Kalman Filter Applied to an Arduino DAQ for a Temperature Control System

Erick Andrés Obregón Fonseca
erickof@estudiantec.cr
MSc in Electronics – Microelectronics Emphasis
Instituto Tecnológico de Costa Rica

***Abstract*—Precise temperature surveillance is indispensable across a myriad of everyday and industrial contexts. To tackle this challenge, this paper delves into the exploration of employing a Kalman filter to proficiently attenuate noise in the measured parameter.**

For this endeavor, an Arduino DAQ was implemented for temperature measurement. Voltage reading were obtained using the Simulink's analog output block and subsequently converted to temperature units through a MATLAB function. Subsequently, the Kalman filter was applied to mitigate the signal noise.

***Keywords*—Arduino, Control System, Kalman Filter, Matlab, Simulink, Temperature**

I. Introduction

In previous research [1], the significance of the heating or temperature control systems across a range of industries including biology, biotechnology, food supply chain, transportation, automotive, agriculture, buildings, and beyond was examined. During the prior study, the methodology was focused on modeling the heating system and implementing a least-squares filter using Simulink. For this work, the approach is centered on the Kalman filter applied to an Arduino Data Acquistion (DAQ) System for a temperature control system.

I-A. Kalman Filter

Kalman filter anticipates forthcoming system states by analyzing past and present states via prediction and update stages [2]. It is employed to gauge system parameters and diminish noise-induced errors during estimation [3]. Ideal for dynamic systems with memory constraints, Kalman Filter is well-suited for real-time and embedded applications [4]. Extended Kalman Filter, tailored for nonlinear systems, often yields superior outcomes compared to conventional techniques [4, 5].

A recursive mean-squared state filter is called a Kalman filter because it was developed by Kalman around 1959 [6]. For the estimator, it is considered a linear feedback system for the state and output estimates, as shown in Eq 1 [7].

$$\dot{\hat{x}}(t) = \bar{F}\ \hat{x}(t) + K\left[\tilde{y}(t) - \bar{H}\ \hat{x}(t)\right], \quad \hat{x}(t_0) = 1 \tag{1a}$$

$$\hat{y}(t) = \bar{H}\ \hat{x}(t) \tag{1b}$$

where $\hat{x}(t)$ denotes the estimate of $x(t)$, K is a constant gain, and $\bar{H} = H = 1$.

The process of deriving the discrete-time Kalman filter commences under the assumption that both the model and measurements exist in discrete-time format. Considering a scenario where the initial state condition x_0 is uncertain, as in Eq 1. Furthermore, it is assumed that both the discrete-time model and measurements are corrupted by noise [7]. So the "truth" model for this is given by Eq 2.

$$\mathbf{x}(k+1) = \Phi\mathbf{x}(k) + \Gamma\mathbf{u}(k) + \Upsilon\mathbf{w}(k) \tag{2a}$$

$$\tilde{\mathbf{y}}(k) = \boldsymbol{H}\mathbf{x}(k) + \mathbf{v}(k) \tag{2b}$$

where $\mathbf{v}(k)$ and $\mathbf{w}(k)$ are assumed to be zero-mean Gaussian white-noise processes, which means that the errors are not correlated forward or backward in time so that

$$E\left\{\mathbf{v}(k)\mathbf{v}(j)^T\right\} = \begin{cases} 0 & k \neq j \\ R_k & k = j \end{cases} \tag{3}$$

,

$$E\left\{\mathbf{w}(k)\mathbf{w}(j)^T\right\} = \begin{cases} 0 & k \neq j \\ Q_k & k = j \end{cases} \tag{4}$$

and

$$E\left\{\mathbf{v}(k)\mathbf{w}(j)^T\right\} = 0 \tag{5}$$

So the final estimator is given by Eq 6

$$\hat{\mathbf{x}}^-(k+1) = \Phi\hat{\mathbf{x}}^+(k) + \Gamma\mathbf{u}(k) \tag{6a}$$

$$\hat{\mathbf{x}}^+(k) = \hat{\mathbf{x}}^-(k) + K\left[\tilde{\mathbf{y}}(k) - H\hat{\mathbf{x}}^-(k)\right] \tag{6b}$$

with the following stages [7]:

1. Initialize:

$$\hat{\mathbf{x}}(t_0) = \hat{\mathbf{x}}_0 \tag{7a}$$

$$P_0 = E\left\{\tilde{\mathbf{x}}(t_0)\tilde{\mathbf{x}}(t_0)^T\right\} \tag{7b}$$

2. Gain:

$$K(k) = P^{-}(k)H^{T}(k)\left[H(k)P^{-}(k)H^{T}(k) + R(k)\right]^{-1} \tag{8}$$

3. Update:

$$\hat{\mathbf{x}}^{+}(k) = \hat{\mathbf{x}}^{-}(k) + K(k)\left[\tilde{\mathbf{y}}(k) - H\hat{\mathbf{x}}^{-}(k)\right] \tag{9a}$$

$$P^{+}(k) = \left[I - K(k)H(k)\right]P^{-}(k) \tag{9b}$$

4. Propagation:

$$\hat{\mathbf{x}}^{-}(k+1) = \Phi\hat{\mathbf{x}}^{+}(k) + \Gamma\mathbf{u}(k) \tag{10a}$$

$$P^{-}(k+1) = \Phi(k) + P^{+}(k)\Phi^{T}(k) + \Upsilon(k)Q(k)\Upsilon^{T}(k) \tag{10b}$$

II. Methodology

III. Results and Discussion

IV. Conclusions

The Kalman filter provides an optimal method for estimating the state of a dynamic system. By efficiently combining noisy sensor measurements with the system's dynamic model, it yields accurate and robust estimation of the true state. Its adeptness at mitigating measurement noise and process disturbances bolsters the accuracy and dependability of state estimation, proving invaluable in scenarios demanding meticulous measurements. It is versatile and adaptable to a wide range of dynamic systems, including linear and nonlinear systems, making it applicable across various domains such as aerospace, robotics, finance, and signal processing. With its recursive nature and computational efficiency, the Kalman filter is a prime candidate for real-time applications, facilitating continuous state estimation and system control while imposing minimal computational overhead.

Arduino serves as a versatile platform appealing to hobbyists, educators, and professionals alike. Offering a pliable framework, it fosters the creation of an extensive range of electronic projects. Boasting an intuitive interface and abundant online support, Arduino serves as a gateway to the realms of electronics and programming, facilitating learning and experimentation with ease. Its compatibility with a various sensors, actuators, and communication modules streamlines incorporation into diverse projects, empowering users to forge interactive and interconnected systems effortlessly. Moreover, its straightforward operation, cost-effectiveness, and swift prototyping prowess render it an ideal candidate for iterative refinement and speedy proof-of-concept validation, expediting the genesis of inventive solutions.

The Simulink with Arduino Add-on combines the Simulink's robust simulation capabilities with the Arduino's versatility, enabling users to prototype and test intricated control algorithms and system designs in a virtual environment prior hardware implementation. This integration facilitates smooth communication between Simulink models and Arduino hardware, allowing for real-time interaction and control of physical devices directly from the Simulink environment. By leveraging Simulink's user-friendly graphical interface and extensive block library in alongside with Arduino's simplicity and hardware flexibility, the add-on expedites the development process, facilitating swift iteration and enhancement of designs for efficient project advancement.

References

[1] E. A. Obregon Fonseca, "Least-Squares Filter Applied to a Temperature Control System using Simulink," 2024, unpublished.

[2] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, 1960.

[3] G. Bishop, G. Welch *et al.*, "An introduction to the kalman filter," *Proc of SIGGRAPH, Course*, vol. 8, no. 27599-23175, p. 41, 2001.

[4] M. Khodarahmi and V. Maihami, "A review on kalman filter models," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, pp. 727–747, 2023.

[5] X. Lai, Y. Huang, X. Han, H. Gu, and Y. Zheng, "A novel method for state of energy estimation of lithium-ion batteries using particle filter and extended kalman filter," *Journal of Energy Storage*, vol. 43, p. 103269, 2021.

[6] J. M. Mendel, *Lessons in estimation theory for signal processing, communications, and control*. Pearson Education, 1995.

[7] J. Crassidis and J. Junkins, *Optimal Estimation of Dynamic Systems*. CRC Press, 01 2004.