Table of Contents

# Kalman Filter

## 1. Introduction

### Linear Kalman filter (KF)

#### Models in Kalman filter

* + In kalman filter, there are 3 kinds of models: noise model, process model and measurement model .
    - Process model:
    - process model
    - process model
    - where  
      *state vector x*:  
      input vector x. x represents the states of the system you design (set of states)  
      *state variable :*  
      *each variable inside x. Each state variable represents a state in the system(velocity, position)*  
      *trainsition matrix phi:*  
      *Transition matrix is the matrix transiting the states at k moment to states at k+1 moment. The transition matrix is dependent on the system you build.*  
      *noise term Wk:*  
      *usually the noise is in Gaussain noise*  
      *For example:*   
      *we know velocity v\_k= a \* t\_k at k moment. In k+1 moment, v\_k+1 = a t\_k+1. Then we have v\_k+1 = v\_k + a* (t\_k+1 - t\_k). To represent v and a in matrix form:  
      [v\_k+1, a\_k+1]^T = [1, delta\_t]\* [v\_k, a\_k]^T  
      [1, delta\_t] is the transition matrix here
      * Measurement model:
      * process model  
        where  
        *zk*: measurement vector obtained from sensors  
        *H*: connection matrix between measurement vector z and the state vector x  
        *vk*: the noise from measurement
      * Noise model:  
        in noise model, there are 3 covariance matrices to representing the information in the model  
        noise model
      * noise model
      * noise model

#### Designing a kalman filter

* + - * The key idea of Kalman filter is to use maximium likehood method to minimize the error between the expected output signal and the measurement. In kalman filter, it defines *estimation* v1 and *prior estimation* v2  
        Their relation is:  
        relationship
      * where "Zk - HXk` " is called measurement residual. It is the error term between actual states and estimated states. In my prospective, v1 can be considered as the expected output states processed with noise term, which can have the ‘ability’ to ‘fight against’ the noise. Then it is used to predict the next output by transition matrix.  
        What’s more, in the equation above, ‘Kk’ is called Kalman gain, which is used to control how the error term affects the estimation. Our goal is to calculate the kalman gain to predict the next output.
        + **To calculate kalman gain**  
          To find the Kalman gain, let think of maximum likehood. In noise model, it’s actually calculating the covariance between measurement and expected value. When the probability is maximized, the error between measurement and the expected value is minimized and this is what we want. Therefore, we need to find a way to maximize the probability and return its arguments to update the Kalman gain.  
          Now calculate the trace of the matrix Pk to get T[Pk], which can convert the matrix to scalar value.  
          Insert the equation noise model to relationship . We get:  
          noise model  
          set the derivative of T[Pk] to zero (since at local maxima, derivative of T[Pk] =0) . Then we reach:  
          relationship  
          relationship  
          Finally, we get Kalman gain:
        + gain
        + gain
        + **Update estimate**  
          To update estimate, use the following equation (which was mentioned above):
        + gain
        + gain
        + **Update Covariance**  
          To update the covariance matrix Pk, we need to plug the kalman gain equation back to
        + gain  
          and then we get:
        + cov
        + cov
        + **Project to k+1 moment**  
          After update the covariance matrix at k moment, we can use the covariance matrix equation to update prior covariance at k+1 moment:
        + prj to k+1
        + prj to k+1
      * **Summary of equations**  
        gain

### Extended Kalman filter (EKF)

* + extended kalman filter is to model the non-linear system. The difference between EKF and linear KF shown as following:  
    EKF
  + where:  
    *F* is the transition matrix.  
    *Bu* is the control input  
    EKF uses derivative method to calcuate the transition matrix **F** and connection matrix **H** in order to update estimate **Pk** and Kalman gain. More details are introduced in [Kalman and Bayesian Filters in Python](https://drive.google.com/file/d/0By_SW19c1BfhSVFzNHc0SjduNzg/view)

## 2. Flowchart

flowchart

flowchart

## 3. My Summary

In my opinion, Kalman filter is to use error minimization and maximum likehood method to minimize the error between actual measurement and estimated states. As the likehood is maximized, the error will get close to zero. It also applies the noise term in the estimate to ‘fight against’ the noise in measurement.  
Generally, There are 2 steps in Kalman filter : predict step and update step. In predict step, it computes the Kalman gain, estimate and covariance at k moment. In update step, it updates the prior estimate and the prior covariance for the next moment.

## 4. Reference

This note is referred to this  [PDF](http://web.mit.edu/kirtley/kirtley/binlustuff/literature/control/Kalman%20filter.pdf) . More details can be found here.  
Further reading:  
[unscented kalman filter](https://www.seas.harvard.edu/courses/cs281/papers/unscented.pdf)  
[An introduction to the Kalman Filter](http://www.cs.unc.edu/~tracker/media/pdf/SIGGRAPH2001_CoursePack_08.pdf)  
[Kalman and Bayesian Filters in Python](https://drive.google.com/file/d/0By_SW19c1BfhSVFzNHc0SjduNzg/view)

## 5. Addition

Since this is my first time to write theoretical note on GitHub, if you have any suggestions, please feel free to commit here.