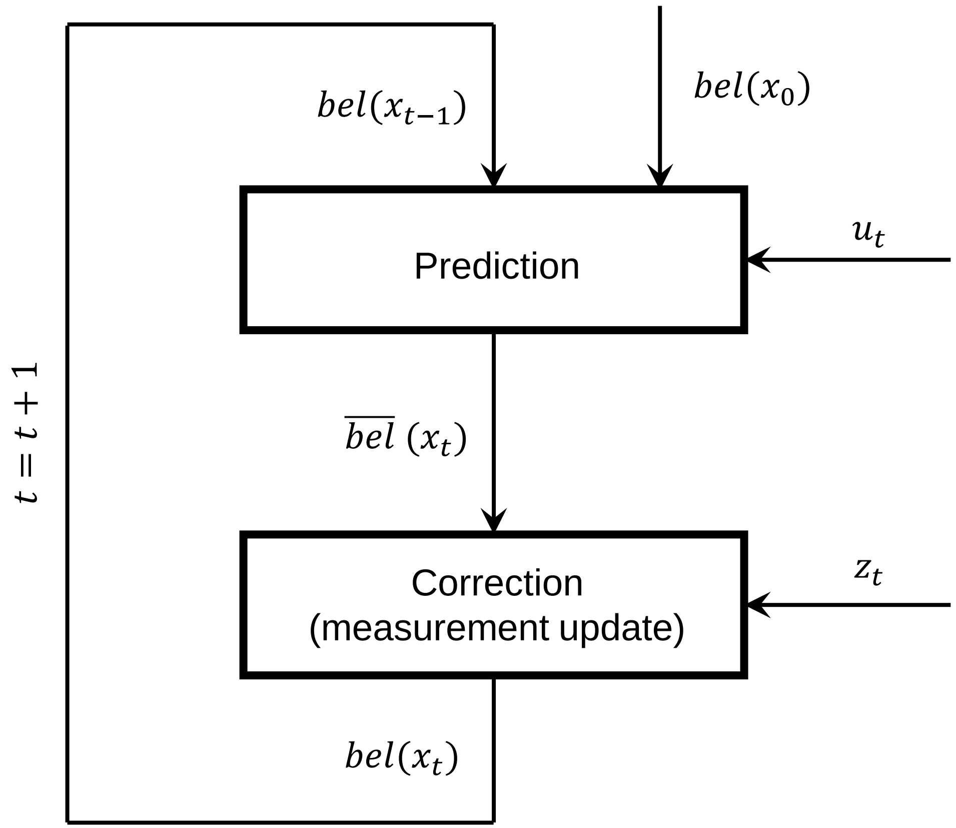
$$p(z_t | x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp\{-\frac{1}{2}(z_t - C_t x_t)^T {Q_t}^{-1} (z_t - C_t x_t)\} \qquad (7)$$


Prediction
Correction

# KF | Algorithm



1. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
2. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$
3. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
4. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
5. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$



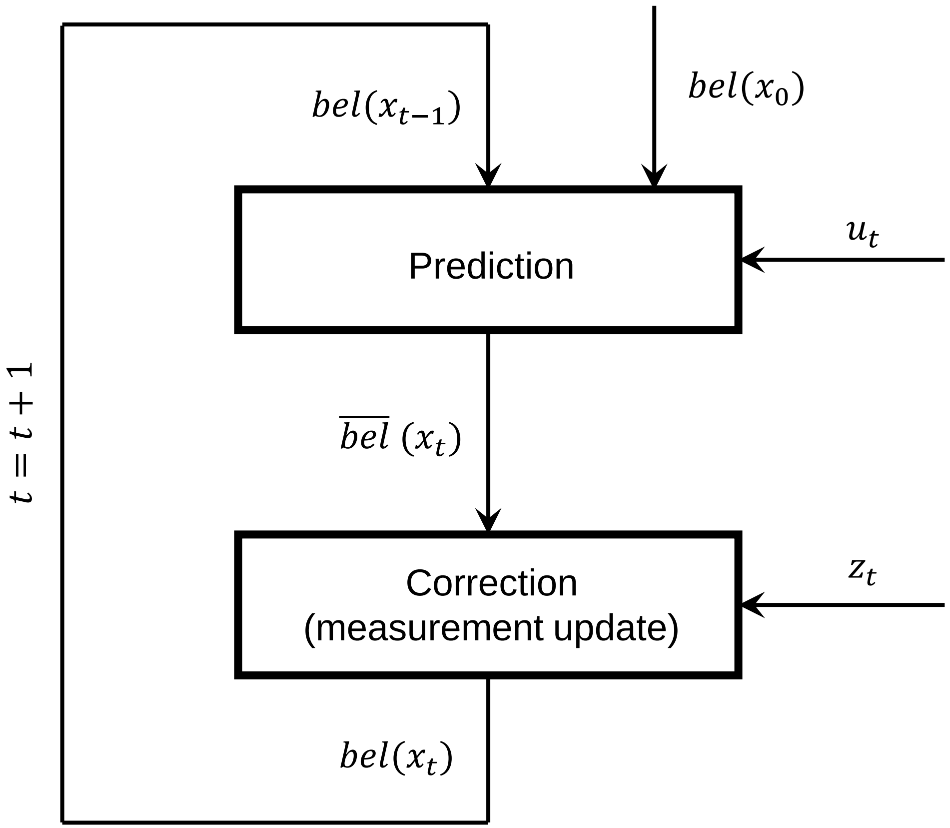
# KF | Algorithm

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

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

3. $K_t = \bar{\Sigma}_t C_t^T (C_t\,\bar{\Sigma}_t C_t^T + Q_t)^{-1}$
4. $\mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t)$
5. $\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$

Correction $bel(x_t)$




| | |
|---|---|
| $\bar{\mu}_t$, $\bar{\Sigma}_t$, $\overline{bel}\,(x_t)$ | Overline means *apriori* (before observations) |
| $\mu_t$, $\Sigma_t$, $bel(x_t)$ | No overline means *aposteriori* (after observations) |
| $K_t$ | Kalman Gain |

# KF | Summary

- Kalman filter represents the belief $bel(x_t)$ by the mean $\mu_t$ and the covariance $\Sigma_t$.
- In the prediction step, the mean $\bar{\mu}_t$ is updated using the deterministic part of the state transition function (4).
- The posterior mean $\mu_t$ is based on the predicted mean $\bar{\mu}_t$ and the difference between measurement $z_t$ and predicted measurement $C_t\bar{\mu}_t$ multipied by the Kalman gain $K_t$.

# KF | Summary

- Kalman filter represents the belief $bel(x_t)$ by the mean $\mu_t$ and the covariance $\Sigma_t$.
- In the prediction step, the mean $\bar{\mu}_t$ is updated using the deterministic part of the state transition function (4).
- The posterior mean $\mu_t$ is based on the predicted mean $\bar{\mu}_t$ and the difference between measurement $z_t$ and predicted measurement $C_t\bar{\mu}_t$ multipied by the Kalman gain $K_t$.

- To keep the posterior Gaussian, ***three conditions must be met***:
  1) The initial belief must be normally distributed.
  2) State transition function must be linear in its arguments.
  3) Measurement function must be linear in its arguments.

1)

$$bel(x_0) \sim \mathcal{N}(x_0; \mu_0, \Sigma_0)$$

2)

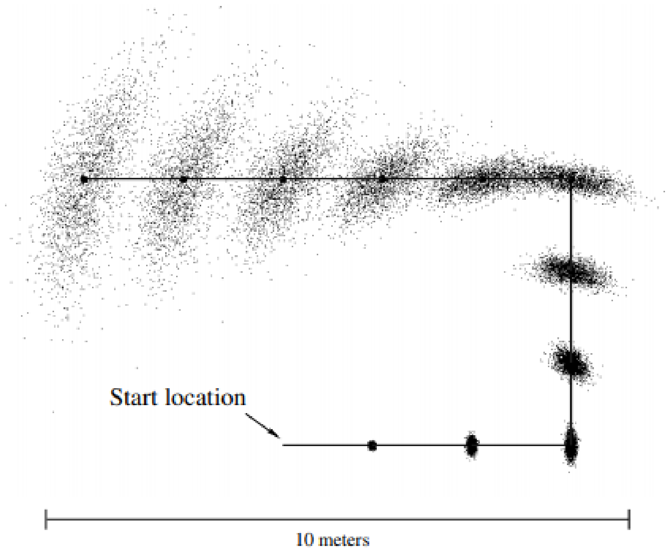
$$x_t = \underbrace{A_t x_{t-1} + B_t u_t}_{\text{Linear}} + \underbrace{\varepsilon_t}_{\text{Gaussian noise}}$$

3)

$$z_t = \underbrace{C_t x_t}_{\text{Linear}} + \underbrace{\delta_t}_{\text{Gaussian noise}}$$

Since the Gaussian is unimodal:

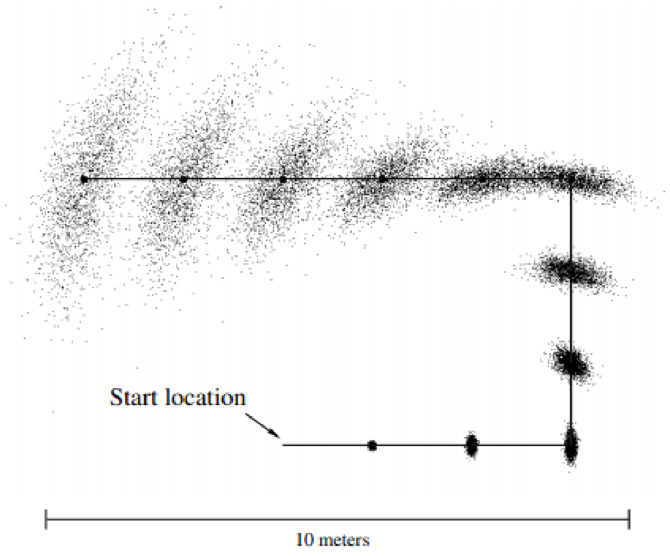
- KF is ***suitable*** for ***local localization problems*** in robotics, e.g.for tracking problems with known initial pose. The belief represents the estimate of the true state with a small uncertainty.



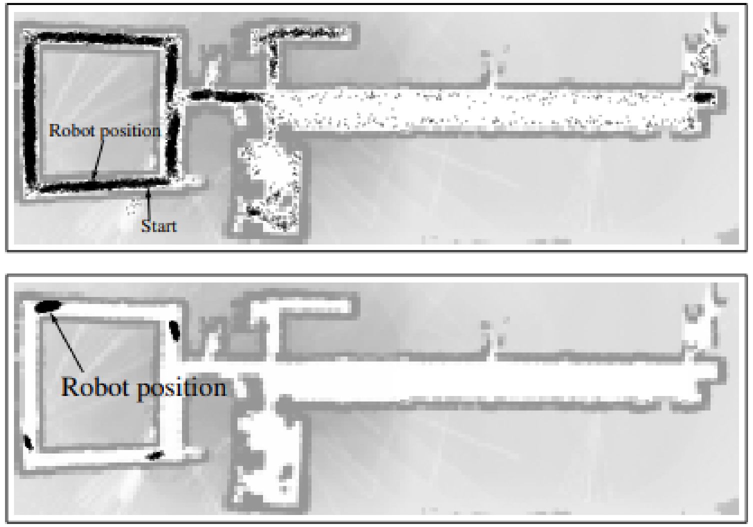

Position tracking (without sensing) [PR]

Since the Gaussian is unimodal:

- KF is ***suitable*** for ***local localization problems*** in robotics, e.g.for tracking problems with known initial pose. The belief represents the estimate of the true state with a small uncertainty.
- KF is ***not suitable*** for ***global localization problems*** in robotics, where many hypotheses exist, e.g. for robot localization in the known map with unknown initial pose.




Position tracking (without sensing) [PR]




Global localization (Particle filter) [PR]

# KF | Example – Robot in 1D

- Robot is moving in 1 dimension (horizontal axis).
- A user controls its velocity; this process comprises Gaussian noise with std. dev. $\sigma = 0.8\ m/s$. The user applies the following sequence of controls: $5, 5, 5$.
- There is a sensor measuring robot's global position; the measurement comprises Gaussian noise with std. dev $\sigma = 0.9\ m$.
- The robot starts at time $0\ s$ at position $0\ m$; the initial belief is defined by $\mu = 0\ m$ and variance $\sigma^2 = 0.5\ m$.
- The iteration period is $1\ s$. Use the KF to estimate robot's position at $\mathrm{t} = 3\ s$.

# KF | Example – Robot in 1D

- Robot is moving in 1 dimension (horizontal axis).
- A user controls its velocity; this process comprises Gaussian noise with std. dev. $\sigma = 0.8\ m/s$. The user applies the following sequence of controls: $5, 5, 5$.
- There is a sensor measuring robot's global position; the measurement comprises Gaussian noise with std. dev $\sigma = 0.9\ m$.
- The robot starts at time $0\ s$ at position $0\ m$; the initial belief is defined by $\mu = 0\ m$ and variance $\sigma^2 = 0.5\ m$.
- The iteration period is $1\ s$. Use the KF to estimate robot's position at $\mathrm{t} = 3\ s$.

---

The linear system:

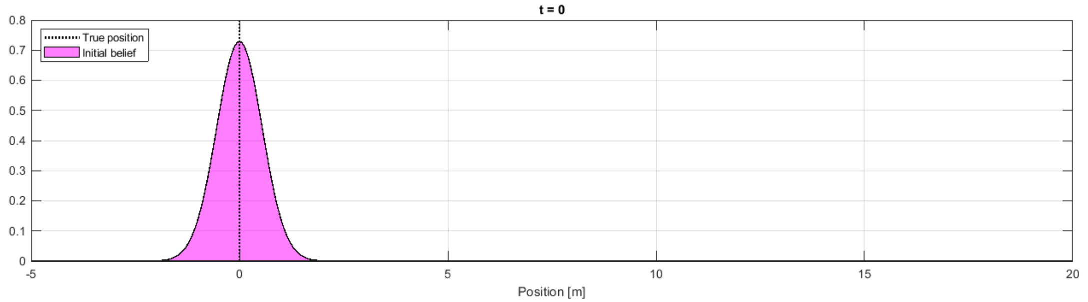
$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$
$$z_t = C_t x_t + \delta_t$$

State $x$ – (1 × 1)

Measurement $z$ – (1 × 1)

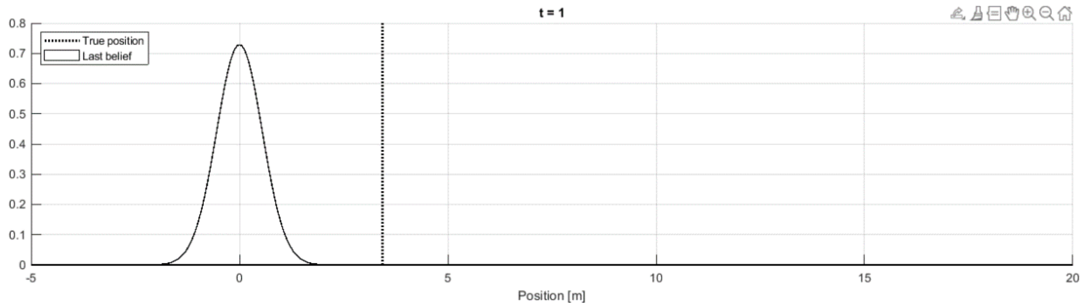
$$A = 1 \qquad \mu_0 = 0$$
$$B = dt \qquad \Sigma_0 = 0.5$$
$$C = 1 \qquad u_1 = 5$$
$$R = 0.8^2 \qquad u_2 = 5$$
$$Q = 0.9^2 \qquad u_3 = 5$$

# KF | Example – Robot in 1D




The initial position and belief $bel(x_0)$.

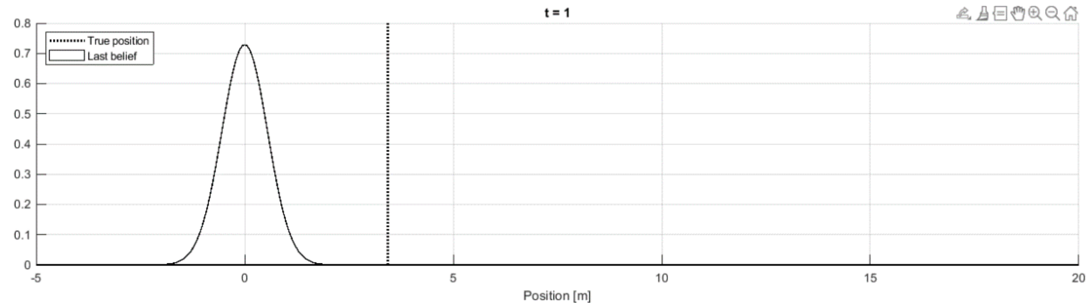
# KF | Example – Robot in 1D

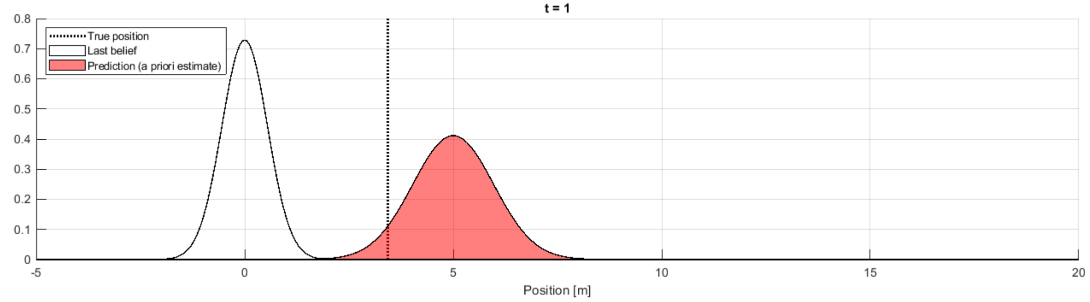



The real motion of the robot at $t = 1$.

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

# KF | Example – Robot in 1D




The real motion of the robot at $t = 1$.

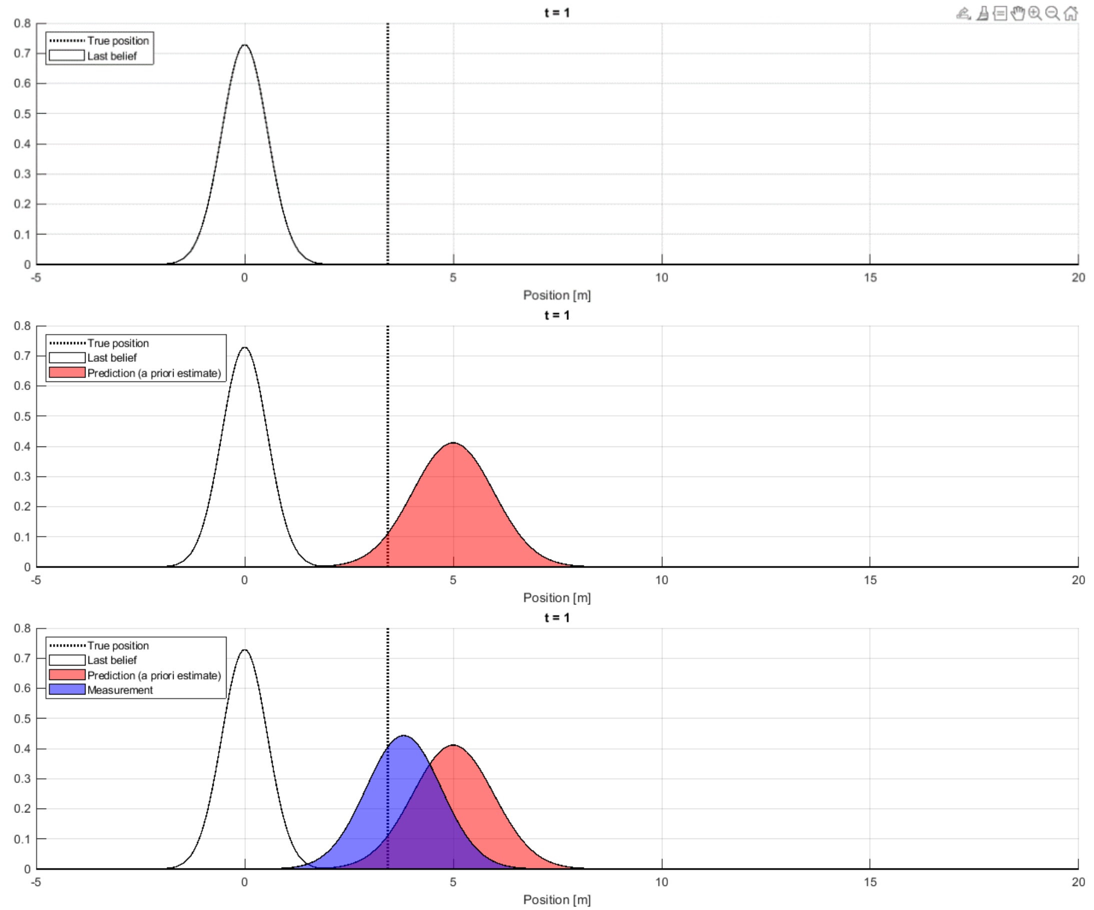
$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

KF prediction at $t = 1$.

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

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

# KF | Example – Robot in 1D




The real motion of the robot at $t = 1$.

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

KF prediction at $t = 1$.

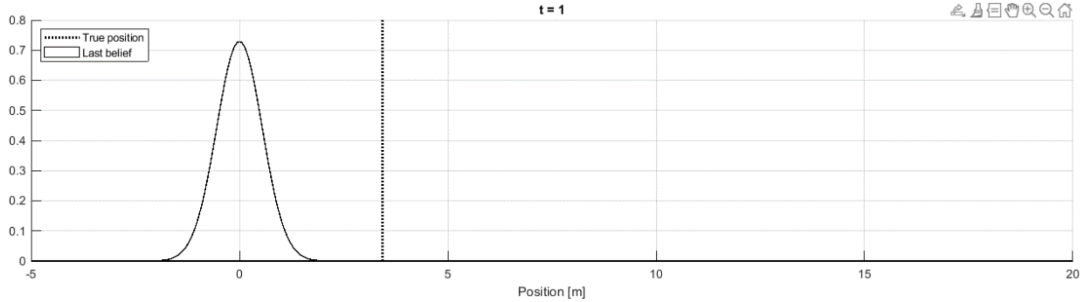
$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$

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

The real measuremen at $t = 1$.

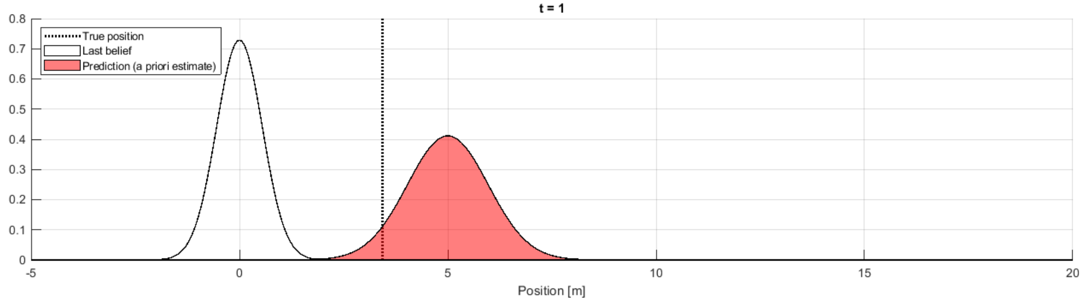
$$z_t = C_t x_t + \delta_t$$

# KF | Example – Robot in 1D




The real motion of the robot at $t = 1$.

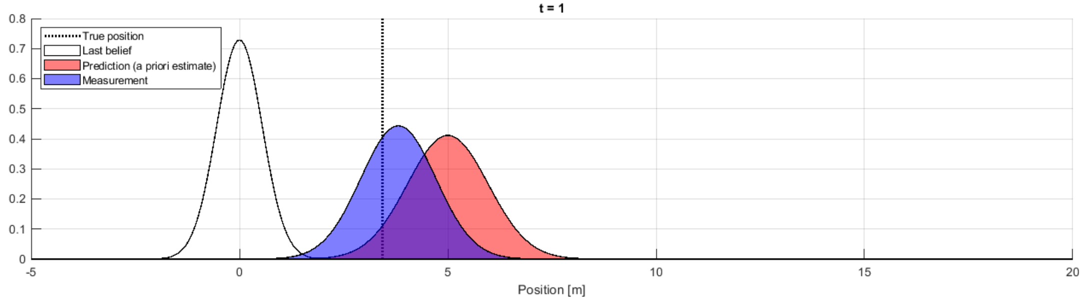
$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$




KF prediction at $t = 1$.

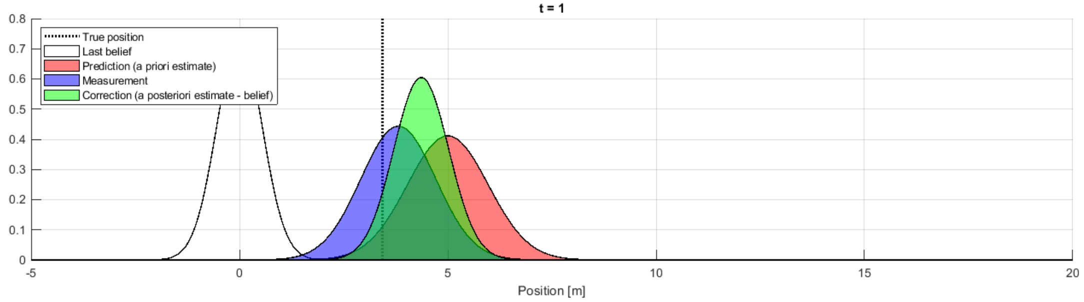
$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$$

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




The real measuremen at $t = 1$.

$$z_t = C_t x_t + \delta_t$$



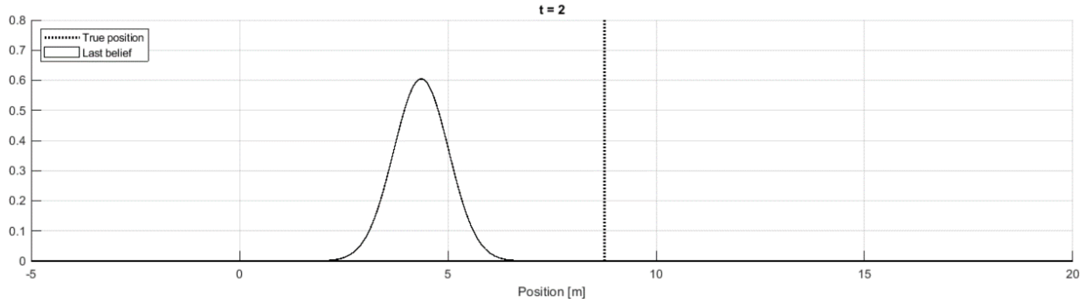

KF correction at $t = 1$.

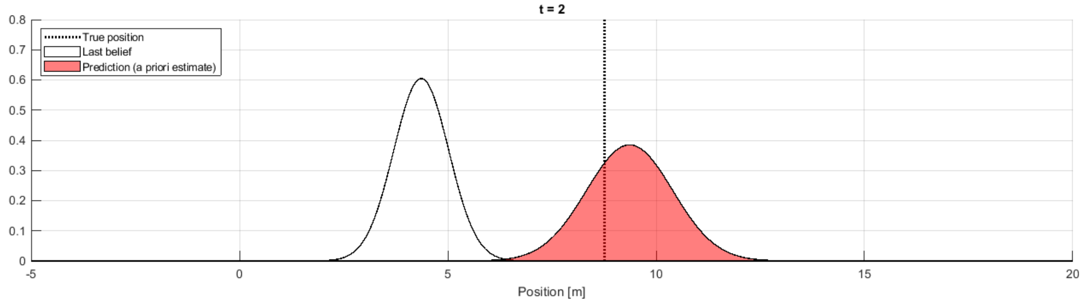
$$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

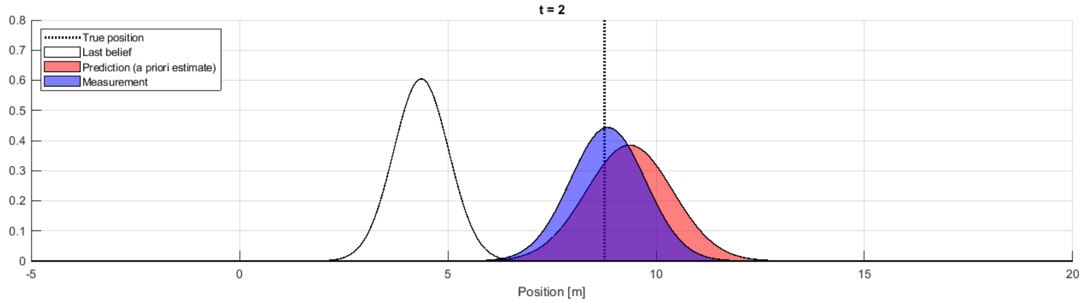
$$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$$

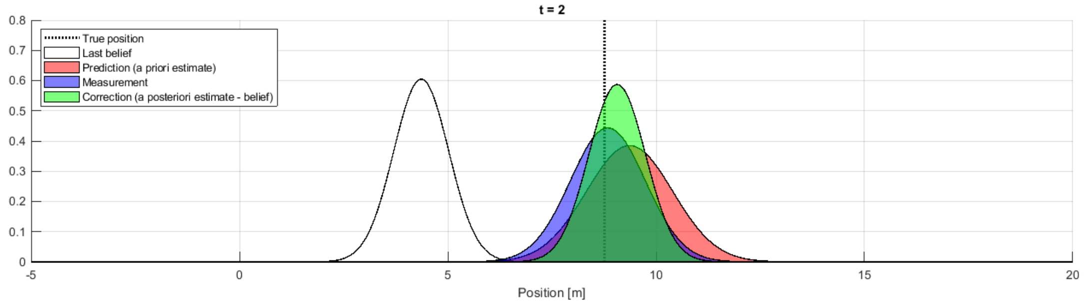
$$\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$$

# KF | Example – Robot in 1D

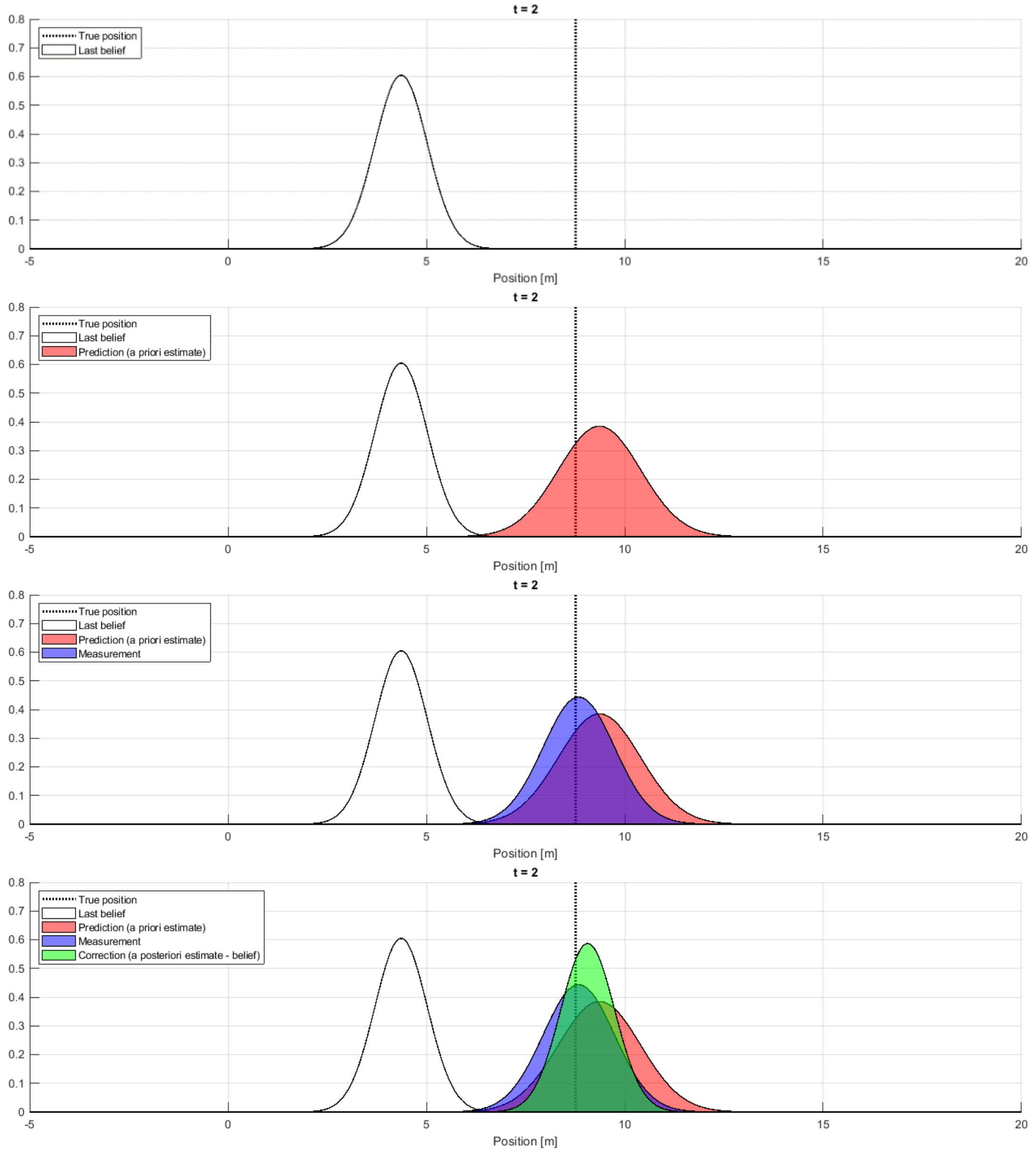


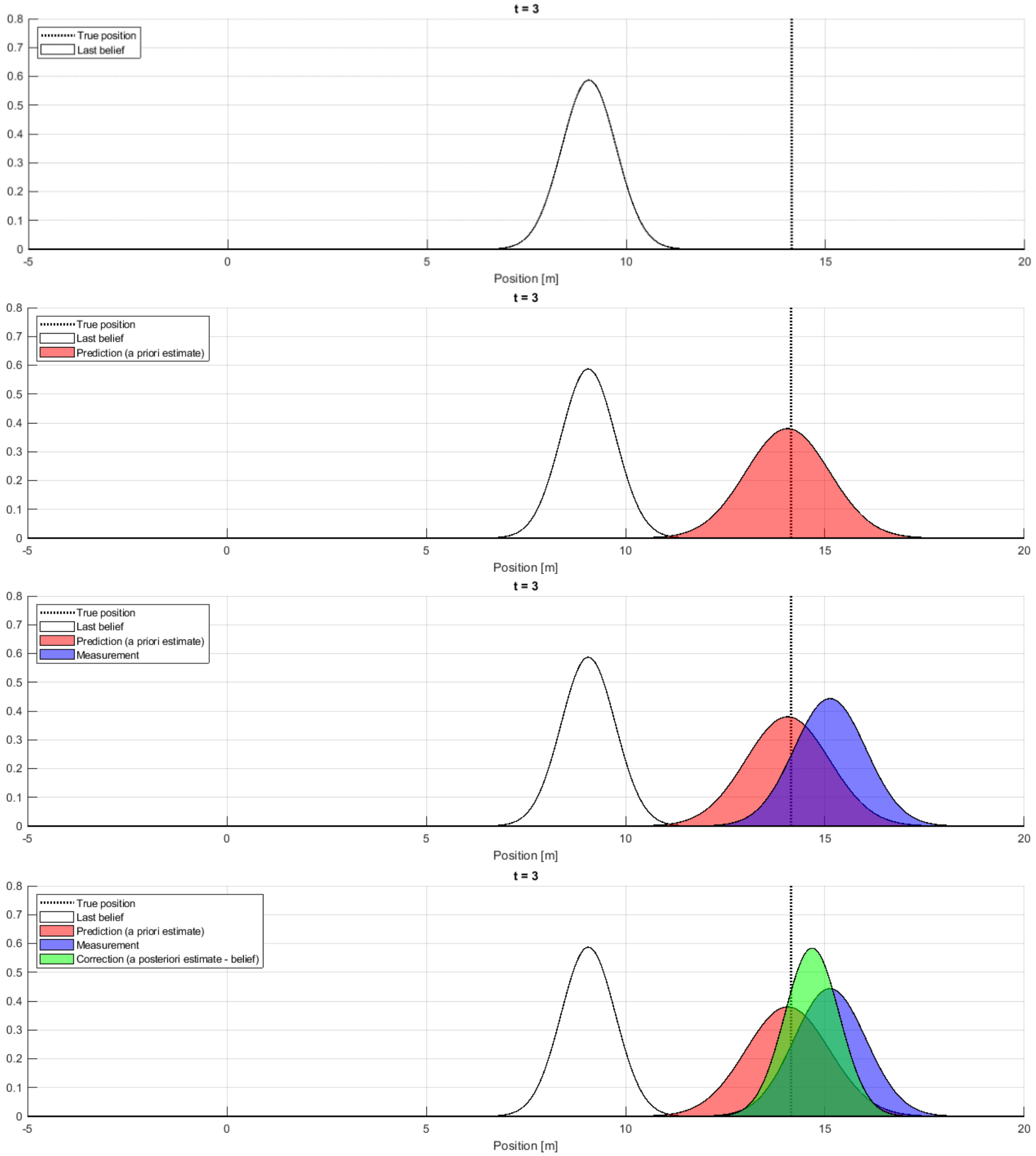



# KF | Example – Robot in 1D



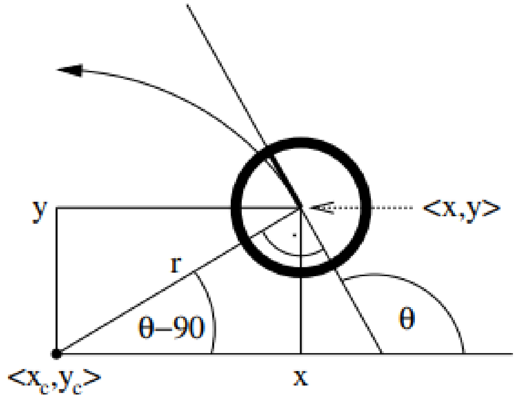



# Extended Kalman Filter (EKF)

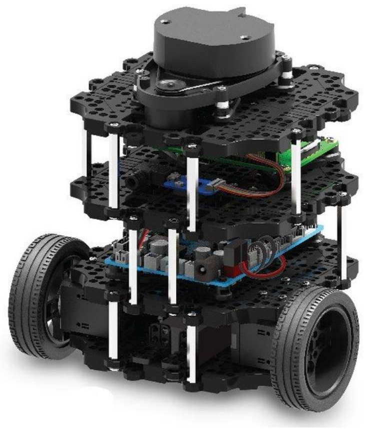
State estimator for nonlinear systems

# EKF | Introduction

- *KF* is optimal for linear Gaussian systems.
- Most real-world systems are ***nonlinear***.
- Nonlinearity in both state transition and measurement.
- KF modifications for nonlinear systems:
  - ***Extended Kalman Filter*** (EKF)
  - ***Unscented Kalman Filter*** (UKF)
- EKF:
  - The standard algorithm for state estimation in *navigation systems* and *robotics*.
  - *Not* an optimal estimator for nonlinear systems.



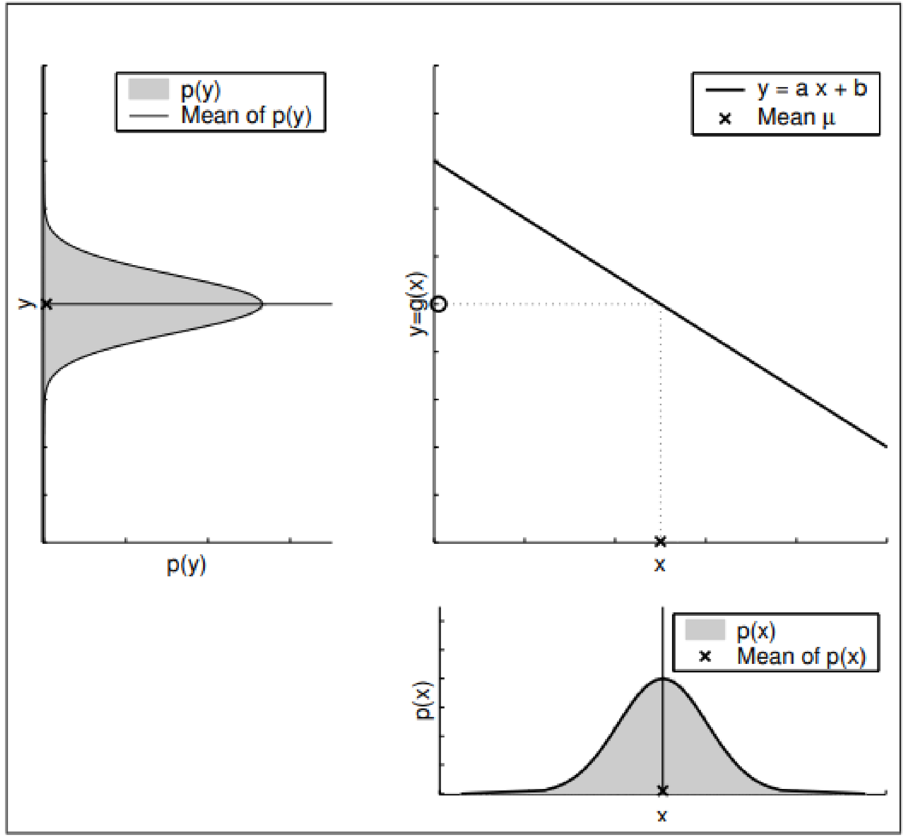

Differential wheeled robot moving with constant velocities $v$ and $\omega$ and starting at $(x\ y\ \theta)^T$ moves on a circular trajectory [PR].



TurtleBot3 [1]

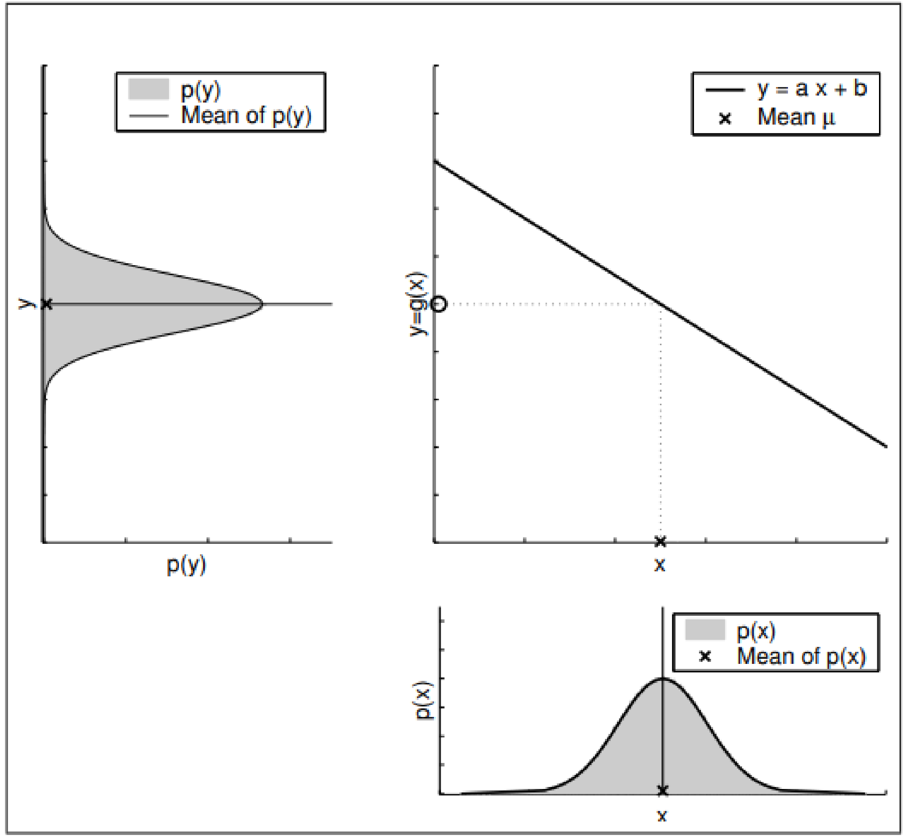
[1] https://www.turtlebot.com/

Linear function



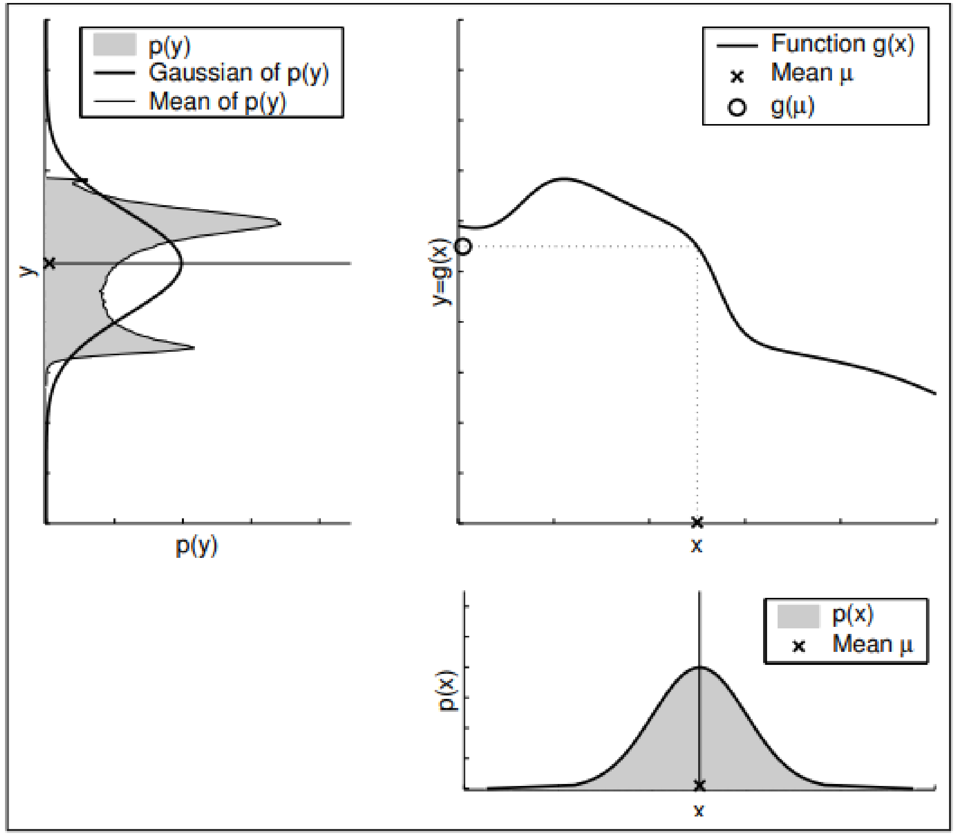

[PR]

Linear function




[PR]

Nonlinear function




[PR]

# EKF | Nonlinear System Description

Nonlinear discrete-time system (process):

$$x_t = g(u_t, x_{t-1}) + \varepsilon_t \quad (8)$$ → defines the state transition probability $p(x_t | u_t, x_{t-1})$

$$z_t = h(x_t) + \delta_t \quad (9)$$ → defines the measurement probability $p(z_t | x_t)$

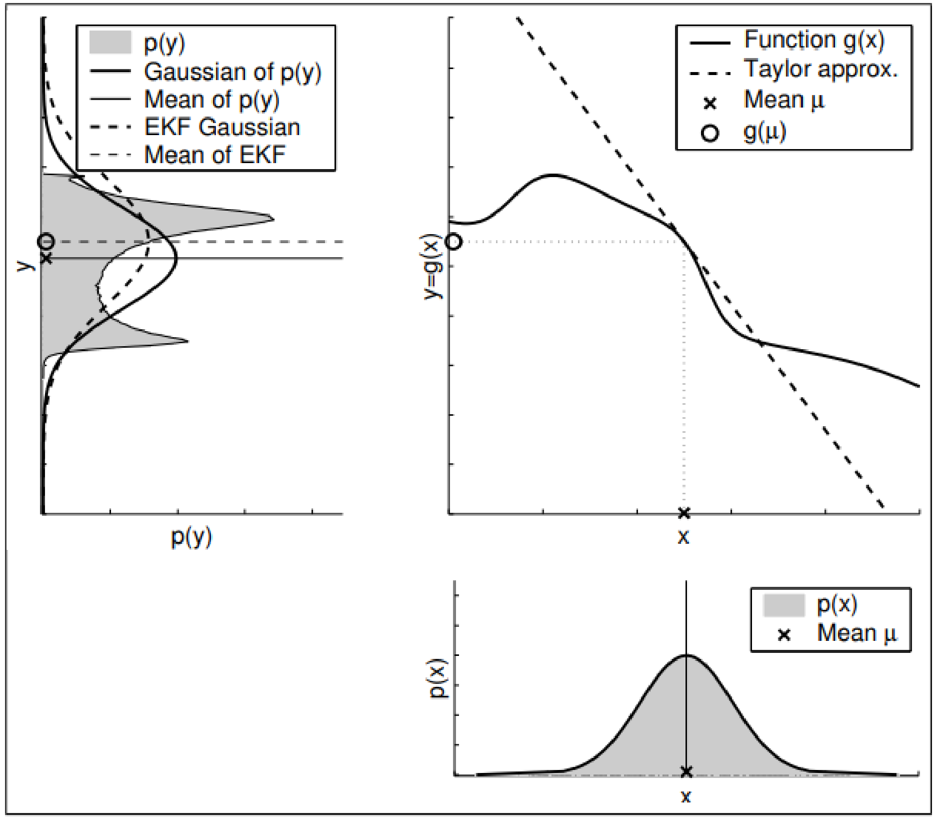
| | |
|---|---|
| $x_t, x_{t-1}$ | (n × 1) state vectors |
| $u_t$ | (m × 1) control vector |
| $g$ | Nonlinear state transition function |
| $\varepsilon_t$ | (n × 1) process noise (state transition randomness); Gaussian with zero mean and covariance $R_t$ |
| $z_t$ | (k × 1) measurement vector |
| $h$ | Nonlinear measurement function |
| $\delta_t$ | (k × 1) measurement noise; Gaussian with zero mean and covariance $Q_t$ |

# EKF | Algorithm

- Approximates nonlinear functions via first order Taylor expansion – ***linearization***.
- Linear approximation → slope computation → partial derivation → Jacobian
- Once $g$ and $h$ are linearized, the algorithm is equivalent to KF.
- Limitations:
  - Highly-nonlinear systems.
  - Noisy systems.
- Nonlinear system:

$$x_t = g(u_t, x_{t-1}) + \varepsilon_t$$

$$z_t = h(x_t) + \delta_t$$



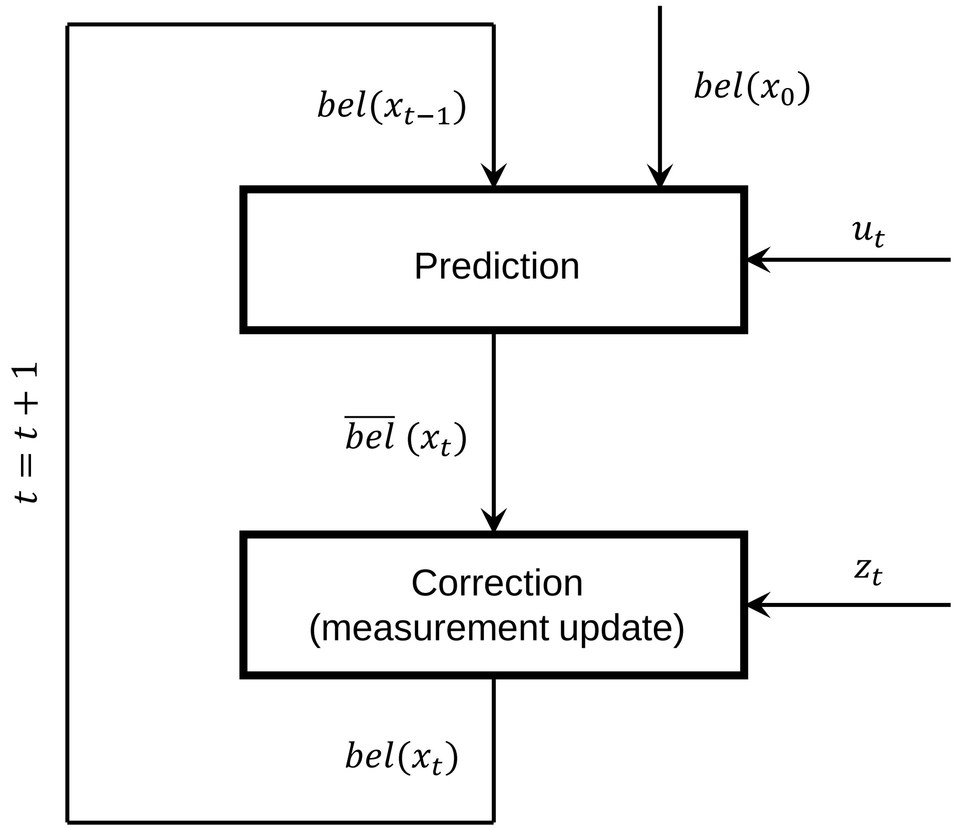

[PR]

# EKF | Algorithm

1. $\bar{\mu}_t = g(u_t, \mu_{t-1})$
2. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$
3. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$
4. $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$
5. $\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$

Steps 1–2: Prediction, $\overline{bel}\,(x_t)$

Steps 3–5: Correction, $bel(x_t)$




| | |
|---|---|
| $\bar{\mu}_t$ , $\bar{\Sigma}_t$ , $\overline{bel}\,(x_t)$ | Overline means *apriori* (before observations) |
| $\mu_t$ , $\Sigma_t$ , $bel(x_t)$ | No overline means *aposteriori* (after observations) |
| $K_t$ | Kalman Gain |

KF Algorithm

1. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
2. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$
3. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
4. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
5. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

vs.

EKF Algorithm

1. $\bar{\mu}_t = g(u_t, \mu_{t-1})$
2. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$
3. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$
4. $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$
5. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$

# EKF | Algorithm Comparison

| KF Algorithm | | EKF Algorithm |
|---|---|---|
| 1. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$ | | 1. $\bar{\mu}_t = g(u_t, \mu_{t-1})$ |
| 2. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$ | | 2. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$ |
| 3. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$ | vs. | 3. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$ |
| 4. $\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t)$ | | 4. $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$ |
| 5. $\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$ | | 5. $\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$ |

$$A_t \mu_{t-1} + B_t u_t \longleftrightarrow g(u_t, \mu_{t-1})$$

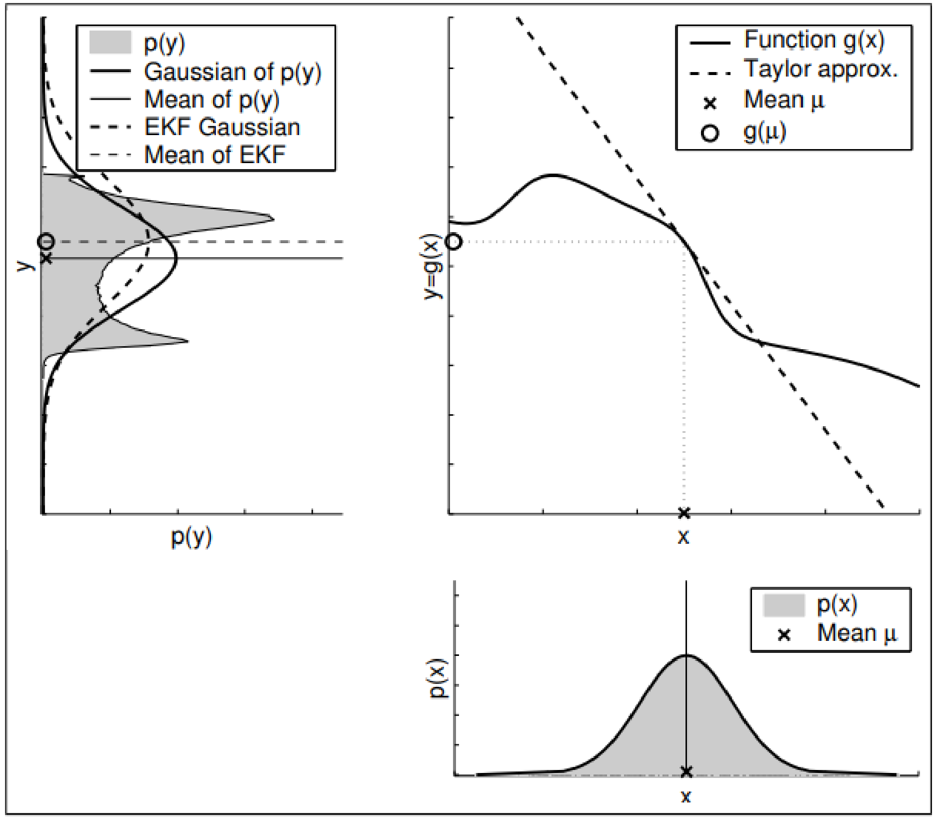
$$A_t \longleftrightarrow G_t$$

$$C_t \bar{\mu}_t \longleftrightarrow h(\bar{\mu}_t)$$

$$C_t \longleftrightarrow H_t$$

- *Jacobian matrix* – the matrix of first-order partial derivatives of a vector-valued function.
- $G$ and $H$ matrices pose Jacobians matrices of the state transition function $g(x)$ and measurement function $h(x)$, respectively.

$$g(x) = \begin{bmatrix} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{bmatrix} \qquad G = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial x_1} & \frac{\partial g_m}{\partial x_2} & \cdots & \frac{\partial g_m}{\partial x_n} \end{bmatrix}$$

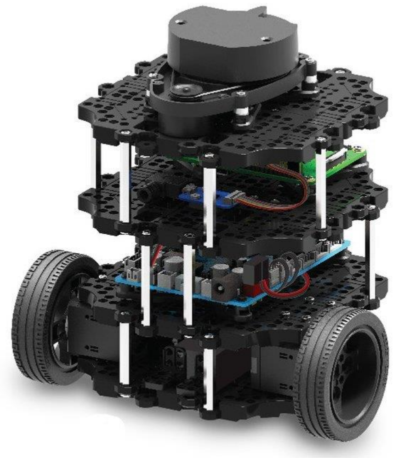



[PR]

- Differential wheeled robot:
  - State comprises position and heading: $x = [x, y, \theta]^T$.
  - Control comprises linear/angular speed: $u = [v, \omega]^T$.

- The relation between the control and state is nonlinear:

$$x_t = x_{t-1} + \cos\theta_{t-1}\, v_t\, \Delta_t$$
$$y_t = y_{t-1} + \sin\theta_{t-1}\, v_t \Delta_t$$
$$\theta_t = \theta_{t-1} + \omega_t \Delta_t$$



# EKF | Example – Differential Drive

- EKF prediction step:

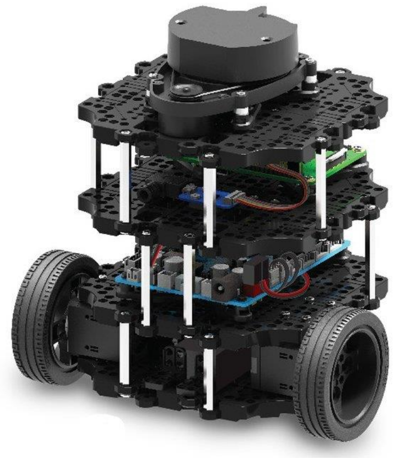
$$\bar{\mu}_t = g(u_t, \mu_{t-1})$$

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

---

$$\bar{\mu}_t = g(u_t, \mu_{t-1}) = \begin{vmatrix} g_1(u_t, \mu_{t-1}) \\ g_2(u_t, \mu_{t-1}) \\ g_3(u_t, \mu_{t-1}) \end{vmatrix} = \begin{vmatrix} x_{t-1} + \cos\theta_{t-1}\, v\Delta_t \\ y_{t-1} + \sin\theta_{t-1}\, v\Delta_t \\ \theta_{t-1} + \omega\Delta_t \end{vmatrix}$$

$$G = \begin{vmatrix} \frac{\partial g_1}{\partial x} & \frac{\partial g_1}{\partial y} & \frac{\partial g_1}{\partial \theta} \\ \frac{\partial g_2}{\partial x} & \frac{\partial g_2}{\partial y} & \frac{\partial g_2}{\partial \theta} \\ \frac{\partial g_3}{\partial x} & \frac{\partial g_3}{\partial y} & \frac{\partial g_3}{\partial \theta} \end{vmatrix} = \begin{vmatrix} 1 & 0 & -\sin\theta_{t-1}\, v_t \Delta_t \\ 0 & 1 & \cos\theta_{t-1}\, v_t\, \Delta_t \\ 0 & 0 & 1 \end{vmatrix}$$



# Supporting Materials

- *Probabilistic Robotics* book, chapters 3.1 – 3.3 [PR]
- KF basics by *R. Faragher* [1]: https://ieeexplore.ieee.org/document/6279585
- Udacity course *Artificial Intelligence for Robotics*: https://classroom.udacity.com/courses/cs373
- *Cyrill Stachniss* KF & EKF presentation (1hr 13min): https://www.youtube.com/watch?v=E-6paM_Iwfc&t=558s

[1] FARAGHER, R., 2012. Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation. *IEEE Signal Processing Magazine*. 2012. Vol. 29, no. 5, p. 128–132. DOI 10.1109/MSP.2012.2203621.

Petr Gabrlik
petr.gabrlik@ceitec.vutbr.cz

Brno University of Technology
Faculty of Electrical Engineering and Communication
Department of Control and Instrumentation

Robotics and AI Research Group