MTRN4010.2024 PROJECT #2

Sensor Data Fusion. EKF state estimation and parameter identification.

Introduction

Project 2 is focused on implementing Sensor Data Fusion, in a Bayesian fashion, via applying the EKF approach. The estimation process is used to estimate the state of a system and simultaneously perform parameter identification so that certain model parameters that are not well known can be tuned in real-time.
In this project, we will explore EKF applying it to two different cases: the same vehicle as in project 1 and an actuated pendulum.

Case A: Vehicle Localization

We continue working on our case of study: "localizing a platform". We have seen in Project 1 that there are diverse sources of information, i.e., those provided by sensors' measurements and from a process model such as the kinematic model of the platform. In Part E of the project 1, you have explored how to estimate the state using state observer, including the correction based on the difference in observation. A problem arises from that method: What if one source of information is more trustworthy than the other? The previous method assumed that information from different sources was equally trustworthy. Is there a way to model the "trustworthy-ness" from all sources?
In addition, that approach was not able to exploit information when insufficient useful OOIs were available (e.g. just one). Finally, how can we make our estimation approach to learn certain model parameters, so that our predictions, during periods of absence of OOIs, are more accurate?

In this part, we combine those sources of information in a statistical way, in which the statistical descriptions of those sources of information are considered for generating estimates of the variables of interest (the state of the system). Those estimates are required to be permanently produced. The estimation process will generate optimal estimates in the form of an expected value and a covariance matrix. The estimation process will operate in real-time.

Part A1 is focused on proposing and implementing an EKF-based localizer, which exploits the platform's kinematic model, measurements from sensors, and a map of surveyed landmarks (the same sources of information you used in project 1).

For Part-A1 you may modify your solution for Project1. Your program will maintain estimates of the vehicle pose, i.e., an expected value and an associated covariance matrix. You will perform the EKF prediction step in the same way you used to run your kinematic model, in Project1. In addition, you will perform update steps at each time at which observations are available, i.e., when *useful OOIs* are detected. The update step will exploit the calculated distances to those useful OOIs, pretending we are using a **SONAR**, which only tells you about the **distance** between the sensor and the OOI. For solving this part, you need to use modules B and C of your Project 1. The data association (DA) module will run as in Project 1; however, in this case, the DA will be based on the prior expected value of the vehicle pose, before the EKF update is applied.

For simulating noisy conditions, the menu offered by the API (version 03) does include edit boxes in which you can specify the characteristics of the noises which affect measurements from some sensors. Those noises are:

1) Bias affecting gyroscope's measurements (in °/second).
2) Standard deviation of WGN that affects gyroscope measurements (in °/second).
3) Standard deviation of WGN that affects speed measurements (in m/s).
4) Speed sensor gain.
5) Amplitude of zero-mean white uniform noise that pollutes LiDAR scans.

You do not need to explicitly add any of those noises to the measurements (as you used to do in part of project1), because now it is done by the API itself, so that it is transparent to your program.
In this part of the project, you are required to implement the EKF localizer, exploiting only the **OOIs' range observations**.

Assume the following characteristics for the uncertainties and noises that pollute the sensors' measurements:

WGN in Gyroscope measurements	:	standard deviation: 3 degree/second.
Bias in Gyroscope measurements	:	0.01 degree/second (almost no bias).
WGN in speed measurements	:	standard deviation: 5 cm/second.
Speed sensor gain	:	= 1 (nominal value, well known).
LiDAR range measurements	:	white uniform noise, amplitude: 15 cm.

Part A2. It requires you to describe the EKF update associated with the detection of a *useful OOI*. We assume that the 2D position of the useful OOI is measured and expressed in LiDAR1's coordinate frame, in cartesian representation. Your proposal must detail the necessary equations for implementing an EKF update step adequate for processing such 2D measurement.
Your proposal must assume the following:

1) Measured output variables (used in the EKF update step): 2D position of a useful OOI (OOIx, OOIy) in LiDAR1's cartesian coordinate frame. It is specified, in this item, that **we are not allowed** to convert those variables to other representations or other coordinate frames before being used in the EKF update. They must be processed in their original representation, consequently the observation models (aka output equations) must correspond to those mentioned physical variables.
2) We assume that the measurements of OOIx and OOIy are polluted by uncertainty (noises) which behave as WGN, of standard deviation equal to 20cm for both measurements.
3) The states being estimated: 2D pose of the platform (x, y, heading), in the Global coordinate frame.
4) We assume that only one useful OOIs is available for the EKF update being described.

Note: We are asking you to consider the case in which the EKF update does process only one OOI, for the sake of simplicity.

You are required to discuss and explain the process in a **report**, including the output equation, the associated analytical expression and the derivation of the H matrix, and a proper R matrix. The proposal must be submitted in a document with the filename ***partA2.pdf***.

Note: If you implemented this approach, you would see that its performance in terms of accuracy is superior to that applied in Part A1. Adapting part A1 to work as described in part A3 is not difficult, but for the sake of keeping this project short, we do not require it.

Part A3. It requires the proposal and implementation of an extension of the estimator previously defined in part **A1**. You are required to modify the EKF process to include the estimation of a **constant parameter** which is not, a priori, accurately known. That parameter is the gyroscope bias, ***b***. It will be assumed that that unknown bias may have any value in the range from -1 to +1 degrees/second. You are required to propose adequate adaptations for making the localizer able to estimate the platform's pose and the required parameter, simultaneously, in real-time. We may assume that the platform's conditions of operation are those of part A1.
In addition to estimate the mentioned variables, your program is required to record, at every LiDAR event,

1. Expected value of the pose and bias.
2. Diagonal elements of the covariance Matrix of the pose and bias estimates.

Those recorded values will be used to verify the performance of the implementation.

Part A is worth 13 marks (of the 25 marks of Project 2).
Relevance of subparts in part A: A1:33%, A2=33%, A3:34% (of Part A)

Marking criterion for Part A1:

Correct implementation of necessary equations:	40% (of Part A1)
Satisfies accuracy specifications.	30%
Satisfies consistency specifications.	30%

Analysis of statistical consistency.
At the end of the trip, we show the recorded discrepancies between expected values of the state vector and the corresponding ground truth values, during the full trip. Those errors should be consistent with the marginal variances reported by the estimator.

Your consistency plots must indicate that most of the time the discrepancy (between ground truth and expected value), for each component of the pose vector, is lower than twice the associated standard deviation at that time. This specification must be satisfied during at least 70% of the trip. Read the document "ConsistencyPlots_Project2.pdf", to know how to produce your *consistency plots*.

Accuracy: The maximum discrepancy (|actual value – expected value|) must satisfy that:
For x and y, the maximum absolute error is <0.4m, always.
For heading the maximum absolute error is <3 degrees, always.

The "duration of trip" is the interval of time starting at the time at which the platform begins to move, i.e., excluding any initial period during which the platform was not moving; however, any subsequent stopping periods are not excluded.

Consistency and accuracy will be marked only if *consistency plots* are produced at the end of the trip.
If no consistency plots are produced, no marks will be given in these marking items.

Marking criteria for Part A2.

Your description will include the following necessary items.

1)	Observation function	(40% of A2, if correct, 0% if incorrect)
2)	H matrix (analytical expression)	(40% of A2, if correct, 0% if incorrect)
3)	R matrix	(20% of A2, if correct, 0% if incorrect)

Marking criteria for Part A3.
Part A3 will be tested in conditions that are equal to those of Part A1, except for the gyroscope bias, that was not present in that part. Your program must work properly under any value we may set for that polluting bias (in the range from -1 to 1 degrees/second).
The specifications for part A3 are the same that are used for part A1, in respect to the accuracy and consistency of the estimates of x, y and the heading. In respect to the estimated bias, its expected value must clearly converge to the actual bias, having a final **discrepancy lower than 0.2 degrees/second**. All these specifications must be satisfied.

Correct implementation of necessary equations:	40% (of Part A3)
Satisfies accuracy specifications.	30%

Satisfies consistency specifications. 30%

Case B. State estimation and parameter tuning.
You are required to implement a state estimator based on the EKF approach, for estimating the state of a system, an actuated pendulum, whose model is expressed by the following ODE.

$$\ddot{\varphi}(t) = -a \cdot \sin\left(\varphi(t)\right) - b \cdot \dot{\varphi}(t) + c \cdot u(t)$$

In which the angular position, velocity and acceleration are expressed in radians, rad/s and rad/s^2. The rest of the model variables and parameters are assumed that are expressed in proper engineering units.

The system does operate under the following conditions:

- We accurately know that the parameters are $a = 8, \quad b = 1.4, \quad c = 1.$
- We obtain a good discrete time model by considering a regular sampling interval of 1ms ($\tau = 1\ ms$).
- We have a sensor that measures the angular rate, every *10ms*.
- The measurements of the angular rate are polluted by WGN of standard deviation = 5 °/s.
- No other disturbances do affect those measurements.
- We assume we do not know the initial angular position and angular velocity, so that we assume initial expected value [0;0]. However, we do know that the actual initial angular position may be between -110° to +110°, and that the initial velocity is close to 0°/s (between -2 °/s and 2 °/s)
- The input, ***u(t)***, is a square wave signal, that switches between +5 and -5, every 1.5 seconds, however our estimator does not know that, and simply relies on measurements of ***u(t)***, which are polluted by WGN of standard deviation 0.1 (in the same units used by variable ***u(t)***).

B1) You are required to implement a **simulation program**, for performing a simulation of duration T=18 seconds. In that simulation you will simulate the actual plant and, concurrently, run the EKF state estimator, which will estimate the plant's state.
You will simulate sensor measurements of certain variables by using the actual values of those variables in the simulated system, and adding to them samples of WGN properly scaled. A way to do it, in general, is as follows:

*noise = randn(1,1)*StdNoise; % Sample of WGN, having standard deviation StdNoise*
simulated_measurement_of_variable = actual_value_of_variable +noise;.

Those versions of the actual variables will be properly fed, by you, to the EKF estimator, as if they were real noisy measurements from sensors.
You will consider the statistical characteristics of the simulated noises (e.g. their variances), for adequately tuning the relevant parameters of the EKF (e.g. Q, R, etc.).

B2) You will simulate a more realistic case. You will consider the case in which you do not accurately know parameter ***a***. For that reason, your estimator will have the additional task of *tuning* the parameter **a** used in your model, simultaneously estimating the plant's state. You will simulate the plant as you did in Part B1, but you will have a initial mismatch between the nominal and the actual value of the parameter ***a***. Your nominal value will be up to 15% away from the actual value. That nominal value will be used in the initialization of the estimation process.
The rest of the conditions are equal to those of part B1. You will show that your estimates of ***a*** do converge to the actual value of parameter ***a*** (which is unknown to the estimator, but it is known to us).
You will record the expected values of $(\varphi, \dot{\varphi}, \mathrm{a})$ at each discrete time (every millisecond). You will also record the variances of all the scalar marginal PDFs, to show (by plotting) the discrepancies at the end of the simulation,

and to show that those discrepancies between expected values and ground truth values, are consistent with reported variances (as you did in B1).

Your results must clearly show that:

1) The estimate of the position does converge to the actual angular position, so that the discrepancies of expected angular position and actual angular position are always less than 0.6 degree after t>6 seconds.
2) The discrepancies of expected angular position and actual angular position, is lower than 3 standard deviations, being the standard deviation calculated from the associated marginal variance.
3) The expected value of parameter ***a*** does converge to the real value of ***a*** (expected value of ***a*** is always less than 2% away of the actual value of ***a***, after t>4 seconds.

Part B is worth 12 marks (of the 25 marks of Project 2).
Part B1 is worth 50% of part B. Part B2 is worth 50% of part B.

Marking criteria for subitem B1,

Correct implementation of necessary equations:	40% (of Part B1)
Satisfies accuracy specifications.	30%
Satisfies consistency specifications.	30%

Marking criteria for subitem B2,

Correct implementation of necessary equations:	40% (of Part B2)
Satisfies accuracy specifications.	30%
Satisfies consistency specifications.	30%

Questions about this project: ask the lecturer via Moodle or by email (j.guivant@unsw.edu.au)

The lecturer will show a possible solution working, during lecture time, on week 7.

Submission of project 2: **Friday, week 9, 23:55 + 2 days** (**Sunday**, 23:55). This means that you have three weeks to complete this project.
Late submissions: Work submitted late without an approved extension by the course coordinator is subject to a late penalty of **18%** (of the maximum mark possible for this assessment) per calendar day. The late penalty is applied per calendar day (including weekends and public holidays) in which the assessment is overdue. There is no pro-rata of the late penalty for submissions made part way through a day.

(End of document)