

## MTRN4010 – Project #3 (Preliminary release)


### Applying EKF for map-based robot localization

Project 3 involves applying the EKF, for solving a robot localization problem, based on maps. We intend to use it in real-time, in the same way you will need in a real industrial/research implementation. For achieving it, we solve this problem incrementally, so we are able to secure that its many parts are working well, before we try the full solution under real-time conditions.

The first part of the project asks the student to adapt the EKF to solve the map-based localization, in a purely synthetic simulation context. The second part aims to use it dealing with real data, but still in an off-line fashion. The last part of the project asks the student to finally try it under real-time constraints.

- 1) Modify the example “DemoEKF\_2018.m” for
  - a) Processing **bearing observations** (in addition to the range already implemented in the example.)
  - b) Using a proper Q matrix, based on the assumed noise in inputs of the process model.

Note: for solving item (a) you need to also **modify** certain parts of the **simulator components**.

- 2) Simulate the existence of bias in the gyroscope’s measurements.  
Extend the EKF solution in (1) for estimating the gyroscope’s bias, as explained in lecture notes “[AAS2018]EKF\_Localizer\_estimating\_bias\_v02.pdf”   
Test your implementation simulating cases having a **bias -1 degree/second and +1 degree/second**.

- 3) Use (1) for implementing the EKF based on Project2.Part4.

This program is based on the data used in Project 2. It is OFF-LINE but using real data, in the same way we solved Project2.

You are requested to adapt your solution for Project2.Part4, adding the EKF component. The estimates should be more accurate than those obtained in P2.4.

Assume the following realistic conditions:

Noise in angular rate measurements:	standard deviation = 1.5 degrees/second.
Noise in speed sensor:	standard deviation = 0.05m/s.
Noise in range measurements:	standard deviation = 0.15m.
Noise in bearing measurements:	standard deviation = 1 degree

Note: you must remove the bias which is present in the angular rate measurements.

- 4) Include the capability implemented in (2) in the solution developed in (3).  
Consider that the gyroscope’s bias can be, in the worst case, limited in the **range [-2,+2] degree/second**.

Note: you must not remove the bias which is present in the angular rate measurements; because your estimation process estimates and remove it on the fly.

- 5) Implement (3) to operate in real-time, based on the UDP approach.

This program is intended to work ON-LINE (“soft real-time”). This program implies using parts of the solution already developed in items 1 to 4 and previous projects. Your program will receive the data via UDP, as in the previously provided example about using UDP. You will use the simulator via UDP, for testing this solution. The dataset which is provided for the playback session is the same data you used in previous projects but is being fed by the simulator in a real time fashion. Consequently, you will assume the same noise characteristics.

6) Implement (4) to operate in real-time, based on the UDP data context. As in (5), reusing and integrating previous solutions.

7) Modify (6) for estimating longitudinal velocity. This means you need to estimate, in addition to the platform’s 2D pose, the gyroscope’s bias and the longitudinal component of the velocity.

Recommendation: you may solve it first modifying your solutions in pure simulation, for verifying performance; before trying in real-time. Assume that the **maximum acceleration** which the platform can achieve is **1.5m/s<sup>2</sup>**.

### Showing results

For item (1), you will plot the result at the end of the process. The style of the plots may be as the ones produced by the provided example program (“DemoEKF\_2018.m”).

For item (2), you will include plots as in (1), and, in addition, the plot of the estimated bias (its expected value). You will also indicate the real bias (which you simulated to be polluting the measured angular rates, in that test). The style of plot should be as the one presented in the lecture notes (file [AAS2018]EKF\_Localizer\_estimating\_bias\_v02.pdf), in which an example of performance is shown.

For (3) you will show results in the same way you did in Project2.Part3/4, using the same graphics resources you implemented in that project.

You may (this is not mandatory) simultaneously run the pure dead-reckoning solution, for visualizing the better performance of the EKF solutions. The lecturer showed a solution, in class, in which both estimation processes were run simultaneously and their estimated compared.

For visualizing that the pose estimates are consistent with the real ones, we will infer it by inspecting the expected global position of the detected OOIs. Those will appear close enough to the map landmarks. This visualization is the same you had implemented for solving Project2.part3/4.

For (4) you will simply reuse the visualization resources used in item (3). In addition, you will print/show, at low frequency (e.g. @1Hz), the currently estimated gyroscope’s bias. You may simply print it as text in the console; or show it as text in some of the graphic visualizations you have.

For items (5) and (6), you simply show the results in the way you do in items (3) and (4).

For item (7) you will show the estimated speed (its expected value) and the real (but unused by the EKF) one. The style used for showing that information is up to you.

Units for reporting/showing variables: For positions and distances: use meters. For angular variables: degrees. For rates: Use units derived from the units of the derived variables “per second” (e.g. speed will be expressed in meters/second.)

### **Alternative programming framework**

Parts 5, 6, 7, can be solved in any context / programming framework. We strongly recommend using Matlab; however, you are free to use an alternative programming framework, if you feel better in using it. E.g. Python, C/C++ or even under ROS. In such a case, you will give your demonstration to the lecturer in place of the standard way being evaluated by tutors. There are no bonus marks for using an alternative programming framework; this option is offered in case you feel more confident implementing the solutions in that way.

### **Relevance of project items**

From parts 1 to 7, in percentages of the project marks, [(1):10%, (2):10%, (3):13%, (4):15%, (5):20%, (6):16%, (7):16%]

### **Submission details**

Deadline: Week 10, during your nominal lab session.

Submission of files: You will submit your solutions immediately after your demonstration. The specifications about how the files have to be submitted will be given via Moodle; it will be similar to the way used in previous submissions.

Late demonstrations/submissions: We will apply the penalties according to the course outline.

### **Quiz**

There will be a quiz which will take place during the lecture time. We will secure a proper classroom for that purpose. (so, the quiz will not take place during the lab time, except we cannot secure a proper classroom). In this way all the students will be exposed to the same conditions, at the same time. The date (week) for the quiz will be informed before week 10.

The marks for project 3 will be affected by your result in the quiz, according to a formula which will be given in week 9.

The questions included in the quiz are fully related to concepts which you apply for solving the project.

If you expect to have a clash with other courses, or you have any other reason for not being able to attend that lecture time, you need to contact the lecturer in advance, before that lecture. An alternative date will be offered for those justified cases.

NOTE: This is a preliminary release. Final version of this project will be released on week 8. The final version will contain the same required work but will provide additional details.

Questions: Ask the lecturer, via Moodle or via email ([j.guivant@unsw.edu.au](mailto:j.guivant@unsw.edu.au))