Albert-Ludwigs-Universität Freiburg, Institut für Informatik
Prof. Wolfram Burgard, Dr. Daniel Büscher, Alexander Schaefer
Lecture: Robot Mapping
Winter term 2017/2018

Sheet 8

Topic: FastSLAM

Due: December 21, 2017

Exercise: FastSLAM Implementation

Implement the basic FastSLAM 1.0 algorithm as presented in the lecture. Assume known correspondences and use a simple feature storage approach, i.e. do **not** use the tree data structure that yields an $\mathcal{O}(N \log M)$ time complexity.

To support this task, we provide a small Octave framework on the see course website. The framework contains the following folders:

data contains files that represent the world definition and sensor readings.

octave contains the FastSLAM framework with stubs to complete.

plots stores the generated images.

The task described below should be implemented inside the framework in the directory `octave` by completing the stubs:

- Implement the correction step in `correction_step.m`. For the noise in the sensor model, assume that Q_t is the diagonal 2×2 matrix

$$Q_t = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}.$$

After implementing the missing part, you can run the FastSLAM system. To do that, change into the directory `octave` and launch Octave. Type `fastslam` to start the main loop. The script will produce plots of the state of the FastSLAM algorithm and save them in the `plots` directory. You can use the images for debugging and to generate an animation. For example, you can use ffmpeg from inside the plots directory as follows:

```
ffmpeg -r 10 -i fastslam_%03d.png -b 500000 fastslam.mp4
```

Some implementation tips:

- Turn off the visualization to speed up the computation by commenting out the line `plot_state(...)` in the file `fastslam.m`.

- While debugging, run the filter only for a few steps by replacing the for loop in `fastslam.m` by something along the lines of `for t = 1:50`.
- When converting implementations containing for loops into a vectorized form, it often helps to draw the dimensions of the data involved on a sheet of paper.
- Many of the functions in Octave can handle matrices and compute values along the rows or columns of a matrix. Some useful functions that support this are `sum`, `cumsum`, `sqrt`, `sin`, and `cos`.