Extended & Unscented Kalman Filter

CMSC 35410

Truong Son Hy [*]

[*] Department of Computer Science
The University of Chicago

Ryerson Physical Lab

Gaussian Filters

- an important family of recursive state estimators.
- constitutes the earliest tractable implementations of the Bayes filter for continuous spaces.
- the most popular family of techniques to date despite a number of shortcomings.

Gaussian Filters

Gaussian Filters

- an important family of recursive state estimators.
- constitutes the earliest tractable implementations of the Bayes filter for continuous spaces.
- the most popular family of techniques to date despite a number of shortcomings.

Basic idea

Beliefs are represented by multivariate normal distribution:

$$p(x) = \det(2\pi\Sigma)^{-\frac{1}{2}} \exp\left\{ -\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}$$

Representations

Moments representation

The representation of a Gaussian by its mean and covariance is called **moments representation** because the mean and covariance are the first and second moments of a probability distribution; all other moments are zero for normal distributions.

Representations

Moments representation

The representation of a Gaussian by its mean and covariance is called **moments representation** because the mean and covariance are the first and second moments of a probability distribution; all other moments are zero for normal distributions.

Alternative

- Canonical/natural representation.
- Both moments and canonical/natural representations are functionally equivalent in that a bijective mapping exits that transforms one into the other (and back).

Kalman Filter

History

Kalman filter (KF) was invented in the 1950s by Rudolph Emil Kalman, as a technique for filtering and prediction in linear systems.

Kalman Filter

History

Kalman filter (KF) was invented in the 1950s by Rudolph Emil Kalman, as a technique for filtering and prediction in linear systems.

Overview

- Kalman filter implements belief computation for continuous states. It is not applicable to discrete or hybrid state spaces.
- Kalman filter represents beliefs by the **moments representation**. At time t, the belief is represented by the mean μ_t and the covariance Σ_t. Posteriors are Gaussian. KF uses Markov assumptions of the Bayes filter.

Linear Gaussian Systems (1)

Next state probability $p(x_t | u_t, x_{t-1})$ must be a linear function in its arguments with added Gaussian noise:

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

where:

- $x_t \in \mathbb{R}^n$ is the **state vector**.
- $u_t \in \mathbb{R}^m$ is the **control vector**.
- $\epsilon_t \sim \mathcal{N}(0, R_t)$ is the **system noise** capturing the randomness of the system with mean zero and covariance R_t.
- $A_t \in \mathbb{R}^{n \times n}$ and $B_t \in \mathbb{R}^{n \times m}$.

This is **linear system dynamics**.

Linear Gaussian Systems (1)

Next state probability $p(x_t|u_t, x_{t-1})$ must be a linear function in its arguments with added Gaussian noise:

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

The mean of the posterior $p(x_t|u_t, x_{t-1})$ is given by $A_t x_{t-1} + B_t u_t$ and covariance R_t:

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

Linear Gaussian Systems (2)

The measurement probability $p(z_t|x_t)$ must also be linear in its arguments, with added Gaussian noise:

$$z_t = C_t x_t + \delta_t$$

where:

- $z_t \in \mathbb{R}^k$ is the **measurement vector**.
- $\delta_t \sim \mathcal{N}(0, Q_t)$ is the **measurement noise** with zero mean and covariance Q_t.
- $C_t \in \mathbb{R}^{k \times n}$.

We have:

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

Kalman Filter Algorithm (1)

The initial belief $bel(x_0) \sim \mathcal{N}(\mu_0, \Sigma_0)$ must be normal distributed:

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

Algorithm:

1. **Input**: μ_{t-1}, Σ_{t-1}, u_t, z_t
2. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
3. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$
4. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1} \leftarrow$ Kalman gain
5. $\mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t)$
6. $\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$
7. **Output**: μ_t, Σ_t

Kalman Filter Algorithm (2)

Line 1 & 2:

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

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

Prediction step:

$$\bar{bel}(x_t) = \int p(x_t | x_{t-1}, u_t) \;\; bel(x_{t-1}) \;\; dx_{t-1}$$

where:

$$p(x_t | x_{t-1}, u_t) \sim \mathcal{N}(x_t; A_t x_{t-1} + B_t u_t, R_t)$$

$$bel(x_{t-1}) \sim \mathcal{N}(x_{t-1}; \mu_{t-1}, \Sigma_{t-1})$$

Kalman Filter Algorithm (3)

Line 4, 5 & 6:

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

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

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

Measurement update step:

$$bel(x_t) = \eta \;\; p(z_t|x_t) \;\; \bar{bel}(x_t)$$

where:

$$p(z_t|x_t) \sim \mathcal{N}(z_t; C_t x_t, Q_t)$$

$$\bar{bel}(x_t) \sim \mathcal{N}(x_t; \bar{\mu}_t, \bar{\Sigma}_t)$$

Extended Kalman Filter (1)

Problem with Kalman Filter

Kalman Filter has the assumptions of **linear** state transitions and **linear** measurements with added Gaussian noise.

Extended Kalman Filter (1)

Problem with Kalman Filter

Kalman Filter has the assumptions of **linear** state transitions and **linear** measurements with added Gaussian noise.

Extended Kalman Filter

The extended Kalman filter (EKF) overcomes the linearity assumptions. Assume that the next state probability and the measurement probabilities are governed by nonlinear functions g and h:

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

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

The function g replaces the matrices A_t and B_t, and h replaces the matrix C_t.

Extended Kalman Filter (2)

Problem with Extended Kalman Filter

- With the arbitrary functions g and h, the belief is no longer a Gaussian.
- Performing the belief update exactly is usually impossible for nonlinear functions g and h, in the sense that the Bayes filter does not possess a closed-form solution.

Extended Kalman Filter (2)

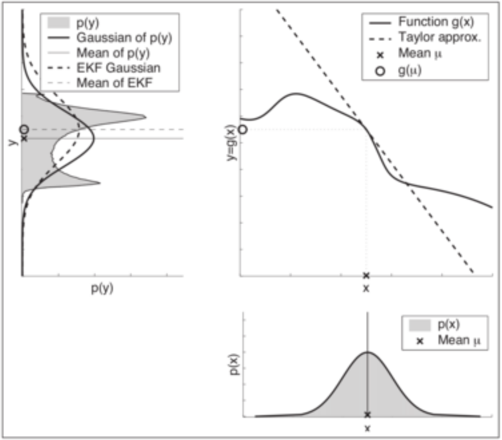
Problem with Extended Kalman Filter

- With the arbitrary functions g and h, the belief is no longer a Gaussian.
- Performing the belief update exactly is usually impossible for nonlinear functions g and h, in the sense that the Bayes filter does not possess a closed-form solution.

EKF's approximation

EKF calculates an approximation to the true belief. It represents the approximation by a Gaussian (with moments representation). The belief $bel(x_t)$ at time t is represented by a mean μ_t and a covariance Σ_t. EKF inherits from the original KF the basic belief representation, but it differs in that this belief is only approximate, not exact.

Extended Kalman Filter (3)



Thrun et. al, 2006

Linearization via Taylor Expansion (1)

Suppose we are given a nonlinear next state function g. Linearization approximates g by a linear function that is tangent to g at the mean of the Gaussian. Once g is linearized, the mechanics of belief propagation are equivalent to those of the Kalman filter. The same argument applies to h.

Linearization via Taylor Expansion (1)

Suppose we are given a nonlinear next state function g. Linearization approximates g by a linear function that is tangent to g at the mean of the Gaussian. Once g is linearized, the mechanics of belief propagation are equivalent to those of the Kalman filter. The same argument applies to h.

First-order approximation

First-order Taylor expansion constructs a linear approximation to a function g from g's value and slope. The slope is given by:

$$G_t = g'(u_t, \mu_{t-1}) = \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}}\bigg|_{x_{t-1}=\mu_{t-1}}$$

G_t is the gradient of g at u_t and μ_{t-1}.

Linearization via Taylor Expansion (2)

g is approximated by its value at u_t and μ_{t-1}:

$$\begin{aligned} g(u_t, x_{t-1}) &\approx g(u_t, \mu_{t-1}) + g'(u_t, \mu_{t-1})(x_{t-1} - \mu_{t-1}) \\ &= g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1}) \end{aligned}$$

The next state probability is approximated as:

$$\begin{aligned} p(x_t | u_t, x_{t-1}) \approx \det(2\pi R_t)^{-1/2} \exp \Big\{ \\ &- \frac{1}{2}[x_t - g(u_t, \mu_{t-1}) - G_t(x_{t-1} - \mu_{t-1})]^T \\ &R_t^{-1} \\ &[x_t - g(u_t, \mu_{t-1}) - G_t(x_{t-1} - \mu_{t-1})] \Big\} \end{aligned}$$

Linearization via Taylor Expansion (3)

The same linearization applies to the measurement function h:

$$\begin{aligned} h(x_t) &\approx h(\bar{\mu}_t) + h'(\bar{\mu}_t)(x_t - \bar{\mu}_t) \\ &= h(\bar{\mu}_t) + H_t(x_t - \bar{\mu}_t) \end{aligned}$$

where

$$H_t = h'(x_t)|_{x_t=\bar{\mu}_t} = \left.\frac{\partial h(x_t)}{\partial x_t}\right|_{x_t=\bar{\mu}_t}$$

We have the approximation for measurement probability:

$$p(z_t|x_t) \approx \det(2\pi Q_t)^{-1/2}$$

$$\exp\left\{-\frac{1}{2}[z_t - h(\bar{\mu}_t) - H_t(x_t - \bar{\mu}_t)]^T Q_t^{-1}[z_t - h(\bar{\mu}_t) - H_t(x_t - \bar{\mu}_t)]\right\}$$

The EKF algorithm (1)

Algorithm:

1. **Input**: μ_{t-1}, Σ_{t-1}, u_t, z_t
2. $\bar{\mu}_t = g(u_t, \mu_{t-1})$
3. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$
4. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1} \leftarrow$ Kalman gain
5. $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$
6. $\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$
7. **Output**: μ_t, Σ_t

The EKF algorithm (2)

	Kalman Filter	EKF
State prediction (Line 2)	$A_t \mu_{t-1} + B_t u_t$	$g(u_t, \mu_{t-1})$
Measurement prediction (Line 5)	$C_t \bar{\mu}_t$	$h(\bar{\mu}_t)$

- The Jacobian G_t corresponds to the matrices A_t and B_t.
- The Jacobian H_t corresponds to the matrix C_t.

The EKF algorithm (3)

Line 2 & 3:

$$\bar{\mu}_t = g(u_t, \mu_{t-1})$$
$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$$

Prediction step:

$$\bar{bel}(x_t) = \int p(x_t | x_{t-1}, u_t) \ \ bel(x_{t-1}) \ \ dx_{t-1}$$

where:

$$p(x_t | x_{t-1}, u_t) \sim \mathcal{N}(x_t; g(u_t, \mu_{t-1}) + G_t(x_{t-1} - \mu_{t-1}), R_t)$$

and

$$bel(x_{t-1}) \sim \mathcal{N}(x_{t-1}; \mu_{t-1}, \Sigma_{t-1})$$

The EKF algorithm (4)

Line 4, 5 & 6:

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

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

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

Measurement update step:

$$bel(x_t) = \eta \;\; p(z_t|x_t) \;\; \bar{bel}(x_t)$$

where:

$$p(z_t|x_t) \sim \mathcal{N}(z_t; h(\bar{\mu}_t) + H_t(x_t - \bar{\mu}_t), Q_t)$$

and

$$\bar{bel}(x_t) \sim \mathcal{N}(x_t; \bar{\mu}_t, \bar{\Sigma}_t)$$

Demo EKF 1D (a)

Consider the dynamnics of one dimensional timeseries:

$$x_t = g(x_{t-1}) + \epsilon_t = \sin(x_{t-1} + \Delta x) + \mathcal{N}(0, 0.01)$$

$$z_t = h(x_t) + \delta_t = x_t + \mathcal{N}(0, 0.01)$$

In this example, we ignore the control signal u_t. All our matrices have size $1x1$:

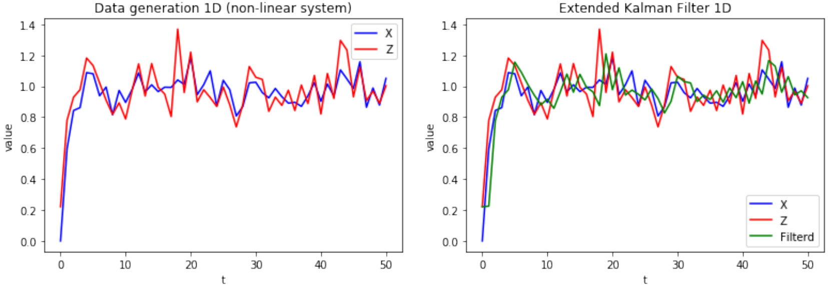
$$G_t|_{x=\mu} = [\cos(\mu)]$$

$$H_t = I_1 = [1]$$

$$R = [0.01]$$

$$Q = [0.01]$$

Demo EKF 1D (b)



Total norm ℓ_2 error ≈ 5.29

Practical considerations

Each update of EKF requires time complexity $O(k^{2.8} + n^2)$, where k is the dimension of the measurement vector z_t, and n is the dimension of the state vector x_t.

Practical considerations

Each update of EKF requires time complexity $O(k^{2.8} + n^2)$, where k is the dimension of the measurement vector z_t, and n is the dimension of the state vector x_t.

Multiple hypotheses

EKF represents the belief by a multivariate Gaussian distribution. How about **multiple distinct hypotheses**?

Practical considerations

Each update of EKF requires time complexity $O(k^{2.8} + n^2)$, where k is the dimension of the measurement vector z_t, and n is the dimension of the state vector x_t.

Multiple hypotheses

EKF represents the belief by a multivariate Gaussian distribution. How about **multiple distinct hypotheses**?

Mixture of Gaussians

A common extension of EKFs is to represent the posteriors using a mixture of J Gaussians (multi-modal representations):

$$bel(x) = \sum_j a_j \det(2\pi\Sigma_{j,t})^{-1/2} \exp\left\{ -\frac{1}{2}(x_t - \mu_{j,t})^T \Sigma_{j,t}^{-1} (x_t - \mu_{j,t}) \right\}$$

where a_j are mixture parameters with $a_j \geq 0$ and $\sum_j a_j = 1$. This is called **multi-hypothesis extended Kalman filter** or MHEKF.

Unscented Kalman Filter (1)

Hardness of EKF

- EKF approximates functions $g(u_t, x_{t-1})$ and $h(x_t)$.
- Intuition: It is easier to approximate a Gaussian than to approximate a function.

Unscented Kalman Filter (1)

Hardness of EKF

- EKF approximates functions $g(u_t, x_{t-1})$ and $h(x_t)$.
- Intuition: It is easier to approximate a Gaussian than to approximate a function.

Unscented transform

Instead of performing a linear approximation to the function and passing a Gaussian through it, we pass a deterministically chosen set of points, known as **sigma points** through the function, and then fit a Gaussian to the resulting transformed points. This is known as the **unscented transform**.

The unscented transform (1)

Assume $p(x) = \mathcal{N}(x|\mu, \Sigma)$, and consider estimating $p(y)$, where $y = f(x)$ for some nonlinear function f. We create a set of $2d + 1$ sigma points x_i given by:

$$x = \left\{\mu, \quad \left\{\mu + \left[(d+\lambda)\Sigma\right]_{:i}^{1/2}\right\}_{i=1}^{d}, \quad \left\{\mu - \left[(d+\lambda)\Sigma\right]_{:i}^{1/2}\right\}_{i=1}^{d}\right\}$$

where

$$\lambda = \alpha^2(d + \kappa) - d$$

is a scaling parameter to be specified, and the notation $M_{:i}$ denotes the i'th column of matrix M. The optimal values of α, β and κ are **problem dependent**. In the one-dimensional case $d = 1$, we have $\alpha = 1$, $\beta = 0$ and $\kappa = 2$. Thus, the three sigma points are:

$$x = \left\{\mu, \quad \mu + \sqrt{3}\sigma, \quad \mu - \sqrt{3}\sigma\right\}$$

The unscented transform (2)

The sigma points are propagated through the nonlinear function to yield $y_i = f(x_i)$ and the mean and covariance for y is computed as follows:

$$\mu_y = \sum_{i=0}^{2d} w_m^i y_i$$

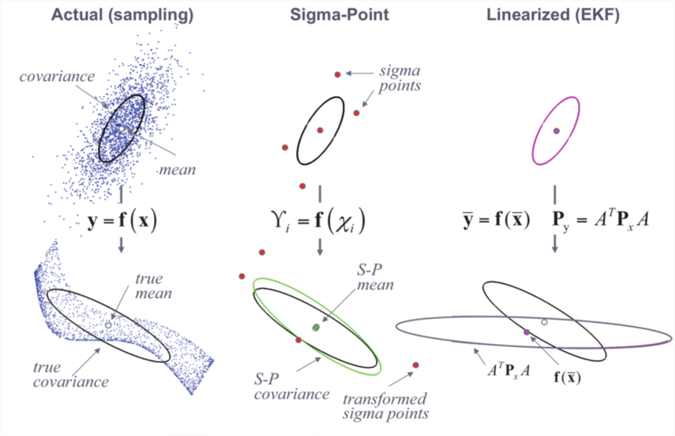
$$\Sigma_y = \sum_{i=0}^{2d} w_c^i (y_i - \mu_y)(y_i - \mu_y)^T$$

where:

$$w_m^0 = \frac{\lambda}{d + \lambda}$$

$$w_c^0 = \frac{\lambda}{d + \lambda} + (1 - \alpha^2 + \beta)$$

$$w_m^i = w_c^i = \frac{1}{2(d + \lambda)}$$



Wan and der Merwe 2001

The UKF Algorithm (1)

Approximate the predictive density $p(x_t|z_{1:t-1}, u_{1:t}) \approx \mathcal{N}(z_t|\bar{\mu}_t, \bar{\Sigma}_t)$ by passing the old belief state $\mathcal{N}(x_{t-1}|\mu_{t-1}, \Sigma_{t-1})$ through non-linearity g:

$$x_{t-1}^0 = \left\{ \mu_{t-1},\ \left\{ \mu_{t-1} + \left[(d+\lambda)\Sigma_{t-1} \right]_{:i}^{1/2} \right\}_{i=1}^d,\ \left\{ \mu_{t-1} - \left[(d+\lambda)\Sigma_{t-1} \right]_{:i}^{1/2} \right\}_{i=1}^d \right\}$$

$$\bar{x}_t^{*i} = g(u_t, x_{t-1}^{0i})$$

$$\bar{\mu}_t = \sum_{i=0}^{2d} w_m^i \bar{x}_t^{*i}$$

$$\bar{\Sigma}_t = \sum_{i=0}^{2d} w_c^i (\bar{x}_t^{*i} - \bar{\mu}_t)(\bar{x}_t^{*i} - \bar{\mu}_t)^T + R_t$$

The UKF Algorithm (2)

Approximate the likelihood $p(z_t|x_t) \approx \mathcal{N}(z_t|\hat{z}_t, S_t)$ by passing the prior $\mathcal{N}(z_t|\bar{\mu}_t, \bar{\Sigma}_t)$ through the non-linearity h:

$$\bar{x}_t^0 = \left\{ \bar{\mu}_t,\ \left\{ \bar{\mu}_t + \left[(d+\lambda)\bar{\Sigma}_t \right]_{:i}^{1/2} \right\}_{i=1}^d,\ \left\{ \bar{\mu}_t - \left[(d+\lambda)\bar{\Sigma}_t \right]_{:i}^{1/2} \right\}_{i=1}^d \right\}$$

$$\bar{z}_t^{*i} = h(\bar{x}_t^{0i})$$

$$\hat{z}_t = \sum_{i=0}^{2d} w_m^i \bar{z}_t^{*i}$$

$$S_t = \sum_{i=0}^{2d} w_c^i (\bar{z}_t^{*i} - \hat{z}_t)(\bar{z}_t^{*i} - \hat{z}_t)^T + Q_t$$

The UKF Algorithm (3)

Finally, get the posterior $p(x_t|z_{1:t}, u_{1:t}) \approx \mathcal{N}(x_t|\mu_t, \Sigma_t)$:

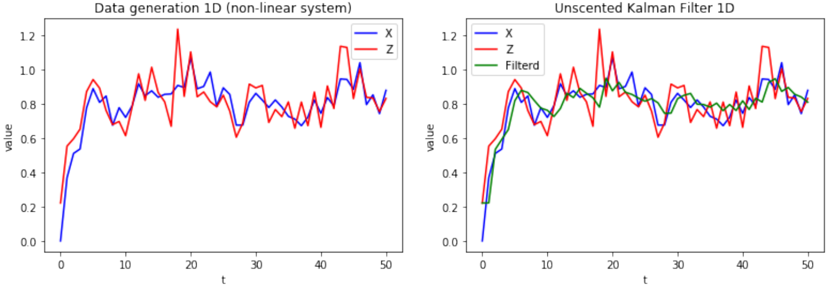
$$\bar{\Sigma}_t^{x,z} = \sum_{i=0}^{2d} w_c^i (\bar{x}_t^{*i} - \bar{\mu}_t)(\bar{z}_t^{*i} - \hat{z}_t)^T$$

$$K_t = \bar{\Sigma}_t^{x,z} S_t^{-1}$$

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

$$\Sigma_t = \bar{\Sigma}_t - K_t S_t K_t^T$$

Demo UKF 1D



Total norm ℓ_2 error of UKF $\approx 3.64 < 5.29$ of EKF.

Demo KF vs. UKF (a)

Consider the **velocity model** for tracking an object in two-dimensional. Suppose the state vector to be:

$$x_t^T = (x_{1t}, x_{2t}, \dot{x_{1t}}, \dot{x_{2t}})$$

where x_{1t} and x_{2t} is the x and y coordinates at time t; $\dot{x_{1t}}$ and $\dot{x_{2t}}$ are the corresponding velocity in x and y axis, respectively. Assume we have a linear dynamic system:

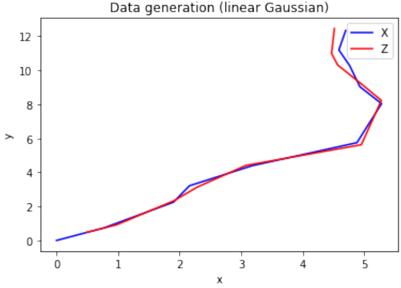
$$x_t = A_t x_{t-1} + \epsilon_t$$

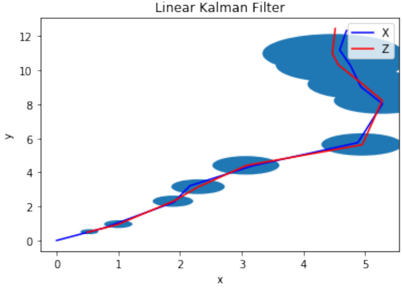
$$\begin{bmatrix} x_{1t} \\ x_{2t} \\ \dot{x_{1t}} \\ \dot{x_{2t}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1(t-1)} \\ x_{2(t-1)} \\ \dot{x_{1(t-1)}} \\ \dot{x_{2(t-1)}} \end{bmatrix}$$

where Δ is the sampling period. Assume we have a linear measurement system:

$$z_t = x_t + \delta_t$$

Demo KF vs. UKF (b)





This case, linearity works fine. Let's make it more complicated!

Demo KF vs. UKF (c)

Consider the robot uses the **polar coordinate** (r, θ) internally, while the camera on top of it still uses the x, y coordinate (pixel!). The robot has a constant **radius velocity** and **angular velocity**:

$$r_t \leftarrow r_{t-1} + \Delta r$$

$$\theta_t \leftarrow \theta_{t-1} + \Delta\theta$$

Conversion:

$$r_t = \sqrt{x_{1t}^2 + x_{2t}^2}$$

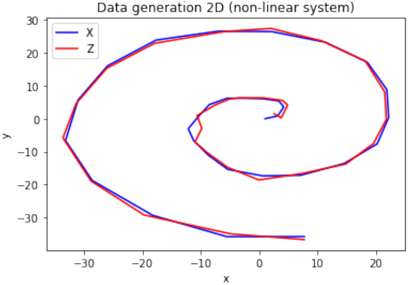
$$\theta_t = \tan^{-1}\left(\frac{x_{2t}}{x_{1t}}\right)$$

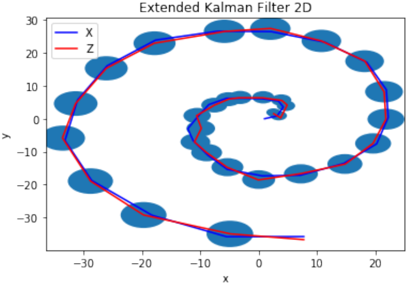
and

$$x_{1t} = r_t \cos(\theta_t)$$

$$x_{2t} = r_t \sin(\theta_t)$$

Demo KF vs. UKF (d)





We get an infinite curve!

Demo multi-object tracking with KF (a)

Consider N objects moving in a 2D plane. Let $x_t^{(i)}$ and $y_t^{(i)}$ be the horizontal and vertical coordinates, and $\Delta x_t^{(i)}$ and $\Delta y_t^{(i)}$ be the velocity:

$$\boldsymbol{x}_t^T = \begin{bmatrix} x_t^{(1)} & y_t^{(1)} & \cdots & x_t^{(N)} & y_t^{(N)} & \Delta x_t^{(1)} & \Delta y_t^{(1)} & \cdots & \Delta x_t^{(N)} & \Delta y_t^{(N)} \end{bmatrix}$$

$$\boldsymbol{x}_t = A_t \boldsymbol{x}_{t-1} + \boldsymbol{\epsilon}_t$$

$$\begin{bmatrix} x_t^{(1)} \\ y_t^{(1)} \\ \vdots \\ \Delta x_t^{(1)} \\ \Delta y_t^{(1)} \\ \vdots \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x_{t-1}^{(1)} \\ y_{t-1}^{(1)} \\ \vdots \\ \Delta x_{t-1}^{(1)} \\ \Delta y_{t-1}^{(1)} \\ \vdots \end{bmatrix} + \boldsymbol{\epsilon}_t$$

where I is the identity matrix of size $2N \times 2N$ and

$$A_{11} = I \qquad A_{12} = \Delta \cdot I \qquad A_{21} = I \qquad A_{22} = I$$

Demo multi-object tracking with KF (b)

Measurement:

$$\boldsymbol{z}_t = C_t \boldsymbol{x}_t + \boldsymbol{\delta}_t$$

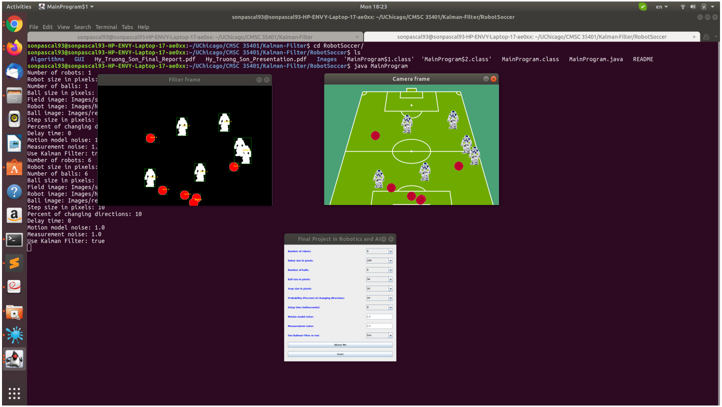
$$\begin{bmatrix} \hat{x}_t^{(1)} \\ \hat{y}_t^{(1)} \\ \vdots \end{bmatrix} = \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} x_t^{(1)} \\ y_t^{(1)} \\ \vdots \\ \Delta x_t^{(1)} \\ \Delta y_t^{(1)} \\ \vdots \end{bmatrix} + \boldsymbol{\delta}_t$$

where:

$$C_1 = I \qquad C_2 = \mathbf{0}^{2N \times 2N}$$

Demo multi-object tracking with KF (c)

We need the **Hungarian matching algorithm on bipartite graph** to match which measurement goes to which object!



https://github.com/HyTruongSon/RobotSoccer

Q & A

Thank you very much for your attention!