Lab 4 - Extended Kalman Filter

Daniel Barmaimon
Probabilistic Robotics - Vibot Master
Degree
University of Girona

May 1, 2014

1 Introduction

Kalman filter is an optimal estimator that infers parameters of interest from indirect, inaccurate and uncertain observations. It is optimal because, if all noise is Gaussian, it will minimises the mean square error of stimated parameter. If the noise is not, extended version of this filter will be the best linear estimator.

2 Tasks to achieve

During this lab the main task is to understand how the extended Kalman filter works and implement it over a simulated robot.

3 Implementation and issues

3.1 Prediction

The first step for each evaluation will be to estimate the values of position, orientation for the robot, and its covariance matrix the next iteration. To get this initial evaluation, as the system doesn't has a linear model, linearization for the functions that defines the model is needed over initial states and inputs. In this case the uncertainty will be considered zero. The problem here was related to the comprehension of the model and the selection of the variables for the state. As it was given, the only difficulty was related with the notation in python and alternatives that *numpy* library gives.

3.2 Data association

Once that the robot take the measurements, they should be compared to the known map lines. These features will allow to get the approximation of the jacobian matrix, that will linearize the data 'filtering' the noise, for each of the features detected by the robot. A list of uncertainty for these features and for the measurements will be obtained at the end of this step.
This was the most difficult step, because it was two different ways to get the jacobian matrix, depending in the election of the variables. Once that x, y, z were chosen as the state variables, and f_ρ and f_φ were derived to obtain the jacobian, the problems appeared related with the election of the structures to have the data (arrays, lists, matrices, np.arrays, etc.) and the parameters that could be used with each one in order to combine with others obtaining the desired format. Something remarkable was related with the use of the values of the χ_2 distribution. The confidence interval was set for 90 % was set for 2 degrees of freedom. and that means a value for $\alpha = 4.605$. The precision of the simulation was much better with lower values for α, what it is just the opposite of what should occur. Another thing that affect to the computation of the simulation was the appearance of commented lines between the code. In this case, the delay generated by these lines leaded to a much smooth trajectory over the simulation.

3.3 Update

Once that the estimation of the position and uncertainty of the features was implemented, it was needed the new position and uncertainty for the robot to be updated. Two matrices should be calculated to accomplish this step. First one, S_k, the uncertainty for the features, is done using the hessian matrix (a.k.a jacobian matrix) with the previous covariance matrix, and the uncertainty for the measurements, to get how 'accurate' each of the features is going to be detected by the robot in the new state. Second one is called *Kalman Gain* and allows to estimate the covariance state matrix for next iteration. It is important to say, that the way that was used to calculate the covariance state matrix, avoids losing precision by rounding (because used a 'squared' subtraction) at expense of computation time.//
Problems faced this time where related with the understanding of this new form and with the functions needed to get the expected format for the matrices.

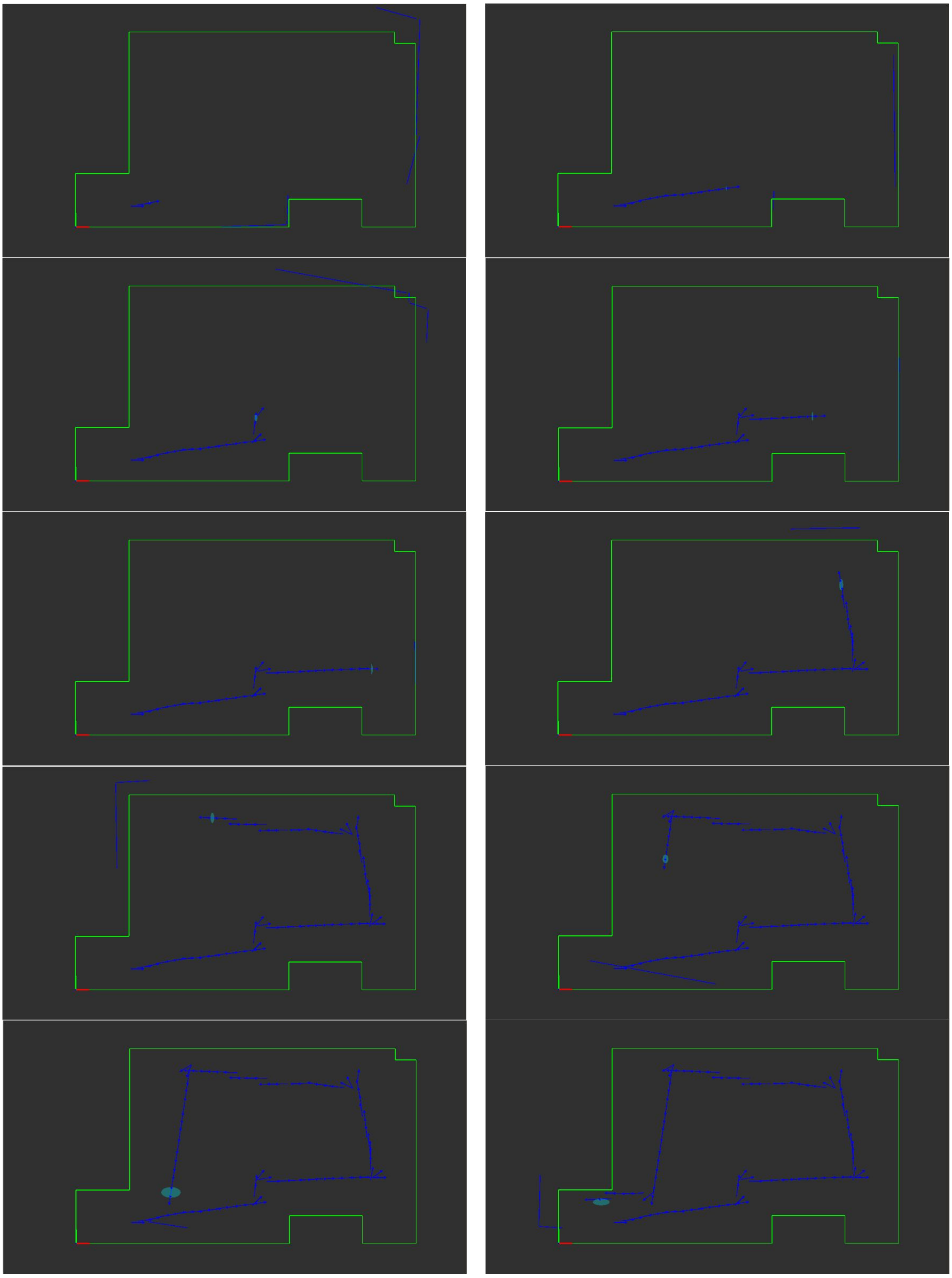

Figure 1: Simulation of evolution of EKF application on robot