**KALMAN FILTERING**

There are instances where you have to take some action on a system based on its current state but there is a delay in getting the feedback from the sensor so you cannot take the appropriate action on the system thereby causing instability in the system. Filtering is necessary in many situations in engineering and embedded systems. For example, radio communication signal are heavily corrupted with noise or a GPS does not give an accurate location of the device, etc. Thus we need a good filtering algorithm which helps us remove the noise from signals as much as possible while retaining the useful information. In another instance, power supplies filter the line voltages in order to remove the unnecessary fluctuations that may result in shortening of the lifespan of devices like Computers, refrigirators. Kalman filters is an optimal estimation algorithm which helps us estimate the variables of a wide range of processes(in other terms, state of a system) when we’re unable to find them correctely(doubtful) due to limitations such as accuracy and other physical constraints. Kalman filters are mainly useful when – i) Variables of interest cannot be measured directly and are measurable only indirectly(for eg, velocity of a bird cannot be found by just observing it, so we try to find it indirectly using its previous velocity and position), ii) Available measurements of different variables from sensors,etc are subject to noise(leading to inaccuracy). Kalman filters are used to filter noisy signals, predict future states and also generate non-observable states.

As the values generated by sensors are noisy and cannot be used directly in state estimation, we have to filter the noisy signals. Kalman filters are reliable as they would not just help us filter the electromagnetic signals but also account for the uncertainity in the generated signal every step. Secondly, Kalman filters use of predicting future states is helpful when there are large time delays in the sensor feedback as this can cause instability in a system. Also, Kalman filters can be used to generate non-observable states like velocity of a moving object. At a time frame we have position of an object(a bird or a vehicle) but just differentiating positions in successive time frames to get the velocity might fetch us inaccurate results.

Kalman filters are reliable because they not only give good results in practicality but also easily and properly understandable theoretically. It can also be shown that of all possible filters, Kalman filters minimizes the variance of estimation error one of the most. They are often implemented in Embedded control systems because they can be used to generate an accurate estimation of a process variable which in turn can be used in controlling the main process. Another major use of this algorithm is that it does not require the history of all the previous states in finding the next state and only requires the immediate last state and a covariance matrix that defines the probability of the state being correct.

Kalman filters produce the optimal estiamate only for a linear system, i.e., in order to use Kalman filters to remove the noise, the process we are measuring must be describable by a linear system. In simpler terms, the system should be close to a linear response with respect to time in order to be applicable for Kalman filters. Examples of linear systems are Car driving, Planet revolution, radio signals etc.

There are two phases in Kalman filter, Prediction equations and Updation equations. Prediction equations give an estimate of the current state based on the previous state(only) and the input action, whereas in Update phase the difference between the current prediction and current observation is taken into consideration to refine the estimated state. The estimate given by prediction equations is called ‘a priori’ state estimate and update equation is called ‘a posteriori’ state estimate. These two phases keep running alternatively, with prediction followed by updation.

Prediction Equations -

  
 

Updation Equations -

  
   
 

X, x – Variables representing the state(the values we are trying to estimate/filter)

u - Action taken at a timestep(input)

P,p - Measure of estimated accuracy of state estimate(covariance matrix)

Q - Covariance of the process noise

R - Covariance of the observation noise

H - Observation(sensors) model

K - Matrix updated every step(called Kalman Gain)

z - Measurements given by the sensors

I - Identity matrix

The matrices A and B can be found if we get system data by doing some experiments before creating the filter. Firstly, if input is 0, newstate can be given by A\*laststate. Thus we can get the value of matrix A. Using this A, we can solve for the value of B by giving a known input.

Thus, once we have all the variables required, we start with current state x, current covariance matrix P, and current input u, we calculate the predicted state X and predicted covariance matrix p. Then we get the measurement z from the sensors, which we use in the update equations to modify the predictions accordingly and get updated state matrix and covariance. This continues based on our requirement of number of predictions(time of the experiment).

There are many applications of Kalman filtering, most of them being the systems where there is delay or inaccuracy in getting the observations. Most important ones are radar tracking of aircrafts, missiles etc, acoustic tracking of submarines, visual tracking of vehicles and people, etc. It can also be used to reconstruct particle trajectories from photographs and ocean currents from satellite surface measurements. It is considered analogous to Hidden Markov Models where we use discrete state variables, but here any system characterised by continuous state variables along with noise can applied on. The main limitation of Kalman filters is that it can be used only on linear systems. The extended Kalman Filters(EKF) tries to overcome this limitation by allowing the non linear systems too. There is also another variation called Unscented Kalman Filter.

The attached file is the implementation of Kalman filters on estimation of position of a vehicle moving randomly with changing its velocities continuously. All the inputs(both measurements and actions) were randomly generated numbers, which led to huge scattering of the output variables too. As explained above, it can be seen in the code that there are two phases ‘Prediction’ and ‘Updation’ in the function ‘update’. All other values, i.e. noise matrices, observation matrices, A and B were also chosen randomly. The library ‘Eigen’ was used to simply the implementation of matrix related calculations.