VNU University of Engineering and Technology

ALGORITHMS FOR INTELLIGENT ROBOTS

Lecture 5
Robot Localization with the Extended Kalman Filter

Kalman Filter Summary

Time Update ("Predict")

(1) Project the state ahead

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$$

(2) Project the error covariance ahead

$$P_k^- = AP_{k-1}A^T + Q$$

Measurement Update ("Correct")

(1) Compute the Kalman gain

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

(2) Update estimate with measurement z_k

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$

(3) Update the error covariance

$$P_k = (I - K_k H)P_k^-$$

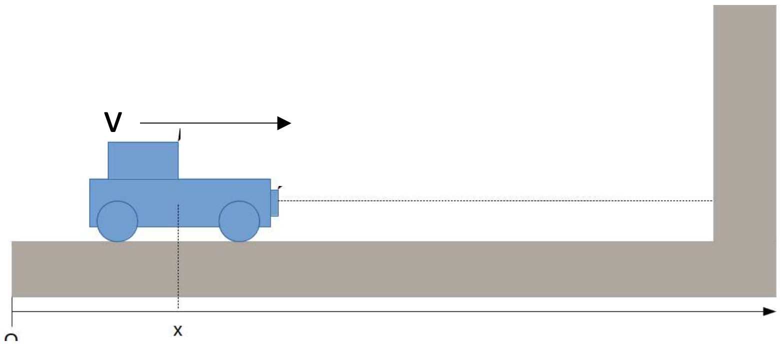
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Kalman filter example

- We'll consider a very simple example for understanding how the filter works.
- Let's consider a robot that moves in a single direction with a constant velocity of v = 1 m/s. Due to the imperfectness of the controller and actuator, the actual velocity of the robot is subjected to noise having the Gaussian distribution with zero mean and the variance of 0.2 m^2/s^2.
- The robot is equipped with a sensor that can measure the position of the robot. However, this sensor is also subjected to noise with zero mean and the variance of 0.1 m^2/s^2. With the sampling rate of 1 Hz, the sensor readings are given in the table below:

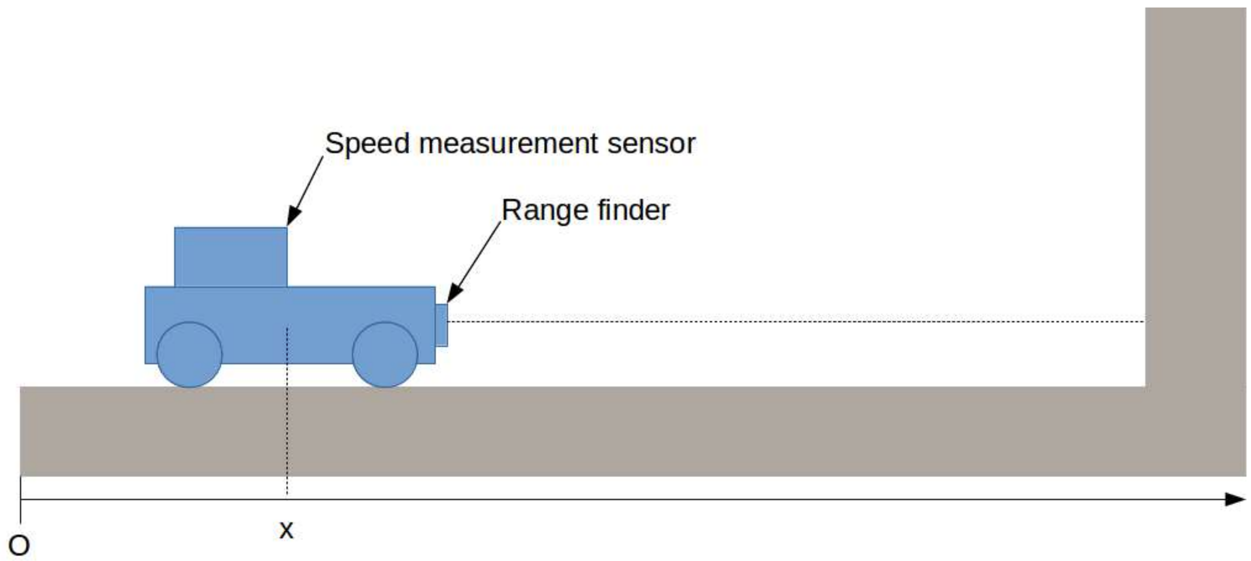
Time (s)	1	2	3	4
Value (m)	1.2	2	3.3	4.1

- Estimate the position of the robot using the Kalman filter given that its initial position is $x_0 = 0$ with the variance $P_0 = 0.1$.

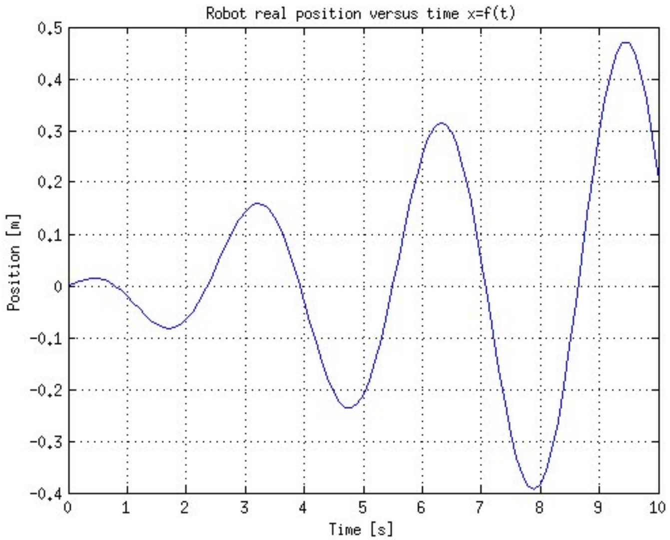


Kalman filter example 2

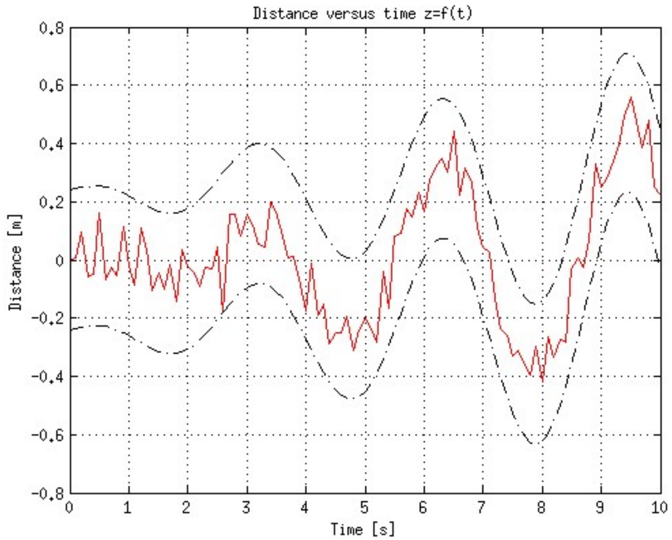
- Again, let's consider a robot that move in a single direction in front of a wall.
- Assume that the robot is equipped with two sensors : a speed measurement sensor and a distance measurement sensor (range finder). We'll consider in the following that both sensors are noisy.

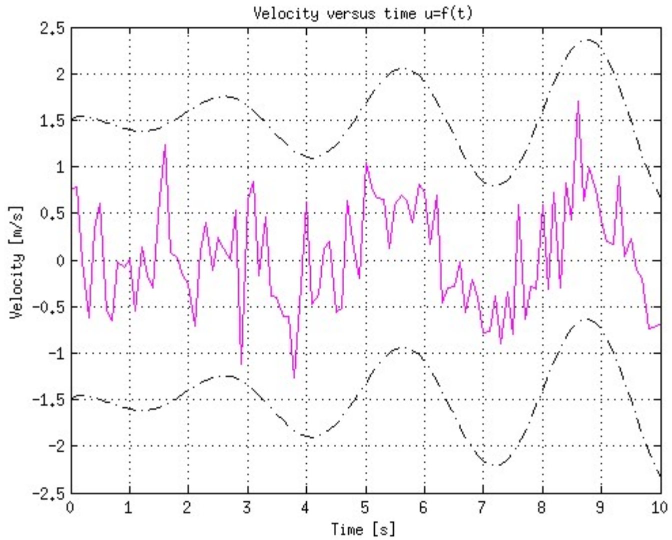


Our goal is to estimate, as accurately as possible, the position x of the robot:

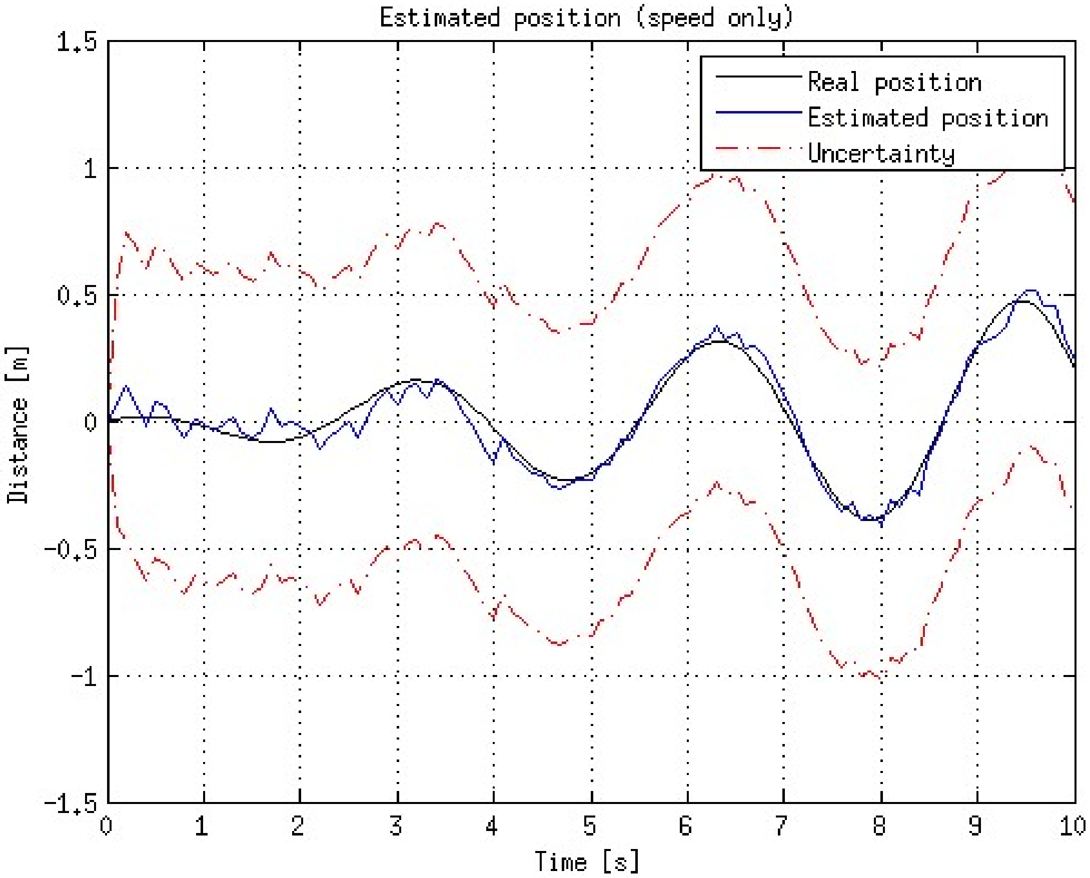


Input of the system are a noisy distance measurement and a noisy velocity measurement:





Result


Estimated position (speed only)
Real position
Estimated position
Uncertainty
Distance [m]
Time [s]
1.5
1
0.5
0
-0.5
-1
-1.5
0
1
2
3
4
5
6
7
8
9
10

Extended Kalman filter

- So long as the errors are *roughly* Gaussian, the Kalman filter can be used, although it may not be provably optimal.
- To cope with nonlinearity, the *Extended Kalman Filter (EKF)* is used. It linearises the plant and the measurement by deleting high-order terms from the Taylor expansion.

$$\begin{aligned} f(x) &= f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 \\ &\quad + \frac{f'''(x_0)}{3!}(x - x_0)^3 + \frac{f''''(x_0)}{4!}(x - x_0)^4 + \cdots \end{aligned}$$

The nonlinear system model can be expressed as:

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + W w_{k-1}$$

$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + V v_k .$$

- $\tilde{x}_k$ and $\tilde{z}_k$ are the approximate state and measurement vectors

$$\tilde{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0)$$
$$\tilde{z}_k = h(\tilde{x}_k, 0)$$

- A is the Jacobian matrix of partial derivatives of f with respect to x,

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}_{k-1}, u_{k-1}, 0),$$

- W is the Jacobian matrix of partial derivatives of f with respect to w

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{x}_{k-1}, u_{k-1}, 0),$$

- H is the Jacobian matrix of partial derivatives of h with respect to x

$$H_{[i,\,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\tilde{x}_k, 0)$$

- V is the Jacobian matrix of partial derivatives of h with respect to v

$$V_{[i,\,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}}(\tilde{x}_k, 0)$$

UET-VNU

Extended Kalman Filter Summary

Time Update ("Predict")

(1) Project the state ahead

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0)$$

(2) Project the error covariance ahead

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$$

Measurement Update ("Correct")

(1) Compute the Kalman gain

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$$

(2) Update estimate with measurement z_k

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0))$$

(3) Update the error covariance

$$P_k = (I - K_k H_k) P_k^-$$

Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

- **Example**:

A differential driving robot with inputs of v_r and v_l for right and left wheel velocity. If the driving signal noise is assumed to be of zero mean and a variance of σ^2 for both velocities. It is also assumed that a range sensor is used to measure the distance between the robot location (x_r, y_r) and three landmarks (x_1, y_1), (x_2, y_2), and (x_3, y_3). The measurement device has a gaussian noise of zero mean and variance of σ_m^2. Using Kalman filter to estimate the location of the robot. Assuming that the initial pose of the robot is [0 0 0] and a state covariance matrix $P=\mathbf{0}_{3x3}$.

- **System Model**:

$$\begin{cases} \dot{x} = \dfrac{V_r + V_l}{2}\cos(\theta) \\ \dot{y} = \dfrac{V_r + V_l}{2}\sin(\theta) \\ \dot{\theta} = \dfrac{V_r - V_l}{d} \end{cases} \Rightarrow \begin{cases} x(k+1) = x(k) + \dfrac{V_r + V_l}{2}\cos(\theta)\Delta T \\ y(k+1) = y(k) + \dfrac{V_r + V_l}{2}\sin(\theta)\Delta T \\ \theta(k+1) = \theta(k) + \dfrac{V_r - V_l}{d}\Delta T \end{cases}$$

- **Measurement Model**:

$$\begin{cases} r_1 = \sqrt{(x - x_1)^2 + (y - y_1)^2} \\ r_2 = \sqrt{(x - x_2)^2 + (y - y_2)^2} \\ r_3 = \sqrt{(x - x_3)^2 + (y - y_3)^2} \end{cases}$$

- **Time update (Prediction stage):**

$$A = \frac{\partial f}{\partial x}(\hat{x}_{k-1}, u_{k-1}, 0) = \begin{bmatrix} \frac{\partial f_x}{\partial x} & \frac{\partial f_x}{\partial y} & \frac{\partial f_x}{\partial \theta} \\ \frac{\partial f_y}{\partial x} & \frac{\partial f_y}{\partial y} & \frac{\partial f_y}{\partial \theta} \\ \frac{\partial f_\theta}{\partial x} & \frac{\partial f_\theta}{\partial y} & \frac{\partial f_\theta}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{V_r + V_l}{2}\sin(\theta)\Delta T \\ 0 & 1 & \frac{V_r + V_l}{2}\cos(\theta)\Delta T \\ 0 & 0 & 1 \end{bmatrix}$$

- Input noise is caused by input velocities:

$$W = \frac{\partial f}{\partial w}(\hat{x}_{k-1}, u_{k-1}, 0) = \frac{\partial f}{\partial u}(\hat{x}_{k-1}, u_{k-1}, 0) = \begin{bmatrix} \frac{\partial f_x}{\partial V_r} & \frac{\partial f_x}{\partial V_l} \\ \frac{\partial f_y}{\partial V_r} & \frac{\partial f_y}{\partial V_l} \\ \frac{\partial f_\theta}{\partial V_r} & \frac{\partial f_\theta}{\partial V_l} \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\cos(\theta)\Delta T & \frac{1}{2}\cos(\theta)\Delta T \\ \frac{1}{2}\sin(\theta)\Delta T & \frac{1}{2}\sin(\theta)\Delta T \\ \frac{1}{d}\Delta T & -\frac{1}{d}\Delta T \end{bmatrix}$$

$$Q = \begin{bmatrix} \sigma_{v_r}^2 & 0 \\ 0 & \sigma_{v_l}^2 \end{bmatrix}$$

- **Measurement update (Correction stage):**

$$\begin{cases} r_1 = \sqrt{(x-x_1)^2+(y-y_1)^2} \\ r_2 = \sqrt{(x-x_2)^2+(y-y_2)^2} \\ r_3 = \sqrt{(x-x_3)^2+(y-y_3)^2} \end{cases}$$

$$R(k) = \begin{bmatrix} \sigma_m^2 & 0 & 0 \\ 0 & \sigma_m^2 & 0 \\ 0 & 0 & \sigma_m^2 \end{bmatrix}$$

$$H_{[i,\,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\tilde{x}_k, 0) = \begin{bmatrix} \frac{\partial r_1}{\partial x} & \frac{\partial r_1}{\partial y} & \frac{\partial r_1}{\partial \theta} \\ \frac{\partial r_2}{\partial x} & \frac{\partial r_2}{\partial y} & \frac{\partial r_2}{\partial \theta} \\ \frac{\partial r_3}{\partial x} & \frac{\partial r_3}{\partial y} & \frac{\partial r_3}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \frac{x-x_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} & \frac{y-y_1}{\sqrt{(x-x_1)^2+(y-y_1)^2}} & 0 \\ \frac{x-x_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} & \frac{y-y_2}{\sqrt{(x-x_2)^2+(y-y_2)^2}} & 0 \\ \frac{x-x_3}{\sqrt{(x-x_3)^2+(y-y_3)^2}} & \frac{y-y_3}{\sqrt{(x-x_3)^2+(y-y_3)^2}} & 0 \end{bmatrix}$$