RL Exercise 1

Ludovica Mazzucco
Department of Control and Computer Engineering
Politecnico di Torino
Italy
s315093@studenti.polito.it

Abstract: The purpose of this document is to present the results obtained working on the second Robot Learning laboratory activity, which was focused on the application of the Extended Kalman Filter to an unactuated pendulum. Pieces of code and plots are shown for the sake of completeness.

Keywords: Exercise1, Robots, Learning

1 Introduction

1.1 Bayesian Filters

In real world systems we usually do not have a precise knowledge about the dynamics behind their evolution in time, nor we can fully trust measures drawn from sensors outputs, since a non-negligible noise term is generally added to them. It might happen that in certain situations sensors are not even an option due to an harsh environment. Nonetheless, it is fundamental both for the system itself and for a third part to get to know the exact values of the state variables at each time step. Take for instance an industrial manipulator, it always has to recognise the angular position of its joints or their velocity in order to fulfill its assignments through the end-effector, plus it is a prerequisite for its controller to be updated with the current configuration of the system it regulates in order to drive it to a desired behaviour. In this context, (Bayesian) filters are of huge importance. As a matter of fact, starting from the very basic rule of the whole probability theory, the Bayes' law [1], they are able to exploit the current (noisy) measurements to refine their $belief$ about the actual state of the system time by time and eventually getting to its real value.
More precisely, this homework is focused on a specific kind of Bayesian Filter, the Extended Kalman Filter, which is, as the name suggests, the extension of the Kalman Filter to non-linear systems.

1.2 Case of Study: the Unactuated Pendulum

The system considered for the experiment is the simple unactuated pendulum, which is free to swing around an equilibrium vertical axis under the only influence of the gravity force through an unextensible wire of length 1 and negligible mass. From the Newton's Law we can derive its continuous time dynamics:

$$\dot{x} \begin{cases} \dot{x_1} = x_2 \\ \dot{x_2} = -\frac{g}{l} sin(x_1) \end{cases} \tag{1}$$

where we assumed the state vector $x = [x_1 \quad x_2]$ represents respectively the angle θ formed by the pendulum string with the central axis and the angular velocity $\dot{\theta}$.
We consider, also, the presence of a sensor able to measure only the first of the two state variables, with an additional noise stochastic term ϵ that adds to it and tampers the true value. The only assumption we have to make about it in order to work with the Kalman Filter is that it is normally

[1] $PosteriorProbability = \frac{Likelihood \times PriorProbability}{Evidence}$

7th Conference on Robot Learning (CoRL 2023), Atlanta, USA.

distributed with variance 0.05 (generally a reliable value for sensor noise which is actuallt unknown to the user). Thus, we will refer to "observation model" as the following expression:

$$z(t) = h(x,t) = [1 \quad 0]\, x + \epsilon = x_1 + \epsilon \tag{2}$$

No uncertainties are supposed to be present in the motion model, due to the fact we can rely on an almost total knowledge of the physics behind. It has to be underlined the presence of the sine function inside equation [1], since this makes the considered model non linear. Hence, as it was disclosed before, the basic form of the Kalman Filter is not enough and we have to resort to the EKF.

2 Experimental Results

In this paragraph are reported the main steps followed in order to develop an Extended Kalman Filter able to reconstruct the state of the unactuated pendulum at each time step given a sensor obeying to the observation model represented in [2].

2.1 EKF setting: Discretization and Linearization

First of all, we need to cope with the non linearity embedded in the dynamics of the pendulum [1], given that if we ignore it than all the prediction steps will fail in generating Gaussians, thus breaking the principles at the base of the KF. The best that we can do is linearizing the system each time around the predicted state at (t-1) in order to approximate with a Gaussian the real probability distribution. Furthermore, the filter works with discrete dynamics while our initial model is expressed in continuous time, this means that we additionally have to discretize it.
The tool exploited for the latter goal is the Euler forward method, then the discrete version of [1] will look like this:

$$x_t = x_{t-1} + \Delta t \frac{dx}{dt}\|_{x_{t-1}} \tag{3}$$

Since the derivative is indeed the one obtained in [1] it can be substituted:

$$\begin{bmatrix} x_{1,t} \\ x_{2,t} \end{bmatrix} = \begin{bmatrix} x_{1,t-1} \\ x_{2,t-1} \end{bmatrix} + \Delta t \begin{bmatrix} x_{2,t-1} \\ -\frac{g}{l} sin(x_{1,t-1}) \end{bmatrix} \tag{4}$$

At this point, it is necessary to apply the linearization by obtaining the Jacobian corresponding to the above system:

$$J_f = \begin{bmatrix} 1 & \Delta t \\ -\Delta t \frac{g}{l} cos(x_1) & 1 \end{bmatrix} \tag{5}$$

In what the $observation\ model$ is concerned, similar steps could be followed. Nonetheless, in this specific case there is no need to further discretize and linearize since it can be directly used without other changes. The corresponding Jacobian is the matrix $[1 \quad 0]$ itself.

Finally, the discrete linear system for the pendulum will look like the following:

$$\begin{cases} x_t = J_f x_{t-1} \\ z_t = \begin{bmatrix} 1 & 0 \end{bmatrix} x_t \end{cases} \tag{6}$$

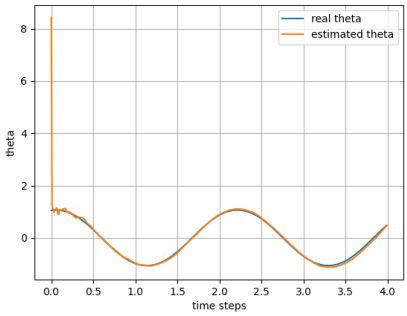
2.2 Implementation of the EKF: influence of Q and R

Either the continuous state space model [1] and the discretized one [6] are paramount for the right behaviour of the filter. The procedure through which it will output the estimation of angle θ related to the pendulum can be summarized as follows:

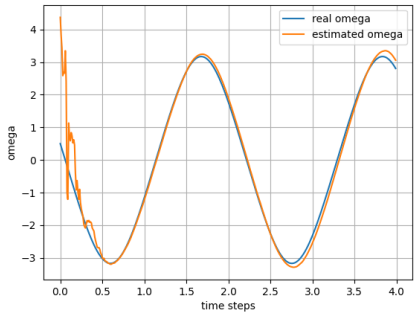
- *prediction step.* Given the (estimated and corrected) state at time $(t-1)$, the so called "belief" $Bel(x_{t-1})$, exploit the knowledge about the model to predict the gaussian corresponding to the state at the next time step $\overline{Bel}(x_t)$, that will show an higher variance with respect to the previous one since it introduces uncertainty;

- *update step.* Take advantage of the measure at time t (still a gaussian because of the random noise) to refine the prediction made. This will generate a gaussian with a lower variance and the mean computed as the convex combination of the two weighted by their normalized covariances , in a way for the new curve to be more shifted toward the one more "reliable".

Q and R are the covariance matrices of the noise in the motion model and of the observation model respectively. Since we supposed the uncertainties in our system only belong to the use of the sensor, Q is set with values close to zero, meaning that the bell curves obtained in the prediction have a tight shape around the mean. Conversely, R reflects the variance of the noise added to the sensor output, in our case 0.05. Increasing this value causes a widening of the observation gaussian, intensifing also the uncertainty. This is proven by the simulation of the EKF on the pendulum, where the plots of θ and ω highlights an initial bias between the true and estimated quantities, which is more evident the higher greater is R.

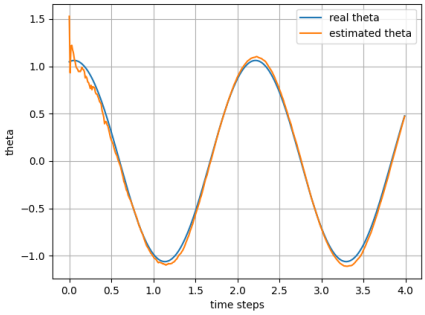


(a) Real and estimated theta

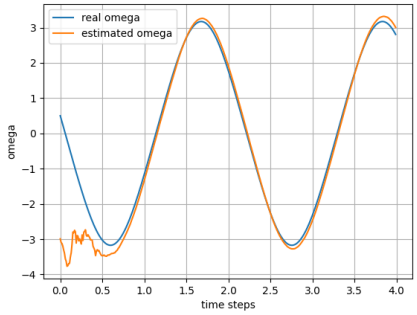


(b) Real and estimated omega

Figure 1: EKF performance using R=0.05

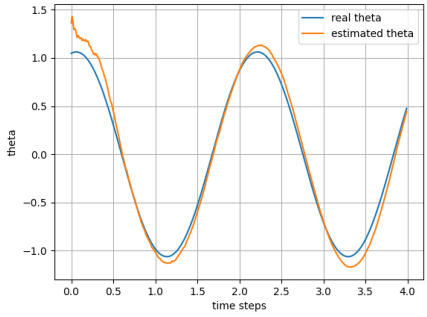


(a) Real and estimated theta

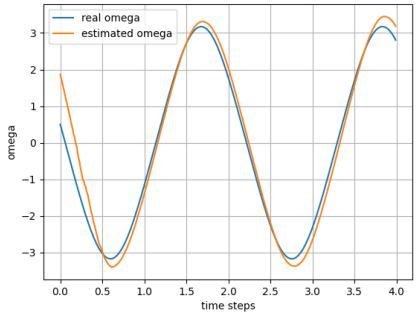


(b) Real and estimated omega

Figure 2: EKF performance using R=0.5



(a) Real and estimated theta



(b) Real and estimated omega

Figure 3: EKF performance using R=5

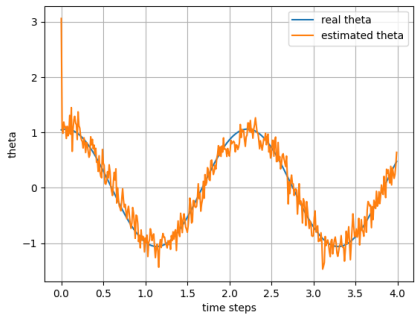
In order to better understand the different behaviour of the filter changing R, it would be wise to numerically evaluate its performance by means of the Root Mean Squared Error (RMSE), obtained through the following formula, taking into account that in this case the square is generalized with the squared 2-norm since the true and estimated states are vectors:

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(x_t^{est} - x_t^{true})^2} \tag{7}$$

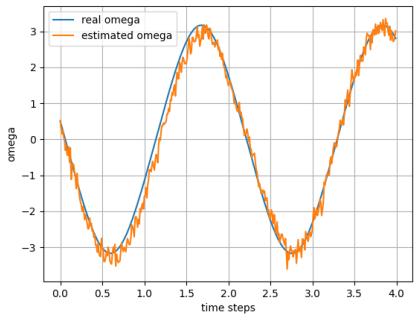
Thus, the slight degradation of performance is highlighted by the values it assumes in the different cases (considering $n = 400$):

- R=0.05, RMSE= 0.6926
- R=0.5, RMSE= 1.0929
- R=5, RMSE= 2.5109

An annotation about the latter results. RMSE does not output a deterministic value since the whole process is stochastic, in other words, running the simulation multiple times we will end up in having different digits, nevertheless we can appreciate on average an increasing trend of this metric with respect to the increasing variance R.
All in all, despite the augmenting variance on the observation model, the EKF is equally able to cope with it and to track to the limit the tendency of the real function. This is due to the low Q counterbalancing the bad effects of R as a consequence of the products between the gaussians as explained above. It is interesting to examine what happens changing Q along with R. The plots below show how noisy the estimation becomes, with a lot of spikes.



(a) Real and estimated theta



(b) Real and estimated omega

Figure 4: EKF performance using R=0.05 and $Q = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$

2.3 Implementation of the EKF in ROS

The last part of this assignment concerns the simulation of the whole system composed by the pendulum, the sensor and the filter in ROS, in order to visualize what happens in a real case example where those three main characters interact with each other.
For this purpose, three nodes has been created using Python scripts corresponding to each one of them and their structure can be enclosed as follows:

- **Pendulum.py**: it represents just the motion of the pendulum itself with the non linear model and the values of θ and $\dot{\theta}$ are published to the topic "TrueState" through a message type "Floats", a vector of floats;
- **Sensor.py**: it reflects the behaviour of the sensor, listening to the TrueState topic and adding a random noise normally distributed with variance 0.05 to the pendulum angle, which is then published to another topic "SensedState" through the "Float32" message type, since unlike the previous case it is just a scalar;
- **Ekf.py**: this is the node where all the computations of the estimated state happen. It directly imports the "ExtendedKalmanFilter.py" module in order to be able to create inside it the class EKF and perform the prediction and update steps making use of the value received from the SensedState topic. In turn, it publishes the estimation of the state to a new topic called "EstimatedState", in favor of the node related to the $rqt\_multiplot$, so that to be able to plot the different variables in function of time.

The result of the above process is proposed here.

For the sake of completeness, the following images show the different terminals during messages exchange between nodes and various information about topics and nodes theirselves.

2.4 Application of the EKF to the Double Pendulum

After having explained the main steps performed in order to generate a Kalman Filter able to track the behaviour of the simple pendulum, this section focuses on its application to the double pendulum. Very briefly, it is a pendulum whose mass is joint to another string supporting a second mass; the same hypothesis stated at the beginning for the simple version about the physical structure still hold for this one (i.e wire of negligible mass). We are now interested in modelling the dynamics not also of the angle subtended by the first thread with respect to the vertical axis but by the second one as well, referred here as θ_1 and θ_2 respectively. The reasoning under this kind of EKF is quite the same of the previous one, it just has to be adapted modifing the equations describing the motion of this new system. Indeed, this is the trickiest side of the job, since the double pendulum seems to be

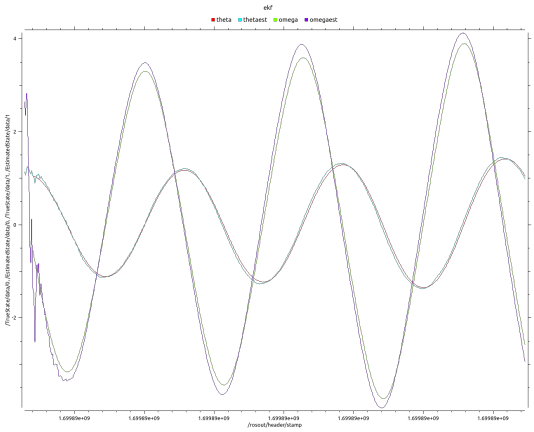


Figure 5: Multiplot of the true and estimated angle/ angular velocity

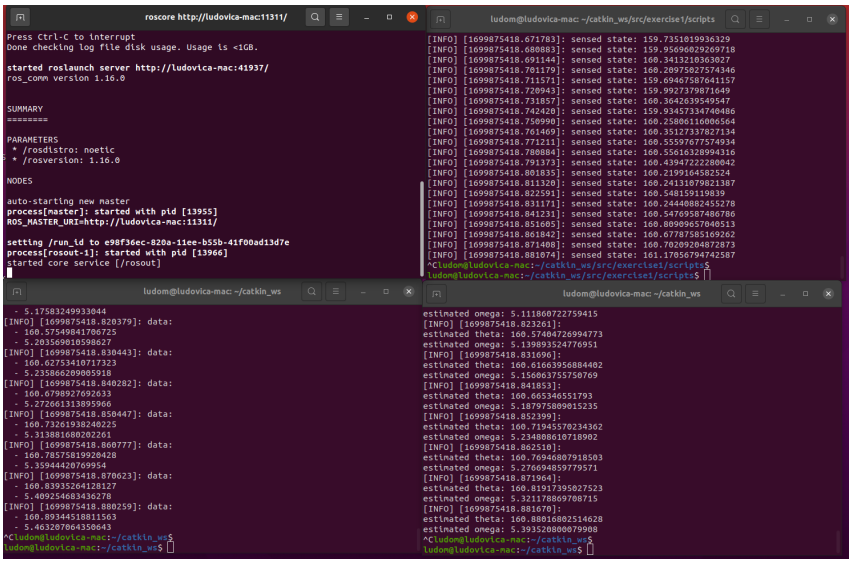


Figure 6: up-left Roscore terminal, bottom-left Pendulum terminal, up-right Sensor terminal, bottom-right EKF terminal.

strongly non linear with the variables coupled with each other, anyway the smartest move would be to resort to the Langrangian equations based on the kinetic and potential energies enclosed in it [2]. Considering for the system two degrees of freedom corresponding to θ_1 and θ_2, we will end up in extracting a system of two equations for the pendulum dynamics, which after some calculations and mathematical reorganization looks like

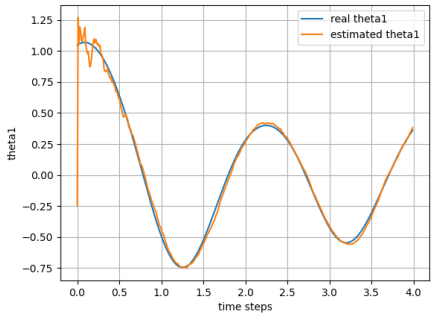
$$\begin{cases} \ddot{\theta_1} = \frac{-l_2 m_2 sin(\theta_1-\theta_2)\dot{\theta_2}^2 - l_1 m_1 \dot{\theta_1} - g sin(\theta_1)(m_1+m_2) + m_2 cos(\theta_1-\theta_2) g sin(\theta_2) + m_2 cos(\theta_1-\theta_2) l_1 \dot{\theta_1}^2 sin(\theta_1-\theta_2)}{l_1(m_1+m_2)+l_1 m_2 cos^2(\theta_1-\theta_2)} \\ \ddot{\theta_2} = \frac{cos(\theta_1-\theta_2)[-l_1 m_2 cos(\theta_1-\theta_2) sin(\theta_1-\theta_2)\dot{\theta_1}^2 + l_1 m_1 \dot{\theta_1} + l_2 m_2 sin(\theta_1-\theta_2)\dot{\theta_2}^2 - m_2 cos(\theta_1-\theta_2) sin(\theta_2) + g sin(\theta_1)(m_1+m_2)]}{l_2 m_2 cos^2(\theta_1-\theta_2) + l_2(m_1+m_2)} + \\ + \frac{-l_1 \dot{\theta_1}^2 sin(\theta_1-\theta_2) - g sin(\theta_2)}{l2} \end{cases} \tag{8}$$

[2] $L = T - U$ $\frac{d}{dt}\frac{dL}{d\dot{q_i}} - \frac{dL}{dq_i} \forall i$

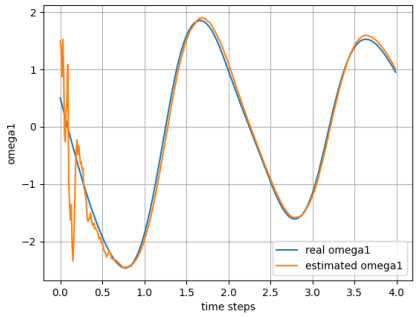
The latter is useful for the representation of the motion model of the double pendulum, with its state being

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \dot{\theta}_1 \\ \theta_2 \\ \dot{\theta}_2 \end{bmatrix}$$

Apart from the discretization and linearization performed likewise in paragraph 2.1, it is worth to underline that in this case the sesnor exploited either measures the first and the second angles adding the same noise seen before (with variance 0.05). Hence, R becomes a matrix 2×2, the vector of measures obviously 2×1 and the Jacobian of the observation model 2×4 as the output must be of dimention 2. Eventually, starting a simulation of 400 steps, the resulting plots highlight the remarkable capability of the EKF to well estimate the true state even though the double pendulum represents a teasing task compared to the simple version.

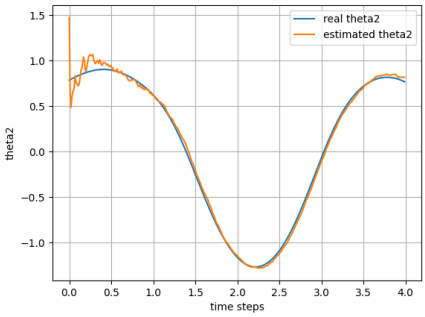


(a) Real and estimated theta 1

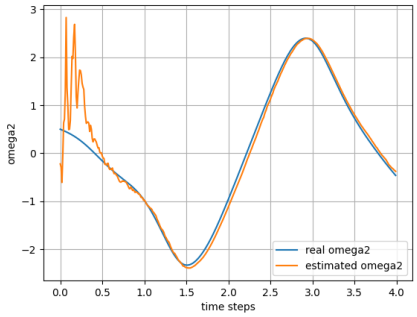


(b) Real and estimated omega 1

Figure 7: EKF performance on the double pendulum



(a) Real and estimated theta 2



(b) Real and estimated omega 2

Figure 8: EKF performance on the double pendulum

Similarly to what it was done for the simple case of the pendulum, the performance of the EKF with this quite more complex system can be evaluated through the RMSE, using the yet explicited formula in [7]. What it outputs is $RMSE = 0.3701$.
Lastly, the figure below shows the trajectory identified by the second mass, that is to say the real and the estimated one.

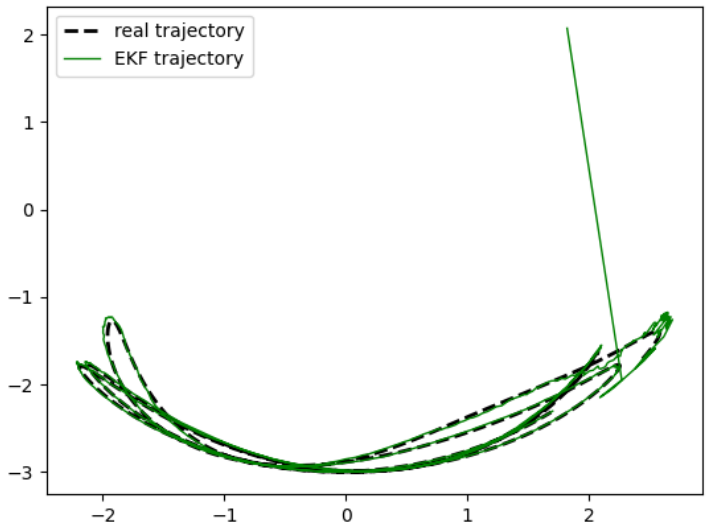


Figure 9: comparison between real trajectory of the pendulum second mass and the estimated one.