Robotics

PS08 – Solutions

Andreas Birk

Constructor University

Part 8: Probabilistic Localization

Problem 1

Given the Gaussian $N(\hat{x}, C)$ representing the estimate of a system state $\hat{x} = (x_1, x_2)$ and its related uncertainty. At time t, $\hat{x}_t$ and C_t are as follows:

$$\hat{x} = (2.1, 3.7)\ , C_t = \begin{pmatrix} 0.021 & 0.004 \\ 0.004 & 0.038 \end{pmatrix}$$

The system evolves according to the following function $F()$:

$$F(x) = \begin{pmatrix} sin(x_1) \cdot x_2 \\ cos(x_1) + x_2^2 \end{pmatrix}$$

Use the error propagation law to compute $\hat{x}_{t+1}$ and C_{t+1}.

Problem 1

new mean

$$\hat{x}_t = (2.1,3.7)^T, C_t = \begin{pmatrix} 0.021 & 0.004 \\ 0.004 & 0.038 \end{pmatrix}$$

$$F(x_1, x_2) = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} s x_1 \cdot x_2 \\ c x_1 + {x_2}^2 \end{pmatrix}$$

$$\hat{x}_{t+1} = F(\hat{x}_t) = F(2.1,3.7) = \begin{pmatrix} s(2.1) \cdot 3.7 \\ c(2.1) + 3.7^2 \end{pmatrix} = \begin{pmatrix} 3.193875 \\ 13.18515 \end{pmatrix}$$

Problem 1

new Covariance

$$F(x_1, x_2) = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} sx_1 \cdot x_2 \\ cx_1 + {x_2}^2 \end{pmatrix} \Longrightarrow J = DF(x_1, x_2) = \begin{pmatrix} cx_1 \cdot x_2 & sx_1 \\ -sx_1 & 2x_2 \end{pmatrix}$$

$$C_{t+1} = J \cdot C_t \cdot J^T$$

$$= \begin{pmatrix} c(2.1) \cdot 3.7 & s(2.1) \\ -s(2.1) & 2 \cdot 3.7 \end{pmatrix} \begin{pmatrix} 0.021 & 0.004 \\ 0.004 & 0.038 \end{pmatrix} \begin{pmatrix} c(2.1) \cdot 3.7 & -s(2.1) \\ s(2.1) & 2 \cdot 3.7 \end{pmatrix}$$

$$= \begin{pmatrix} 0.088688 & 0.218324 \\ 0.218324 & 2.045426 \end{pmatrix}$$

Problem 2

Given a simple system with a 1D state x that moves proportionally to a system input $u()$, concretely $x_k = x_{k-1} + 5\, u_{k-1}$. Its state, i.e., its 1D location, can be measured with a sensor that behaves linearly, i.e., $z(x) = 0.1\, x$. Both the motion and the measurement are noisy and can be modeled with zero-mean Gaussians with a variance of $Q = 0.2$, respectively $R = 0.3$.

The system starts at $k = 0$ in state $x = 0$ with no uncertainty. Use a Kalman filter to estimate the system states and the related variances for following inputs and measurements:

k	u_{k-1}	z_k
1	2.4	1.330
2	1.8	2.031
3	-3.1	-0.370
4	-2.7	-0.588

Problem 2

given: linear system with white Gaussian noise

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$
$$z_k = Hx_k + v_k$$

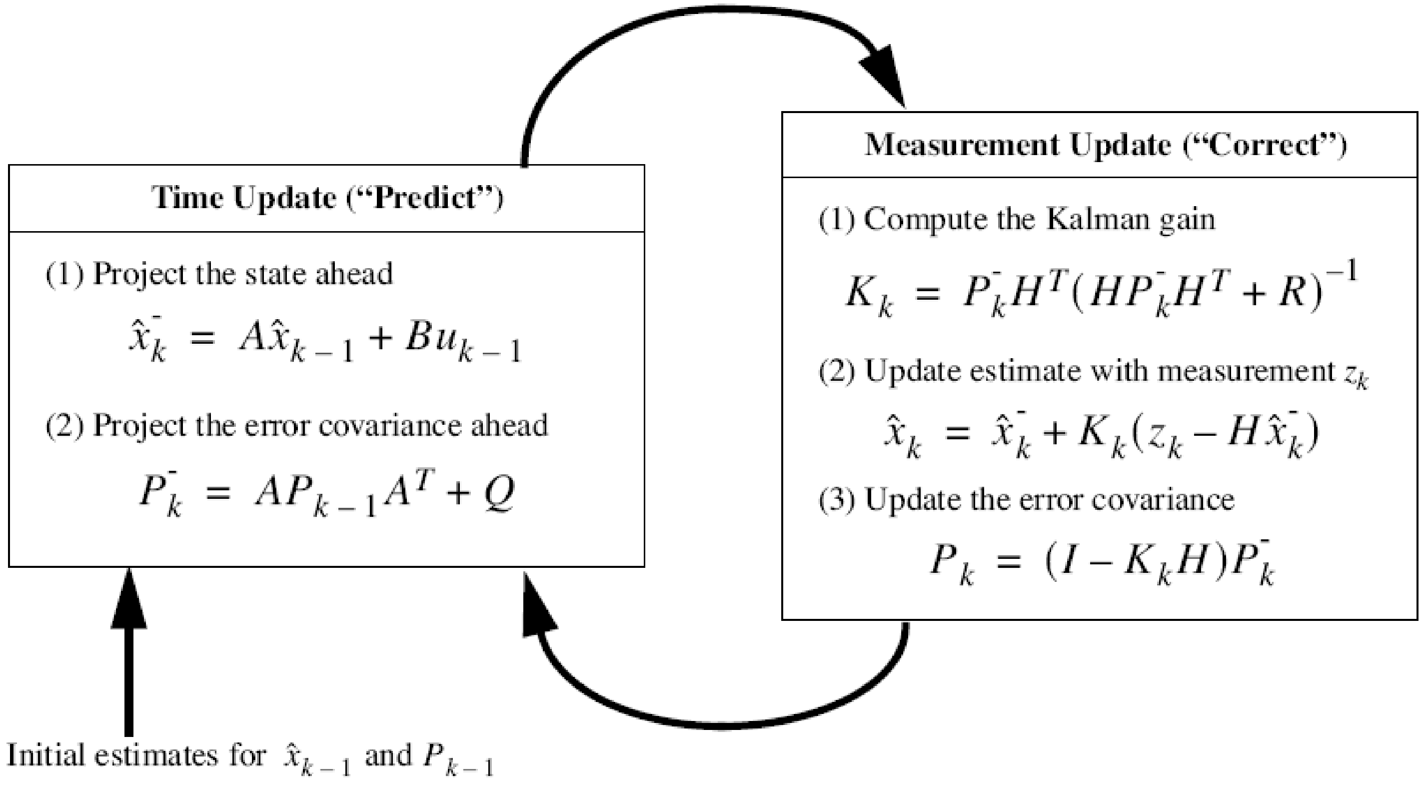
$$p(w) = N(0, Q), p(v) = N(0, R)$$

here:

$$A = 1, B = 5, H = 0.1$$
$$Q = 0.2, R = 0.3$$

Problem 2

Kalman Filter



$$A = 1, B = 5, H = 0.1, Q = 0.2, R = 0.3$$

PREDICT

$$\hat{x}_k^- = 1 \cdot x_{k-1} + 5 \cdot u_{k-1}$$
$$P_k^- = 1 \cdot P_{k-1} \cdot 1 + 0.2$$

$$K_k = P_k^- 0.1(0.1 P_k^- 0.1^T + 0.3)^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - 0.1\hat{x}_k^-)$$
$$P_k = (1 - K_k 0.1)P_k^-$$

CORRECT

Problem 2

$$\hat{x}_1^- = x_0 + 5 \cdot u_0$$
$$= 0 + 5 \cdot 2.4$$
$$P_1^- = P_0 + 0.2$$
$$= 0 + 0.2 = 0.2$$

k	u_{k-1}	z_k
1	2.4	1.330
2	1.8	2.031
3	-3.1	-0.370
4	-2.7	-0.588

$$K_1 = \frac{0.1 P_1^-}{0.01 P_1^- + 0.3}$$

$$= \frac{0.1 \cdot 0.2}{0.01 \cdot 0.2 + 0.3} = 0.066225$$

$$\hat{x}_1 = \hat{x}_1^- + K_1(z_1 - 0.1\hat{x}_1^-)$$
$$= 12 + 0.066225(1.330 - 0.1 \cdot 12)$$
$$= 12.009$$

$$P_1 = (1 - 0.1K_1)P_1^-$$
$$= (1 - 0.1 \cdot 0.066225)0.2$$
$$= 0.198675$$

Problem 2

k	x_k	x_k^-	u_{k-1}	z_k	P_k	P_k^-	K_k
0	0				0.000		
1	12.009	12.000	2.4	1.330	0.199	0.200	0.066
2	20.999	21.009	1.8	2.031	0.393	0.399	0.131
3	5.321	5.499	-3.1	-0.370	0.582	0.593	0.194
4	-8.121	-8.179	-2.7	-0.588	0.762	0.782	0.254

Problem 3

Given a simple non-linear system with a 1D state x that evolves with input $u()$ as follows $x_k = x_{k-1}^2 + sin(u_{k-1})$. Its state x can be measured with a sensor that also behaves non-linearly with $z(x) = x^3$. Both the motion and the measurement are noisy and can be modeled with zero-mean Gaussians with a variance of $Q = 0.2$, respectively $R = 0.3$.

The system starts at $k = 0$ in state $x = 0$ with no uncertainty. Use an Extended Kalman filter to estimate the system state and the related variance for input $u_0 = \pi/2$ and measurement $z_1 = 1.1$.

Problem 3

non-linear system with white Gaussian noise

$$x_k = {x^2}_{k-1} + \sin(u_{k-1}) + w_{k-1}$$
$$z_k = {x^3}_k + v_k$$

$$p(w) = N(0, Q), p(v) = N(0, R)$$
$$Q = 0.2, R = 0.3$$

$$f(x) = x^2 + \sin(u) \Rightarrow J_f = \frac{\partial f}{\partial x} = 2x$$
$$h(x) = x^3 \Rightarrow J_h = 3x^2$$

note: Jacobian wrt x,
u is a constant

Problem 3

Extended Kalman Filter (EKF)
linearization of update equations with Jacobians J_f and J_h of f and h

predictor step:
$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1})$$
$$P_k^- = J_f P_{k-1} {J_f}^T + Q$$

Kalman gain:
$$K_k = P_k^- {J_h}^T (J_h P_k^- {J_h}^T + R)^{-1}$$

corrector step:
$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - h(\hat{x}_k^-)\right)$$
$$P_k = (I - K_k J_h) P_k^-$$

Problem 3

predictor step:

$$\hat{x}_k^- = \hat{x}^2{}_{k-1} + \sin(\,u_{k-1})$$
$$P_k^- = 2x_{k-1}P_{k-1}(2x_{k-1})^T + 0.2$$

Kalman gain:

$$K_k = P_k^-(3(\hat{x}_k^-)^2)^T\left((3(\hat{x}_k^-)^2)P_k^-(3(\hat{x}_k^-)^2)^T + 0.3\right)^{-1}$$

corrector step:

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - (\hat{x}_k^-)^3)$$
$$P_k = \left(I - K_k(3(\hat{x}_k^-)^2)\right)P_k^-$$

Problem 3

$$x_0 = 0, u_0 = \frac{\pi}{2}, z = 1.1, P_0 = 0$$

predictor step: $\hat{x}_1^- = 0^2 + \sin(\pi/2) = 1$

$P_1^- = (2 \cdot 0) \cdot 0 \cdot (2 \cdot 0)^T + 0.2 = 0.2$

Kalman gain: $K_1 = 0.2(3 \cdot 1^2)^T \left((3 \cdot 1^2)0.2(3 \cdot 1^2)^T + 0.3\right)^{-1}$

$= 0.2857$

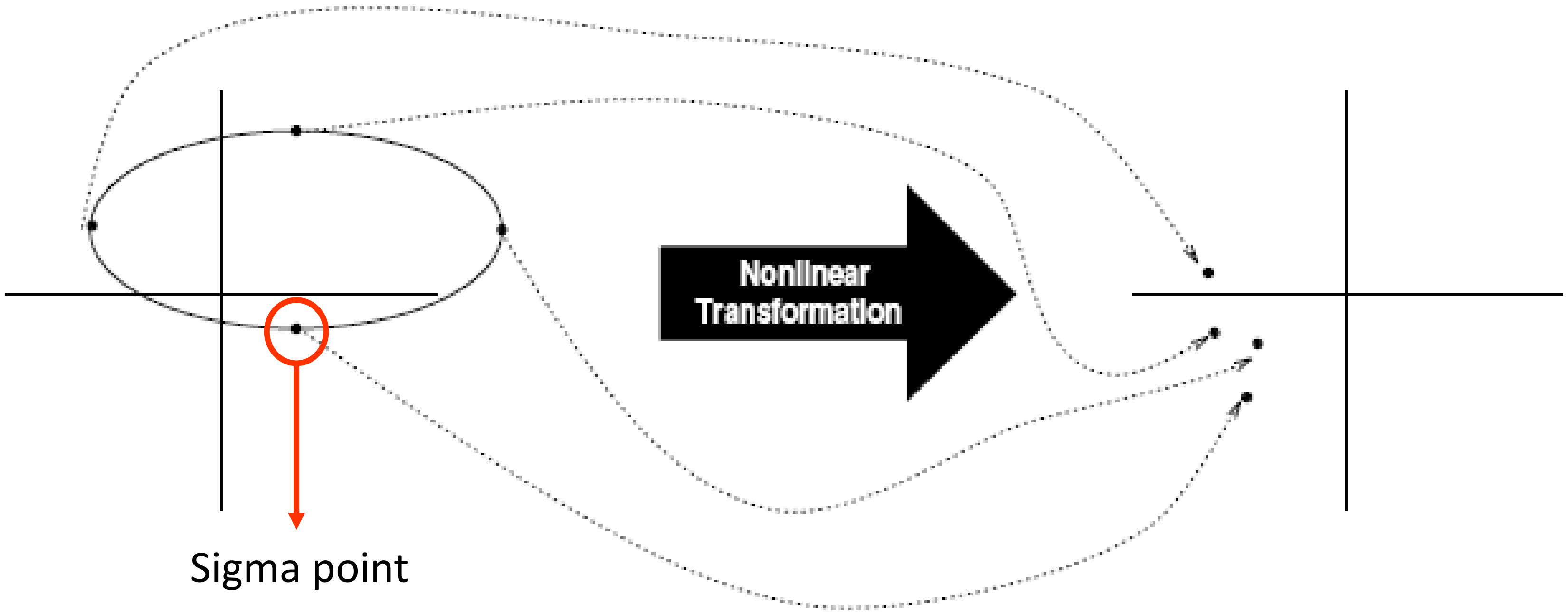
corrector step: $\hat{x}_1 = 1 + 0.2857(1.1 - 1) = 1.02857$

$P_1 = \left(1 - 0.2857(3 \cdot 1.02857^2)\right)0.2 = 0.018645$

Note: Unscented Kalman Filter (UKF)

basic idea:

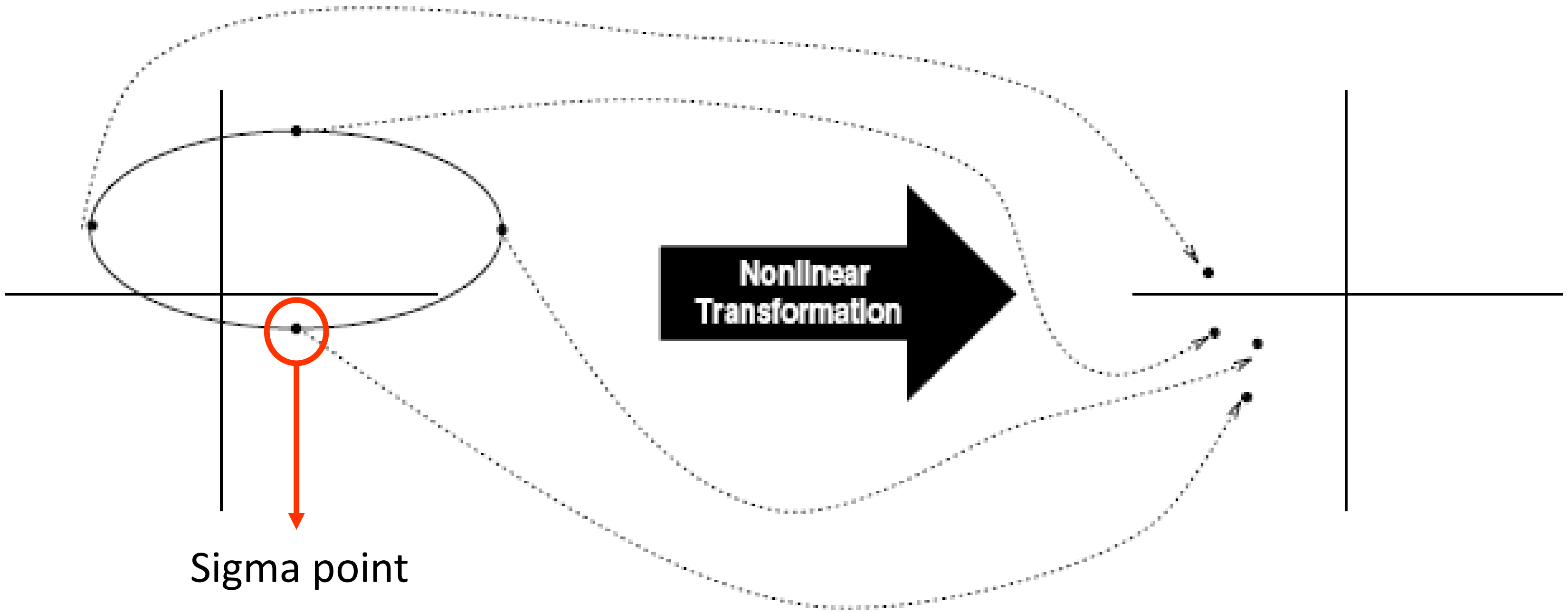
- do not linearize transformation
- but choose (few) sample points
- to represent mean and covariance



Note: Unscented Kalman Filter (UKF)

basic idea: sample points for mean and covariance

advantages:
- (can be) more accurate than EKF
- no need for Jacobians



Note: Particle Filter

alternative for both EKF and UKF

- can represent arbitrary distributions, not only Gaussians
- e.g., the estimate to be either at place $A = (x, y)^T$ or at a very different place $B = (x', y')^T$

but

- it needs many particles, i.e.,
- representations of the expected state
- and the computation of their evolution
- hence, computationally more expensive