Information about the documents for the localization lab/homework

All documents are to be retrieved from Hippocampus.

File organization

Create a folder somewhere in your personal space, say "Localization_lab_1". All the files, including the "data" folder should be copied to that folder.

When you start Matlab, make sure to move to the "Localization_lab_1" folder. The programs will look for data files in the "data" sub-folder. If you follow that organization, you will not have to browse through the folders to find the data files.

The programs

- "PlotRawData.m" can be used to see the path, estimated by odometry only, of a given data file. It also shows the speed and rotation speed, as estimated using encoder data.
- "`MagnetLoc.m`" is the main program. It implements the Kalman Filter localization algorithm, which is applied to one of the data files provided for the lab (cf. infra).
  Type "`help MagnetLoc`" for help. Beware, the program is not executable until you supply the missing code and fill the informations in "`DefineVariances.m`", explained below.
- "`PlotResults.m`" is used to plot the results of the Kalman filter. The description of the various figures is given below. It can only be invoked after "`MagnetLoc.m`" has been executed.
- "`RobotAndSensorDefinition.m`" is a script which defines the characteristics of the robot, of the sensor, and the location of the magnets. You don't need to go into the details of it because the necessary information is pre-defined, as you do not have the robot, but you can have a look if you wish. Some comments are given. Do not modify this file.
- "`DefineVariances.m`" is a script which defines the various noise covariance matrices. Some parameters have been replaced by "`***`". Your task is to set them to proper values for the Kalman Filter to operate in a satisfactory way. All these parameters have also a comment "% Determined by student" so you always know which they are. **Do not modify anything else in this file**.

Other programs are called by the main program. Using "`help`" gives a basic description. Most of them are not fundamental to the algorithm, mainly dealing with initialization of the data and storage of results for display. Those not directly related to the Kalman Filter are not commented.

Tip: use the following line of instructions each time you execute and recall it every time you need, using the "up" key. It is the fastest way.

```
clear all; close all; MagnetLoc; PlotResults;
```

If you want to compare plots between two executions, remove the "close all" instruction.

The data files

They contain real data recorded by students during the project in the framework of which the system was developped. The initial posture indicated for each file is of course approximate, the robot being positioned by hand on the table.

The various data files are located in the "`data`" folder. They are:

- "`circles.txt`": The robot performs several circles. Due to wheel slippage, the circles slowly drift. Posture `(0,0,0)` is a correct initialization for this path.
- "`diagonal45degrees.txt`": The robot moves along a straight line diagonally. The posture `(0,0,pi/4)` is a correct initialization for this path.
- "`line1magnet.txt`": The robot moves along the line y=0. The robot goes over the magnets at positions `(55*`k`,0)`. Posture `(0,0,0)` is a correct initialization for this path.
- "`line2magnets.txt`": The robot moves along the line y=27. The robot goes over the magnets at positions `(55*`k`,0)` and `(55*`k`,55)`. Posture `(0,27,0)` is a correct initialization for this path.
- "`oneloop.txt`": The robot starts from posture `(0,0,0)` and performs a trajectory that takes it back to `(0,0)` but with a different orientation. The path consists of a single loop.
- "`twoloops.txt`": The robot starts from posture `(0,0,0)` and performs a trajectory that takes it back to `(0,0)` but with a different orientation. The path consists of a two loops and is longer than the previous one.

You can visualize the paths using the programme "`ShowOdometry.m`". As the name suggests, it uses odometry only and can be executed completely independently of any Kalman Filter tuning. The program also shows the speed and rotation speed of the robot as functions of time.

The two "looped" data files play a major role in determining whether your localization system works. Indeed, they are the only files for which the end position is know: the operator was instructed to take the robot back to (0,0), with orientation not imposed. Of course, it is not easy to do that accurately, but the operator did take the robot back to the origin the best he could.
In the "twoLoops" data file, there is an interval of three seconds where no magnets are detected. Some problem happened, but I decided to keep the data file as is, because it is interesting to see what happens when there is no exteroceptive data during a certain amount of time.

The outputs of PlotResults

For a description of each figure generated by PlotResult, type: **`help PlotResults`**.

`PlotResults` also writes the following information:

- Traveled distance in mm as determined by odometry (sum of all ΔDs).
- Odometry error, *i.e.* the distance between the odometry estimated final position and the Kalman filter estimated position, divided by the traveled distance, in percent.
- The percentage of cases when the Mahalanobis distance calculated for the closest magnet was above the threshold, thus being rejected (no update performed).
- The percentage of cases when the Mahalanobis distance calculated with a neighbor magnet (not the closest) was below the threshold.