Application of Kalman Filter to a Potentiometer

nijatabbasov06

September 2024

Table of Contents

1 Abstract

Utilization of Kalman filter is ubiqitious in the fields of Control Engineering, Finance, Robotics, Navigation. Kalman Filter can be used for Sensor Fusion, Noise Reduction, Predictive Analysis etc. In this project, Kalman Filter is used as a Noise Reduction method, to get rid of the noise of the potentiometer which emerges because of the effect of the nearby devices. We will use Arduino Uno to detect the Potentiometer value and apply Kalman Filter to smooth out the signal.

2 Theoretical Background

The following is the State Space representation of our dynamic system:

$$x_{k+1} = \Phi x_k + w_k$$

Figure 1: Updating State x

$$z_k = H x_k + v_k$$

Figure 2: Measurement Equation

As our implementation is only related to 1D values, all of the matrices and the vectors are substituted by scalar values such that the state vector x_k and z_k is actual potentiometer value, the measured value, respectively, at time k. Also, w_k and v_k are the random variables sampled from Gaussian Distribution which represent the noise on the x and z, respectively. We also utilize P to represent covariance of error which is the difference between estimated x and real x. Moreover, R and Q are the Covariance matrices for v and w, respectively.

For our project, H and Φ, namely, the State Transition matrix and the Observation matrices are Identity matrices.

The following is the Kalman Filter algorithm[1].

Description	Equation
Kalman Gain	$K_k = P'_k H^T \left(H P'_k H^T + R\right)^{-1}$
Update Estimate	$\hat{x}_k = \hat{x}'_k + K_k \left(z_k - H\hat{x}'_k\right)$
Update Covariance	$P_k = \left(I - K_k H\right) P'_k$
Project into $k+1$	$\hat{x}'_{k+1} = \Phi \hat{x}_k$ $P_{k+1} = \Phi P_k \Phi^T + Q$

Figure 3: Kalman Filter Algorithm

Now let's dive deep into how to determine R, Q and initial values of P and x to set up the recursive algorithm.

2.1 Finding R

The error in the sensor value exists because of the nearby devices, loose connection between Arduino pins and the middle port of the potentiometer, etc. Calculating this error would be unnecessary and impossible theoratically. The solution for this problem is that we can observe the potVoltage val and approximate the variance of R solely by the change of this value. To elaborate, We can see that this voltage fluctuates by 0.03. For this reason, we will assume std to be 0.03. and calculate the variance to be 0.001.

2.2 Finding Q

In our project, the potentiometer value at time k and k+1 should be exactly same unless we are deliberately changing the potentiometer value.

If, for example, we change the potentiometer value at time k, then we can assume that the kalman filter starts operating at time k again with different initial x and initial P and the state space derivation of constant x value would hold. To exp,ain more, our new initial P and x would be the x and P at time k. As a result, we will consider Q to be 0 during the project.

2.3 Finding intial P and x

We are changing the potentiometer voltage in range 0 and 5, so choosing x to be 2.5 would be best decision for the robustness and the convergence of the Kalman value to the measured potVoltage.

The following is the Kalman Algorithm we are using after considering the parts discussed in theoretical background:

```
*/
KalmanResult KalmanFilter(float x_prev, float P_prev, float z){
  KalmanResult result;
  float K = P_prev / (P_prev + R);
  result.x = x_prev + K * (z - x_prev);
  result.P = (1- K) * P_prev;
  result.x_next = result.x;
  result.P_next = result.P + Q;
  return result;
}
```

Figure 4: Kalman Filter in our project

When it comes to the P, choosing P to be small would result in small kalman gain from the start. And as the Kalman filter only becomes smaller and smaller, because P would be decreasing after each loop, we can see that choosing P small as in order of R would result in a lot of time for the kalman value converge to the actual voltage. It is also possible that P become so small that the computer saves the value of Kalman Gain as 1 because of truncation of the float value. That is why, we need to choose reasonable P value for the Kalman value to converge to the real value. We choose 1 for the P value in this experiment.

Note that, these calculated values are prone to error, so during the implementation of the code, we will change the Q and R around these values and tune the parameters by finding out the most robust parameters.

3 Reference

1. https://web.mit.edu/kirtley/kirtley/binlustuff/literature/control/Kalman