ECE 4984 & 5984: (Advanced) Robot Motion Planning (Spring 2017)

Homework 6

Due: Sunday April 30th, 9PM

April 20, 2017

Instructions. This homework constitutes 8% of the course grade. You must work individually on all problems. It is okay to discuss the problems with other students in class. However, your submission must be your original work. This implies:

- If you discuss the problems with other students, you should write down their names in the pdf report.
- If you refer to any other source (textbook, websites, etc.), please cite them in the report at the relevant places.
- The answers in the pdf report must be written entirely by you from scratch. No verbatim copy-paste allowed without citations.
- Any software you submit must be written entirely by you and your partner with no copy-pasting of significant portions of code from other sources.

Please follow the submission instructions posted on canvas exactly. You must submit your assignment online on canvas by the due date. Your submission must include one pdf file with the answers to all the problems and one or more files containing your code. It is okay to scan your answers and create the pdf submission.

Problem 1 — 100 points

Read the paper [1] which is also uploaded on canvas. The paper gives a good, albeit limited, overview of some of the motion planning algorithms that we have covered in class. The paper focuses on how to select and how to combine motion planning algorithms based on a number of criteria. It should be an easy read and hopefully gives you a chance to revisit some of the topics we have covered but with the bigger picture in mind. This is also good preparation for the test. Nevertheless, the paper has its own limitations and misses out on some of the important considerations in designing and applying motion planning algorithms. Your assignment is to write a critique ($\sim$ 2 pages) of the paper and answer the following questions. Your critique must consist of three parts:

- Summary of the paper **in your own words**. The summary must clearly state the motivation of the authors in writing the paper, the main contributions, and what you think are its strengths and limitations.
- On page 108, the authors list seven requirements of a motion planning algorithm. Consider the five classes of motion planning algorithms we have covered in class: *Bug algorithms, dynamic programming, A* and its variants, RRT and its variants, and MDPs/POMDPs/belief MDPs.* For each family, write down whether it meets (completely or partially) all the seven requirements. Justify your answers (*e.g., A* is complete for a given graph. However, if the graph is obtained by discretizing the environment,*

then A^* may not be complete unless the discretization is fine enough. Some requirement may not be applicable for a particular algorithm or may be applicable only to a specific variant, in which case you can state it so.

- Consider the decision flowchart given in Figures 1, 2, and 3. Are the flowcharts complete? If not, what are they missing? (*Hint: Think about the five classes of algorithms and how each one fits in the flowchart. If there is an algorithm which doesn't fit in the flowchart, then this implies that we need to modify the flowchart.*).

Bonus Problem — 25 points

Part A — 5 points

Implement GPS-based localization for a 1D robot using a Kalman Filter. We will use a discrete time formulation, where consecutive timesteps are separated by dt seconds. Let x_k denote the position of the robot at timestep k. The state transition equation is given as,

$$x_k = x_{k-1} + u_k \cdot dt + \epsilon_k,$$

where ϵ_k is the Gaussian process noise with zero mean and σ_u standard deviation. At each time step, the robot obtains a noisy GPS measurement of its true position,

$$z_k = x_k + \delta_k,$$

where δ_k is zero mean Gaussian noise with σ_g standard deviation.

Implement a Kalman Filter for estimating the robot's state in MATLAB. Starter code is provided in the file `starter_oneD_KF.m`. You will have to implement the following two functions:

1. `KF_propagate()` that implements the Kalman filter propagation step and returns the propagated state and covariance estimates.
2. `GPS_update()` that implements the Kalman filter update step and returns the updated state and covariance estimates.

In addition to the usual propagate-update cycle, we will also keep track of the robot's estimate based purely on the odometry. This will be done by only propagating the robot's state, but with no GPS updates. The starter file shows how to do so.

The starter file also contains code to generate three figures:

- the true trajectory of the robot along with the KF estimates and the odometry-only estimates;
- error in the odometry estimate along with the corresponding 3σ variance bounds (determined from the propagated variance); and
- error in the KF estimate along with the corresponding 3σ variance bounds (determined from the propagated/updated variance).

Submission Instructions

1. In the pdf report, write the equations for the Kalman filter mean and covariance propagation and updates (not the general equations in the slide, but with correct substitutions for $A_k, B_k, etc.$ for our formulation).
2. The MATLAB files `KF_propagate.m` and `GPS_update.m` that implement the corresponding functions.
3. In the pdf report, include the three figures that are generated by the starter code.

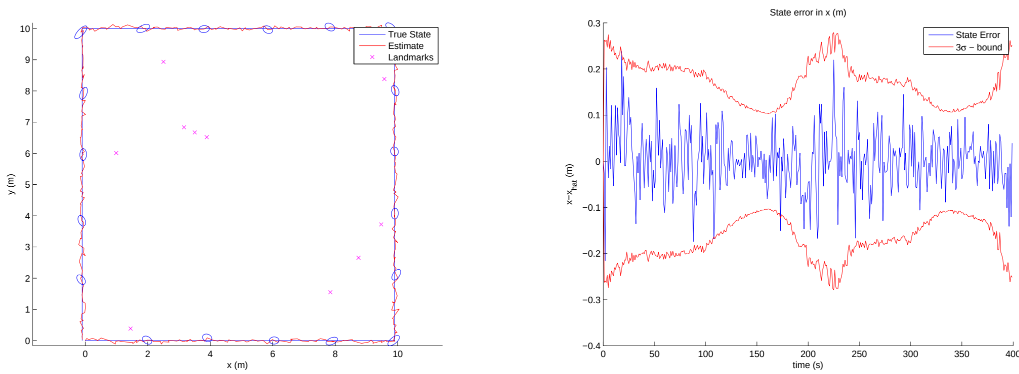

Figure 1: (a) Estimated and true trajectory along with the error ellipse. (b) Error in the x-coordinates. Your code must generate similar figures for error in y coordinates.

Part B **20 points**

Implement a landmark-based localization for a 2D robot using an Extended Kalman Filter. Let $X_k = [x_k, y_k]^T$ denote the state of the robot at timestep k. The state transition equation is given by,

$$X_k = X_{k-1} + \begin{bmatrix} u_k^x \\ u_k^y \end{bmatrix} \cdot dt + \epsilon, \tag{1}$$

where u_k^x and u_k^y denote the control velocities along x and y axes. The process noise ϵ is given by a zero mean Gaussian with covariance $R_k = \begin{bmatrix} \sigma_u^2 & 0 \\ 0 & \sigma_u^2 \end{bmatrix}$. Note that this is still a linear system.

Unlike part A, where we had direct GPS measurements of the state, here we will only use range measurements to n_{lm} landmarks placed in the environment. Let $m = [m_x, m_y]^T$ be one of the landmarks. The measurement with respect to m is given by,

$$z_r = \sqrt{(x(k) - m_x)^2 + (y(k) - m_y)^2} + \delta,$$

where $[x(k), y(k)]^T$ is the true position of the robot. This is a non-linear measurement equation and δ is zero mean Gaussian noise with standard deviation equal to σ_r. In order to use the EKF, you will have to linearize this using the Taylor series expansion.

The starter code is given in `starter_twoD_EKF.m`. In the starter code, we randomly generate n_{lm} landmarks. You will have to implement the following:

1. `rws_2D()` that applies the true control values to generate the following: (i) true coordinates of the states; (ii) noisy control values drawn from a Gaussian distribution with mean equal to the true control values, and covariance R; and (iii) noisy range measurements between the true robot's state and each of the landmark, for each timestep. The noisy measurements must be drawn from a Gaussian distribution with mean equal to the true range and standard deviation equal to σ_r. Use `rws.m` provided for the previous problem as reference.

2. `KF_propagate()` that implements the Kalman filter propagation step and returns the propagated state and covariance estimates.

3. `EKF_update_range()` that implements the Kalman filter update step and returns the updated state and covariance estimates.

4. Plot the error in the true and estimated robot position. Use the starter code for part A as reference.

Submission Instructions

1. In the pdf report, write the equations for the EKF mean and covariance propagation and updates (not the general equations in the slide, but with correct substitutions for $A_k, B_k, etc.$ for our formulation). In particular, write the linearized measurement equation.

2. The two MATLAB files `EKF_propagate.m` and `EKF_update_range.m` that implement the corresponding functions.

3. In the pdf report, include the figures that are generated by your code (similar to Figure 1).

References

[1] Janno Johan Maria Lunenburg, Sebastiaan Antonius Maria Coenen, GJL Naus, Marinus Jacobus Gerardus van de Molengraft, and Maarten Steinbuch. Motion planning for mobile robots: A method for the selection of a combination of motion-planning algorithms. *IEEE Robotics & Automation Magazine*, 23(4):107–117, 2016.