Ecole Centrale de Nantes

Autonomous Vehicles: Lab 2 Kalman filter

Isabella-Sole Bisio

December 3, 2020

Contents

List of Figures

1 Introduction

The following is the report for the second assignment of the Autonomous Vehicles course at Ecole Centrale Nantes. The Laboratory has been performed with the use of **MATLAB R2020a**. To properly check and run the simulation of this part of the Laboratory the Main file is the one named *Kalman_ Filter*: the subsections of the code correspond to the ones in the PDF. The report contains an analysis of the implemented Kalman filter. In this Lab a DC motor is considered, driven by the input voltage *u(t)*. The angular position of the rotor $\theta(t)$ is measured with an incremental encoder, having a precision L = 512 angles per lap, which provides the measure *y(t)* of $\theta(t)$. $\Omega(t)$ is the derivative of $\theta(t)$, thus it is the angular velocity. The purpose is to perform a velocity control, and in order to do this, an online estimation of $\theta(t)$ and $\Omega(t)$ should be done from *(t)* and *y(t)*.
To achieve this, the computation of a model for the system will be required, alongside with the definition of a Kalman Filter.

2

2.1 Input voltage

u(t) is the input voltage of our system and it is a zero-mean square wave with a period $\Delta = 100$ ms, and a peak-to-peak amplitude A = 0,1 V. In order to sample the signal a sample time of Ts = 1 ms is used. At this level a Matlab function was created that provides the sampled input in an interval of time with a choosen duration D. The function created is the following:

$$u = inputvoltage(D, A, Delta, Ts); \tag{1}$$

where the inputs are: D simulation time (set to 1 second), A is the peak-to-peak amplitude (set to 0.1 Volt), $Delta$ is the period of my zero-mean square wave signal (set to 0.1 second) and the Ts is the sample time (set to 0.001 second). The function gives as output u, a column vector that contains the sampled input.

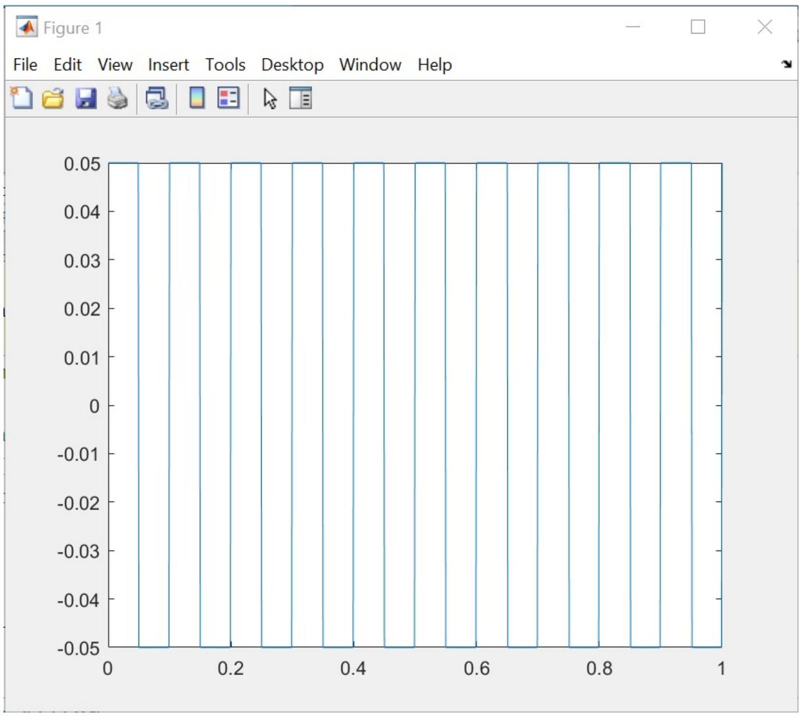

Figure 1: Square wave signal

2.2 System modeling and simulation

In this section the main objective is to find a column matrix y with same size of u, that contains all the evolution of y_n (measure provided by the incremental encoder that is a quantization of the actual angular position $\theta(n)$), and x a 2-columns matrix which contains the evolution of the state vector. To perform this task, first of all, I had to pass from continuous to discrete time: this action is possible since the input *u(t)* is constant between two sampling times. Thus, the continuous-time can be sampled, using Ts, without any approximation using the step invariance method or the zero-order-hold method. Starting from the beginning, a continuous-time system can be associated to the following state space representation:

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ \\ y(t) = Cx(t) + Du(t) \end{cases} \tag{2}$$

For our DC motor we have:

$$\begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & -1/T \end{bmatrix} x + \begin{bmatrix} 0 \\ G/T \end{bmatrix} u \\ \\ \theta = \begin{bmatrix} 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \end{bmatrix} u \end{cases} \tag{3}$$

Now, using the *c2dm* Matlab function, I can convert my system to a discrete one, using the zero-order-hold *'zoh'* method. The system becomes:

$$\begin{cases} x_{n+1} = \tilde{A}x_n + \tilde{B}u_n \\ \\ y_n = \tilde{C}x_n + \tilde{D}u_n \end{cases} \tag{4}$$

At this point the function

$$[y, x] = \mathit{simulate}(u, G, T, Ts, L, x1); \tag{5}$$

was created in order to obtain y_n. It takes as input the u column vector that contains the sampled input, G and T values suggested by the text, Ts sample time, L precision of the encoder and *x1* the initial state vector set to 0. The outputs are *y*, a column matrix (same size as u) which contains the evolution of the output y_n and *x*, a 2-columns matrix which contains the evolution of the state vector. This function mainly:

- Computes the state given the input
- Computes the output given the input
- Performs the observation of the angle with the encoder

Since the angle $\theta(t)$ is measured by using an incremental encoder with a precision of $L = 512$ angles per lap, the output vector will be produced accordingly. The output vector y_n is, in fact, a quantisation of the result given by the model according to the precision of the sensor employed. The formulation that has been used is:

$$y_n = \mathit{round}((\theta * L/2)/\pi) * 2\pi/L \tag{6}$$

These steps are repeated in a for loop and the already mentioned output vectors are filled.

2.3 Kalman Filter

The goal of this section is to build a Kalman filter in order to estimate x_n. The quantisation noise of the incremental encoder W_n is now kept into account, along with its variance R: to pick a value for R it is convenient to model the quantisation error as a uniform random variable, as suggested by the PDF instructions. Hence, the variance is given by the formula

$$R = (b-a)^2/12 \tag{7}$$

where the *(b-a)* interval corresponds to $(2\pi)/L$.
Furthermore, the actual input is added a white noise V_n with its variance Q, which will be chosen by trial and error. The first things to estimate are $\hat{x}_{1/0}$ and $P_{1/0}$. From the PDF instructions it is possible to read that the motor is initially stopped (the second component of $\hat{x}_{1/0}$ is equal to zero), but the initial angular position is unknown; I have chosen a value close to 0 to allow a fast convergence which avoids sharp slopes. The value of $P_{1/0}$ is 0 everywhere (since the initial velocity is null, also its variance will be null) except for the element (1,1) that will be $\sigma_\theta^2 = \pi^2/3$.

$$P_{1/0} = \begin{bmatrix} \sigma_\theta^2 & 0 \\ 0 & 0 \end{bmatrix} \tag{8}$$

The model that includes also quantisation and input errors can be expressed as the following system

$$\begin{cases} x_{n+1} = \tilde{A}x_n + \tilde{B}u_n + V_n \\ \\ y_n = \tilde{C}x_n + \tilde{D}u_n + W_n \end{cases} \tag{9}$$

2.3.1 Kalman Filter

The Kalman Filter is nothing but a special case of the Bayesian Filter, in which a linear model is considered and the noise is assumed to be Gaussian. It is very useful while trying to estimate an unknown variable and obtaining a good approximation of it. The function

$$[x_e] = kal(y, u, G, T, Ts, L, x_{1/0}, P_{1/0}, Q, flag); \tag{10}$$

was built in order to apply this Filter to our problem and to estimate the state vector.
The input parameters are the ones already used and explained previously in this report, whereas the output x_e is a 2-columns matrix which contains the evolution of the state vector estimation. This function is basically an implementation of the Kalman Filter, in which the following steps are performed:

- n^{th} observation prediction

$$\begin{cases} \hat{y}_{n-1} = C_d x_{n-1} + D_d u(n) \\ \\ C_{yy} = C_d P_{n-1} C_d^T + R \\ \\ C_{xy} = P_{n-1} C_d^T \end{cases} \tag{11}$$

- n^{th} state estimation

$$\begin{cases} \hat{x}_n = \hat{x}_{n-1} + C_{xy}C_{yy}^{-1}(y(n) - \hat{y}_{n-1}) \\ P_n = P_{n-1} - C_{xy}C_{yy}^{-1}C_{xy}^T \end{cases} \tag{12}$$

- $(n+1)^{th}$ state prediction

$$\begin{cases} \hat{x}_{n+1} = A_d\hat{x}_n + B_d u(n) \\ P_{n+1} = A_d P_n A_d^T + Q \end{cases} \tag{13}$$

The following figures represent the trend of the two variables of the state vector, θ angle and Ω angular velocity estimated with the Kalman Filter. Notice that to obtain these figures the value of Q was set to 0.005. To reproduce the figures, uncomment the section *Plot fot test* in the *kal.m* function.

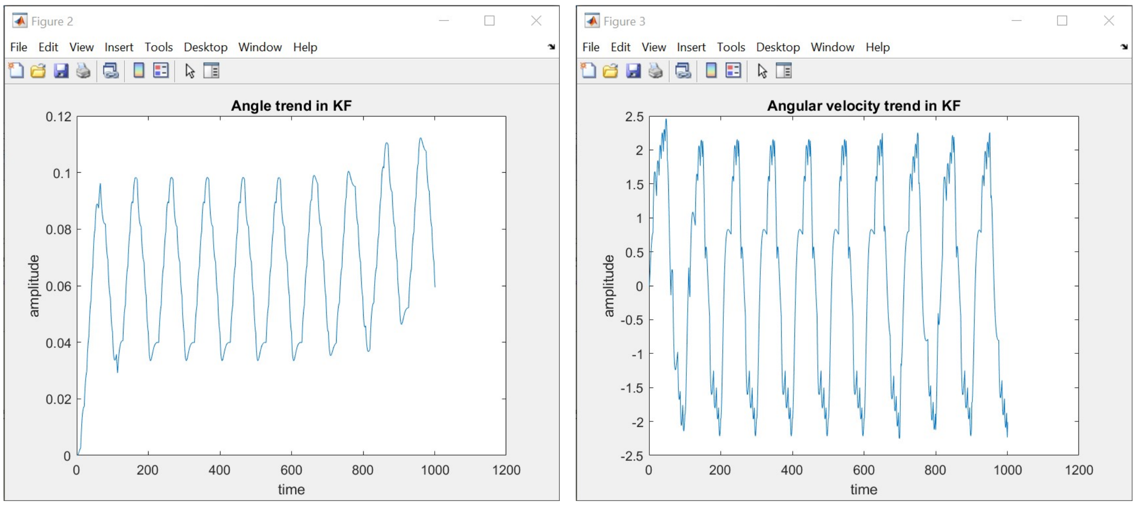

Figure 2: Angle trend, Angular velocity trend in Kalman Filter

2.3.2 Static Kalman Filter

If the matrices A_d, C_d, Q_n and R_n are time-independent, then the matrix $P^{|n-1}[n]$ tends to the limit $P[\infty]$ which fulfills the discrete Riccati equation; in this case we obtain a Stationary Kalman Filter that is simplified in its equations. To perform this filtering the function

$$[xe_s] = kal_s(y, u, G, T, T_s, L, x_{1/0}, Q, flag); \tag{14}$$

was created, in which the input parameters are the ones already used and explained above, whereas the output x_{es} is a 2-columns matrix which contains the evolution of the state vector estimation. The steps pf this function are the following:

- n^{th} observation prediction

$$\hat{y}_{n-1} = C_d x_{n-1} + D_d u(n) \tag{15}$$

- n^{th} state estimation

$$\hat{x}_n = \hat{x}_{n-1} + K(\infty)(y(n) - \hat{y}_{n-1}) \tag{16}$$

- $(n+1)^{th}$ state prediction

$$\hat{x}_{n+1} = A_d\hat{x}_n + B_d u(n) \tag{17}$$

The following figures represent the trend of the two variables of the state vector, θ angle and Ω angular velocity estimated with the Stationary Kalman Filter. Notice that to obtain these figures the value of Q was set to 0.005. To reproduce the figures, uncomment the section *Plot fot test* in the *kal_s.m* function.

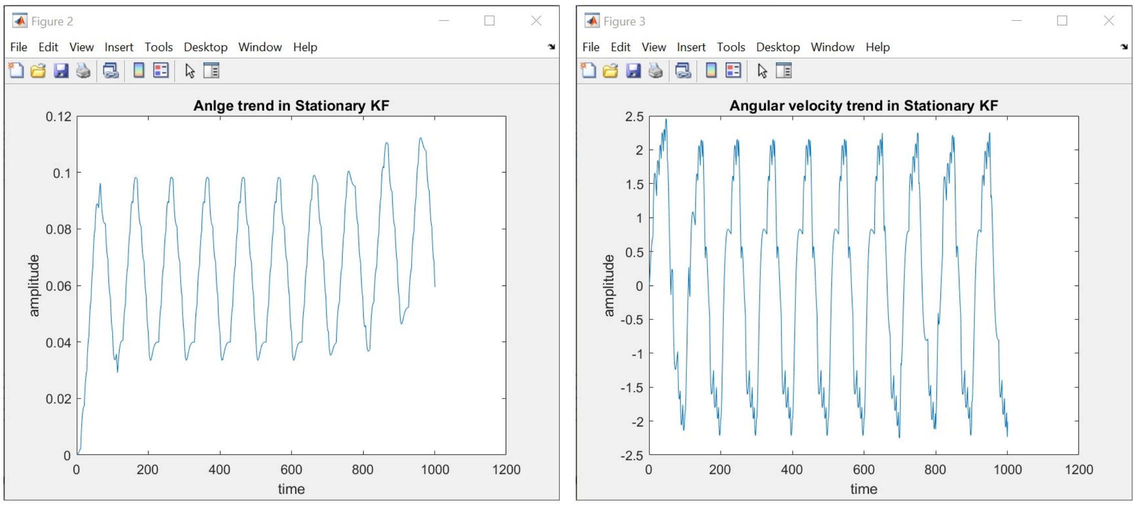

Figure 3: Angle trend, Angular velocity trend in Stationary Kalman Filter

2.4 Simulations

This section is the conclusion of the whole work; here in fact the result of the different behaviours are analysed. We were asked to compare the estimation of the position and the velocity given by both filters for different values of Q using two different models:

- CASE 1: a perfect model of the system, where T_{actual} and T_{filter} have the same value
- CASE 2: a rough model of the system, where T_{actual} and T_{filter} have a 5ms mismatch

Before comparing the results, there are some remarks that should be taken into account:
the initial estimation of the state is set with a small difference of 0.05, as suggested in the PDF instructions, in order to avoid a big initial correction that may lead to a noteworthy difference in the results. Furthermore, there are no information for

what concerns the initial position, meaning that it could then be set to any value between 0 and 2π.
Moreover, the value of the variance of the white noise Q is set arbitrarily; it is in fact a tuning parameter that has to be set by trial and error. It allows to tune the trade-off between the information brought by the model and the information brought by the observation. The value that I believe to be the best is $Q = 0.005$; the input signal as a peak-to-peak amplitude of $A = 0.1$ so it seems reasonable to assign a value which is one tenth of it.
I have decided to plot CASE 1 and CASE 2 in a single figure window in order to make a better comparison between the two. Moreover, in each case three lines are plotted: the red one representing the actual model, the green one representing the Kalman Filter estimation and the blue one representing the Stationary Kalman Filter estimation.

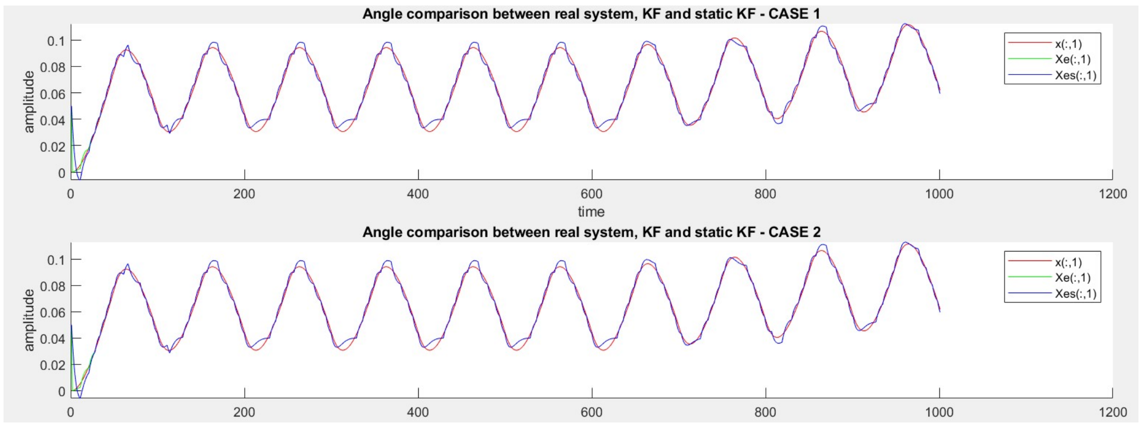

Figure 4: Angle comparison between real system, KF and Stationary KF, Q = 0.005

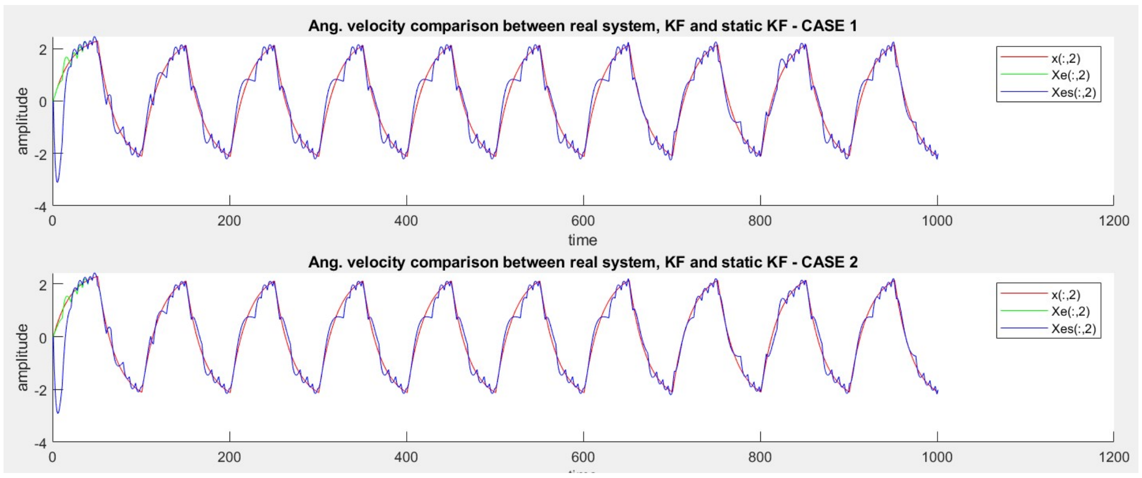

Figure 5: Anglular velocity comparison between real system, KF and Stationary KF, Q = 0.005

It is possible to see that with both system's models the two filters are able to estimate the state with great precision: at a glance the two graphs seems indistinguishable. Conversely, the difference can be easily seen between the two Kalman

Filters: at the initial instances the normal version (green line) takes less time to converge to the real model values, whereas the Stationary version (blue line) needs more time; this happens because in the Static case the Kalman gain is an approximation and this explains why it takes longer.
I have also tried to modify the value of Q in order to increase or decrease the trust in the model: intuitively, the more noise is introduced, the more difficult it is for the filter to estimate the real values of the state since the filter trusts more the measurements with respect to the estimations. Setting Q higher or lower is possible to see that when Q is equal to 0.1 the Kalman Filter cannot handle this noise properly and a lot of distorsion is still present. On the other hand, when setting Q to 0.0005 the model trusts more the estimations and the noise is so little that is nearly completely eliminated by it.

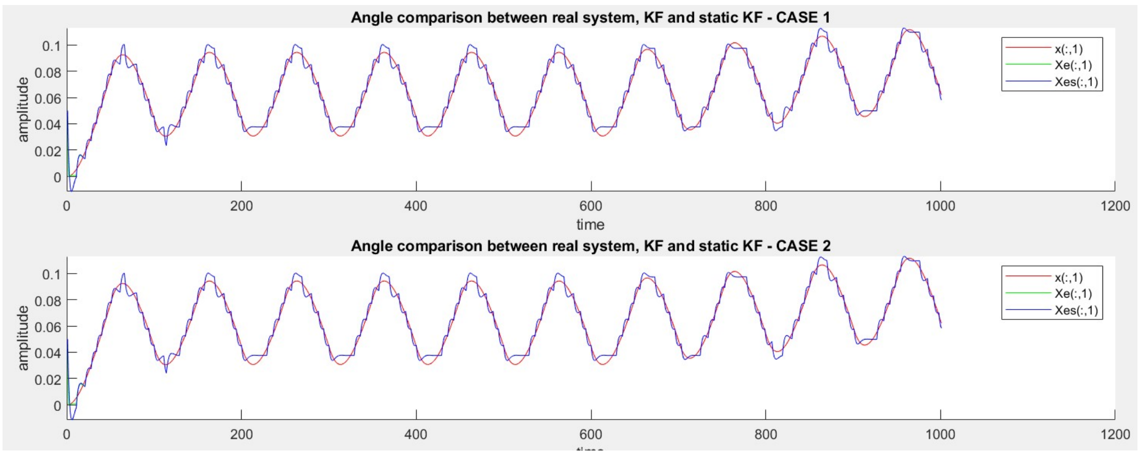

Figure 6: Angle comparison between real system, KF and Stationary KF, Q = 0.1

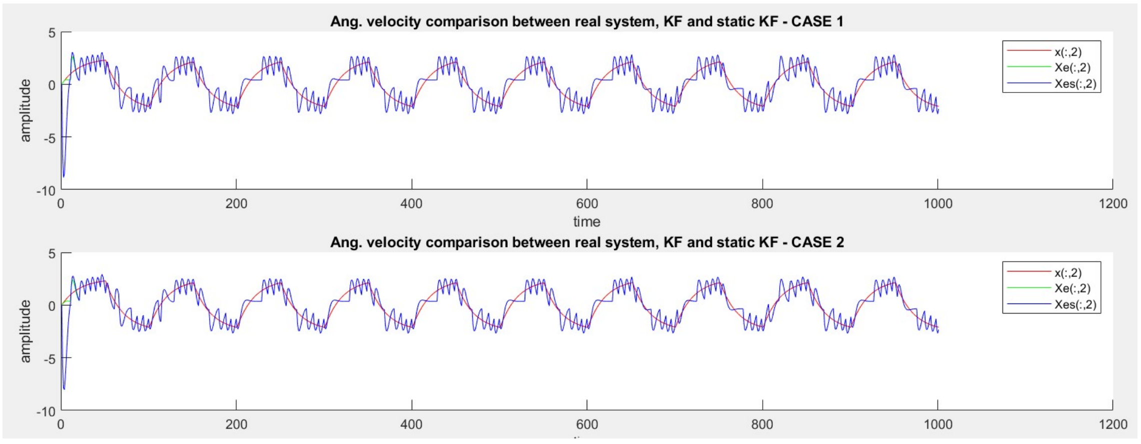

Figure 7: Angular velocity comparison between real system, KF and Stationary KF, Q = 0.1

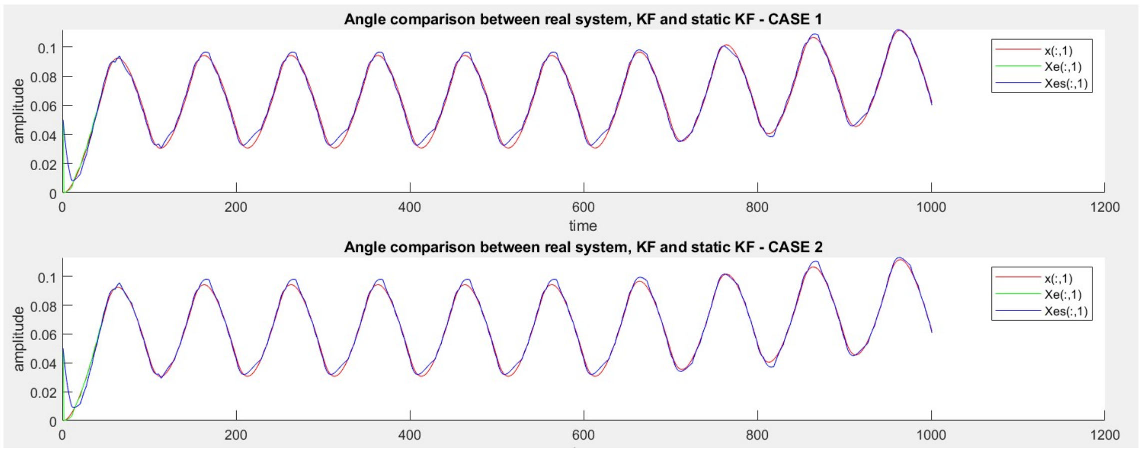

Figure 8: Angle comparison between real system, KF and Stationary KF,Q = 0.0005

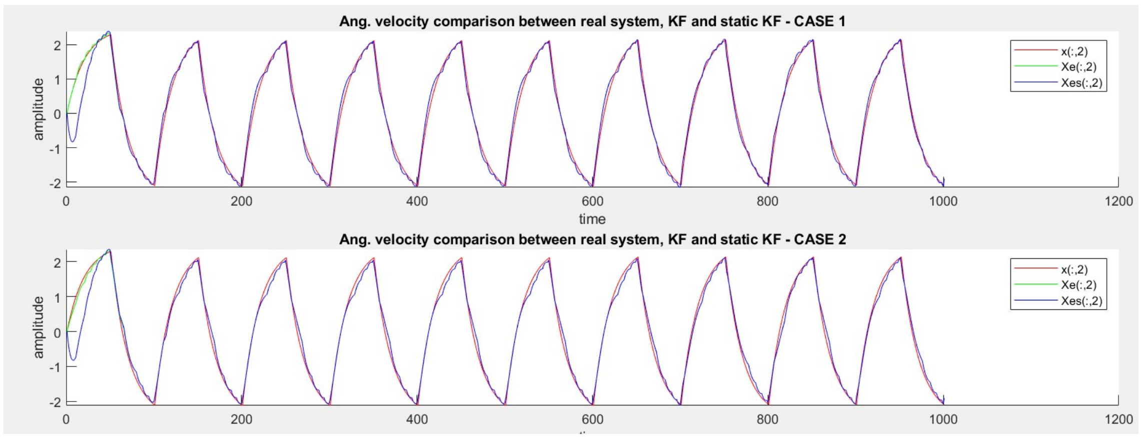

Figure 9: Angular velocity comparison between real system, KF and Stationary KF, Q = 0.0005