Information about the documents for the localization lab/homework

All documents are to be retrieved from the "Serveur pédagogique"

The programs

- "`ShowOdometry.m`" can be used to see the path, estimated by odometry, of a given data file. Type "`help ShowOdometry`" for instructions on how to use.
- "`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 Some parameters have been replaced by "`***`".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.

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. Usage is: `ShowOdometry('circles.txt')`
Or any other file name you wish to check. The command shows the path calculated by odometry and the speed and rotation speed of the robot as functions of time.

The outputs of PlotResults

- **Figure 1: paths.**
  Red curve: odometry-estimated path.
  Bold blue curve: Kalman filter estimated path.
  Black dots: real positions of the magnets.
  Green crosses: estimated positions of the magnets. That's $\hat{Y}$ in the Kalman filter.
- **Figure 2: speeds.**
  Estimated speed and rotation speed of the robot using odometry. They are simply $\Delta D/T$ and $\Delta\theta/T$, where T is the sampling period (here 50 ms). No filtering is applied.
- **Figure 3: variances.**
  Estimated error variances for x, y and θ. The vertical green dotted lines indicate the time instants when an update has been performed using a magnet sensor measurement.
- **Figure 4: Mahalanobis distances.**
  Each time a magnet is detected by the sensor, its position in the robot frame is in fact measured. Knowing the robot estimated posture, the position of the magnet in the absolute reference frame is estimated. The real magnet position is the position of the closest real magnet. It is the most likely magnet that was actually detected. The Mahalanobis distance is calculated for this magnet and reported as a green dot. The red dots at the same time instants are the Mahalanobis distances calculated with the nearest neighbors of this magnet. In the perfect cas, all green dots should be below the horizontal blue line (the acceptance threshold), and all the red dots above the threshold.
- **Figure 5: posture evolution.**
  Evolution of x, y and θ as a function of time.
- **Figure 6: number of measurements per period.**
  The graph shows how many magnets are detected at each step. You can see that, sometimes the sensor detects two magnets, one at each end of the linear sensor.
- **Figure 7: raw sensor data.**
  Shows the state of each elementary magnet detector (there are 8 of them) that constitute the sensor. A vertical blue bar indicates that the corresponding detector is on (detects a magnetic field). This graph will be useful in determining the measurement noise. Later, you can actually comment it out in the file "`PlotResult.m`" if you wish.

`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.