Localization lab and homework
Work to be done

Preparation

1. Retrieve the file `StudentDocsLab1.zip` from the Hippocampus Moodle.
2. Create a folder for the lab.
3. Decompress the contents of the zip file in the folder.
4. Read the documents in the following order: the present document, then the "system description" document and then the "programs and data" document. "System description" is fundamental to understand the equations and some parts of the code you have to provide.
5. Using "`ShowOdometry.m`", check the various paths corresponding to each data set. Don't forget to look at the speed data.
6. The files "`EvolutionModel.m`" is missing (you have to write your own). It is called in ShowOdometry, line 48. To write your own version, you have to clearly identify the definitions of X and U in "`ShowOdometry.m`", which are the same as in "`MagnetLoc.m`". Make sure you get the exact same odometry results with your function and the provided one. To make sure it's your version that is run, rename "`ShowOdometry.p`" into something else. **Note**: this task is elementary, but I want to make sure everyone is aware of what the evolution model exactly is.
7. "`MagnetLoc.m`" contains missing code which you need to provide. The missing code is replaced by "***". You can use the localization book to help you in this task.
8. Put all noises and the Mahalanobis threshold to zero in "`DefineVariances.m`" and execute the program. You will get a lot of warnings from Matlab. Ignore them. Use figure 8 and what you know about the sensor construction to evaluate the measurement noise variance. **You will submit a short report for this**, giving details of how you evaluate the variance. It can be handwritten, as long as it is clear. What is important here is to clearly justify the variance estimate. Write and submit the short report as soon as you have determined the variance estimate, so I can give you a feedback.
9. There is another important parameter to be set: the threshold for the Mahalanobis distance (`mahaThreshold`). It is normally set by using the Matlab `chi2inv` function or using chi_square tables. You must understand what this function does and use it to determine a proper value for the threshold. **Submit** to AulaWeb a short report (max ½ page) which explains your understanding of how to determine the threshold, and the chosen value. Write and submit as soon as you have made up your mind, so you can receive a feedback rapidly.
10. Once you have a measurement noise you consider reasonable, set the initial covariance matrix `Pinit`. The standard deviation `sigmaTuning` is now your tuning parameter. If the measurement noise value and initial covariance matrix have been properly set, you should be able to find a proper value for `sigmaWheels`. **Submit** a report that explains the methodology to determine a proper value of the tuning parameter.
11. Starting with a correctly tuned filter, test the effect of over-estimating (resp. under-estimating) each of the following parameters and make sure you understand and can explain what happens. In particular, analyze how the term $CPC^t+Q\gamma$ evolves, and how it helps understand the behavior of the Kalman filter. To over-estimate (resp. under-estimate) a parameter, multiply (resp. divide) its standard deviation by 10.
    - Initial robot position variance.

- Measurement noise variance.
- The same test with `sigmaTuning` should be done and understood during the tuning part of the lab, in question 10.