Albert-Ludwigs-Universität Freiburg, Institut für Informatik
Dr. Gian Diego Tipaldi, Dr. Luciano Spinello, Prof. Wolfram Burgard
Lecture: Robot Mapping
Winter term 2014

Sheet 6

Topic: Unscented Kalman Fitler SLAM

Submission deadline: Dec. 4

Submit to: `robotmappingtutors@informatik.uni-freiburg.de`

Exercise: UKF SLAM

Implement an unscented Kalman filter SLAM (UKF SLAM) system. You should complete the following parts:

(a) Implement the prediction step of the filter by completing the function in `prediction_step.m` to compute the mean and covariance after incorporating the odometry motion command.

(b) Implement the correction step in `correction_step.m` to update the belief after each sensor measurement according to the UKF SLAM algorithm.

To support this task, we provide a small *Octave* framework (see course website). The above-mentioned tasks should be implemented inside the framework in the directory `octave` by completing the stubs. After implementing the missing parts, you can test your solution by running the script in `ukf_slam.m`. The program will produce plots of the robot pose and map estimates and save them in the `plots` directory. Figure 1 depicts some example images of the state of the UKF.

Note that, as opposed to the EKF SLAM system you implemented in sheet 4, here the state vector and covariance matrix are incrementally grown with each newly-observed landmark. The mean estimates of the landmark poses are stacked in μ_t in the order by which they were observed by the robot, as described in the *map* vector in the framework.

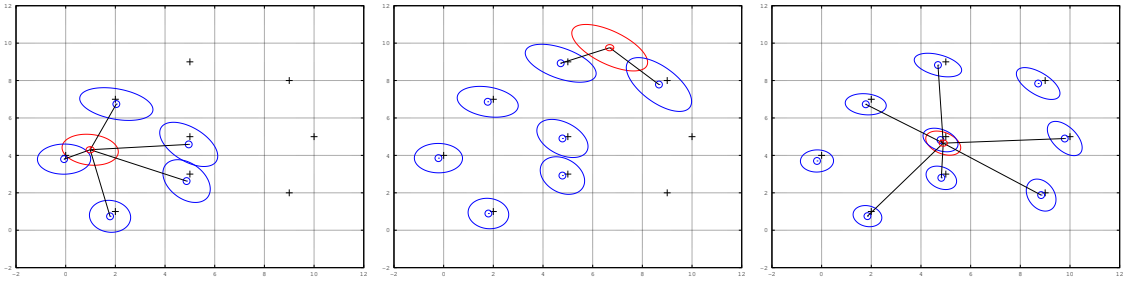
Some implementation tips:

- Be careful when averaging angles. One way to average a set of angles $\{\theta_1, \dots, \theta_N\}$ given their weights $\{w_1, \dots, w_N\}$ is to first compute the weighted sum of the unit-vectors of each rotation

$$\bar{x} = \sum_{i=1}^{N} w_i \cos(\theta_i), \qquad \bar{y} = \sum_{i=1}^{N} w_i \sin(\theta_i).$$

  Then the average angle $\bar{\theta}$ is given by

$$\bar{\theta} = \operatorname{atan2}(\bar{y}, \bar{x}).$$



$t = 50$ $t = 150$ $t = 330$

Figure 1: Example images of the state of the UKF at certain time indices.

- Remember to normalize the angles after you computed a sum or a difference involving angles.
- Turn off the visualization to speed up the computation by commenting out the line `plot_state(...)` in the file `ukf_slam.m`.
- While debugging, run the filter only for a few steps by replacing the for-loop in `ukf_slam.m` by something along the lines of `for t = 1:50`.
- The command `repmat` allows you to replicate a given matrix in many different ways and is magnitudes faster than using for-loops.
- 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`, `sqrt`, `sin`, `cos`, and many others.