Prediction is hat → $\hat{x}$?

Implementation of Kalman' filter with python language

→ Using Numpy package

→ Carried out in two function

(I) Prediction

(II) Update

→ stochastic estimation (?) from noisy sensor measure_ments

→ Is space and water same kind of space. Like

→ Recursive solution

→ Discrete-data linear problem. (?)

→ a set of mathematical equation that implement predictor-corrector type estimator

→ Optimal in a sence the minimize the estimated error covariance.

→ The Kalman filter has been the subject... autonomous vehicle

→ Mathematical formulation:

linear stochastic difference equation:

$$X_K = Ax_{K-1} + B.u_K + w_{K-1}$$

$$y_K = Hx_K + v_K$$

$x \in K^n$; $y \in K^n$ state

w_K, v_K Random variable → Process & measurement noise

w_k and v_k to be independent of each other with normal probability distribution

$$P(w) \approx N(0, Q)$$
$$P(v) \approx N(0; R)$$

process Noise Covariance $\rightarrow Q$

Measurement NOISE $\rightarrow R$

$A \rightarrow n \times n$ Matrix

↳ Relates to previous time step.

$B \rightarrow n \times 1$ matrix

↳ Relates to optional control input $u \in K^1$ to the state x

$H \rightarrow m \times n$ matrix

↳ relates to the state to the measurement y_K.

So,

Prediction: ②

$$\hat{X}_K = A_{K-1} X_{K-1} + B_K U_K$$

$$\hat{P}_K = A_{K-1} P_{K-1} A^T_{K-1} + Q_{K-1}$$

UPDATE

$$V_K = Y_K - H_K \hat{X}_K$$

$$S_K = H_K \hat{P}_K H^T_K + R_K$$

$$K_K = \hat{P}_K H_K^T S_K^{-1}$$

$$X_K = \hat{X}_K + K_K V_K$$

$$P_K = \hat{P}_K - K_K S_K K^T_K$$

→ hell of a equation ?

$\hat{X}_K$ → Predicted mean

$\hat{P}_K$ → Predicted covariance

↓

On the time step K
before seeing the measurement

X_K → Estimated mean

P_K → Estimated covariance

↓

On the time step K
after seeing the measurement

Y_K → Mean of the measurement on the time step K

V_K → Innovation/measurement residual on time step K

S_K → Measurement prediction covariace

K_K → The filter gain
how much prediction should be corrected.

→ It can provide accurate, continuously updated information about the position and velocity of an object.

→ Python Code for Kalman filter: