Mobile Robotics – Perception
Lab report: Attitude Estimation using Indirect Kalman Filtering

Date: 17 December 2020

1 Introduction

For this lab, we rely on the method in the article "Orientation Estimation Using a Quaternion-Based Indirect Kalman Filter With Adaptive Estimation of External Acceleration" by Prof. Young So Suh. Orientation estimation, i.e., estimating the pitch, roll, and yaw angles, has many robotics and biomedicine applications. This is usually done using three triaxial sensors: a gyroscope (to measure angular velocity), an accelerometer (to measure gravity), and a magnetic sensor (to measure the Earth's magnetic field). The problem with the gyroscope is bias and numerical integration errors, while that of using the accelerometer and the magnetic sensor output is external acceleration and magnetic disturbance. The mentioned paper discusses how to combine the outputs from the three sensors to use in an orientation estimation algorithm.
The chosen structure is a quaternion-based indirect Kalman filter, as quaternion has a singularity-free orientation representation, and the orientation error is estimated instead of the orientation (indirect). In this work, we program, using MATLAB, an adaptive orientation filter with the mentioned structure that has two-step measurement updates and that compensates external acceleration by estimating the direction of that acceleration and decreasing only the weight corresponding to accelerometer values affected by it. We then validate the method using simulation with the given sensor data. As an extension, a comparison with a norm-based adaptive and the standard algorithm is proposed.

2 The orientation estimation problem

In this orientation estimation problem, we need to get the correct values of the Euler angles (plotted in Figure 5 together with the results) from the following outputs by the three triaxial sensors:

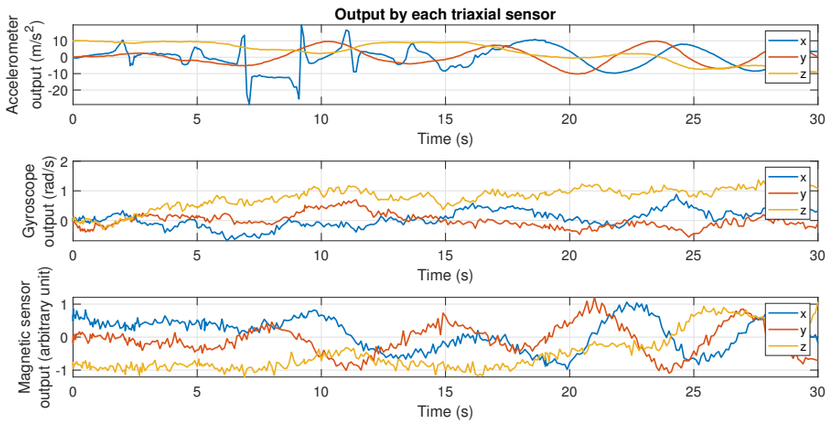

Figure 1: Sensor outputs.

The noises affecting these outputs on each sensor and axis are as follows, which are assumed to be zero-mean white Gaussian noises satisfying the conditions mentioned in the paper:

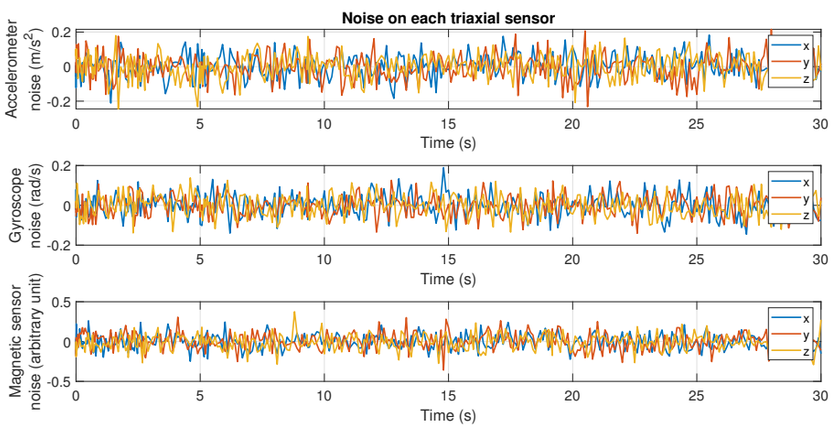

Figure 2: Sensor noises.

Besides, the measurement is affected by external acceleration, which is present mainly in the x-axis:

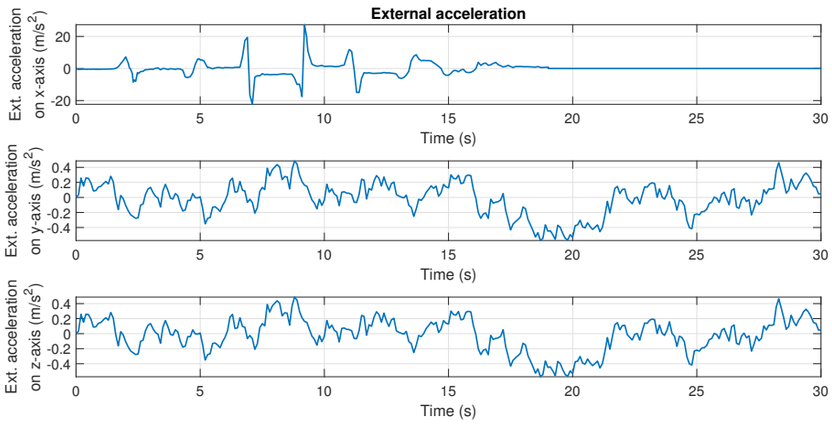

Figure 3: External acceleration.

To do the mentioned work, we program the algorithm explained and analyzed hereafter (see the `Compute_Attitude` function). This function takes the measurements and the noise error covariance matrices as input and computes the quaternions, Euler angles, and the estimated sensor biases using the proposed algorithm.

3 Algorithm explanation and remarks

3.1 The indirect Kalman filter and its discretization

This filter is an indirect one, i.e., the orientation error is estimated instead of directly estimating orientation. Its structure is as follows:

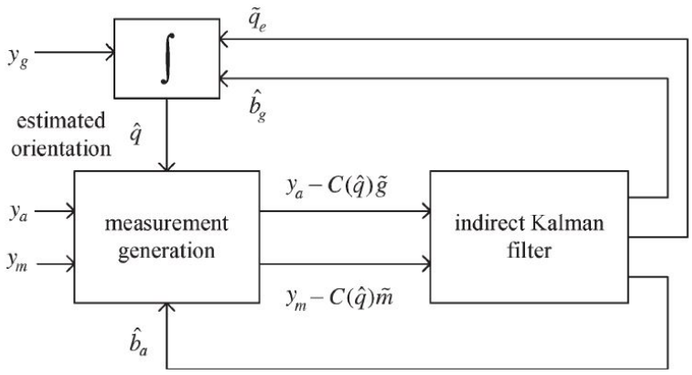

Figure 4: The indirect Kalman filter structure.

We iteratively update the quaternion and the sensor biases b_g and b_a by the indirect Kalman filtering algorithm where the state is $x = \begin{bmatrix} q_e & b_g & b_a \end{bmatrix}^T$. The system is periodically sampled with the sampling period T:

$$x_{k+1} = \phi_k x_k + w_k \tag{1}$$

where in real-time application $\phi_k = e^{AT} \approx I + A(kT)T + 0.5A(kT)^2T^2$ (Question 14 in the codes) and $Q_{d,k} = E\{w_k w_k'\} = \int_{kT}^{(k+1)T} e^{At} Q e^{(At)'}\, dt \approx QT + \frac{1}{2}A(kT)Q + \frac{1}{2}QA(kT)'$ (Question 15 in the codes). Based on the discrete model, a standard project ahead algorithm is used, i.e.,

$$\begin{cases} \hat{x}_{k+1}^- = \phi_k \hat{x}_k \\ P_{k+1}^- = \phi_k P_k \phi_k' + Q_d \end{cases} \tag{2}$$

where $\hat{x}_k^-$ and $\hat{x}_k$ are a state estimate before and after a measurement update and P_k^- and P_k are the estimation error covariances (Question 16 in the codes).

3.2 Two-step measurement updates

The magnetic disturbance could be considerable, and it affects not only yaw but also pitch and roll, so normally, we use the magnetic sensor output only for yaw estimation, but doing so, we risk losing information. Here, we use a two-step algorithm as follows:

- An accelerometer measurement ($y_a - C(\hat{q})\tilde{m}$ is used to update $\hat{x}_k^-$, where the updated state is denoted by $\hat{x}_{k,a}$). In the codes, this corresponds to Questions 19 to 22.
- $C(\hat{p})$ is updated using $\hat{x}_k^-$ and $\hat{q}$ is normalized. In the codes, this corresponds to Questions 23 and 26.
- A magnetic sensor measurement update ($\hat{x}_{k,a}$ is updated to $\hat{x}_k$ using $y_m - C(\hat{q})\tilde{m}$ while q_e is constrained so that the magnetic sensor output only affects the yaw angle). In the codes, this corresponds to Questions 24 and 25.

3.3 Adaptive algorithm compensating external acceleration

An adaptive algorithm to estimate external acceleration from the residual $r_{a,k}$ is proposed. $E\{r_{a,k}r'_{a,k}\}$ is a function of $Q_{a_b,k}$, the time-varying covariance of the external acceleration a_b. Since $E\{r_{a,k}r'_{a,k}\}$ cannot be obtained, it is approximated and so is $\hat{Q}_{a_b,k}$. An algorithm estimates the direction of the external acceleration. In the codes, this is done from line 118 to 136.
To guarantee that $\hat{Q}_{a_b,k} \geq 0$, an adaptive estimation algorithm of $Q_{a_b,k}$ has been proposed by defining two modes to be used in the case of no external acceleration and where there is external acceleration. Doing this, we can also prevent $\hat{Q}_{a_b,k}$ from being affected by normal fluctuation of accelerometer noises. The variable M_2 is introduced in the condition of the first mode to prevent falsely entering this mode when there is external acceleration (we only switch modes if the condition is satisfied $M_2 + 1$ times). In the codes, the switch condition is from line 138 to 149.

3.4 Some remarks about the codes

Remarks about the codes include our comments in the code files and the following:

- The measurements by the sensors are subtracted by the latest estimated bias of the corresponding sensor so as to model the bias feedback characteristic (see the filter structure).
- Even though in the paper, different notations of x have been used, here we only have one variable called `x` that is updated many times in each iteration.
- The rotation matrix C and the $[p\times]$ operator are programmed in separate functions which are the `quaternion2dcm(q)` and `vec2product(v)`, respectively, and the conversion from quaternions to Euler angles is done using the `quaternion2euler` function. Lastly, quaternion multiplication is performed using `quaternionmul(p,q)`.
- The unit of the Euler angles and the dip angle $\alpha = 50°$ (degrees or radians) must be respected.

Some useful comments can also be found in the codes.

4 Application and results

4.1 The proposed method

4.1.1 Orientation estimation

Here are the results of applying the proposed method to the given sensor data. By comparing the obtained results with the correct measurements, it can be seen that orientation estimation using this method is effective, as also proven in the paper.

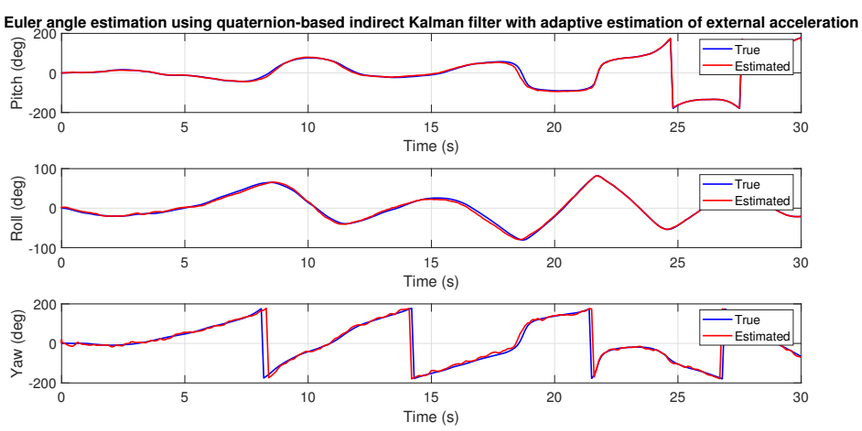

Figure 5: Orientation estimation using the proposed method.

4.1.2 Sensor bias estimation

We also wish to see the process of sensor bias estimation. To do this, we have developed two more arrays to store the values of the biases and plotted them:

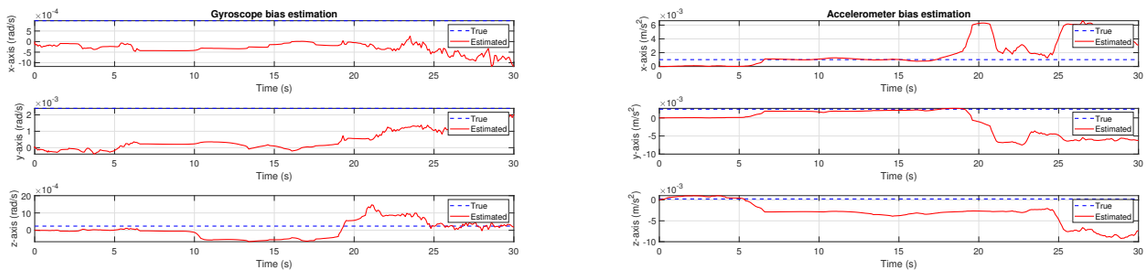

Figure 6: Gyroscope (left) and accelerometer (right) bias estimation.

It can be observed that the bias estimation does not have enough time to converge. In the paper, it is shown that this process takes around 60 to 100 seconds while the given data only allows us to simulate up to 30 seconds. We have also tried to concatenate the data, i.e., lengthen them by repeating one after another in hopes of having more simulation time. However, doing so results in an abrupt change that would cause the estimation to diverge again like at the initial time. Therefore, we need more data for this experiment.

4.2 Extension: Comparison with a norm-based adaptive method

We want to compare this method with an accelerometer norm-based adaptive algorithm described in the paper:

$$\hat{Q}_{a_b,k} = \begin{cases} 0, |\|y_{a,k} - 9.8\|_2 - 9.8| \leq 0.2 \\ sI, otherwise \end{cases} \tag{3}$$

and also with the standard Kalman filter where no adaptive algorithm is used, i.e., $\hat{Q}_{a_b,k} = 0$ all the time. In terms of programmation, as the only difference is with the $\hat{Q}_{a_b,k}$, we have only had to change the codes related to that part (both cases are programmed in the `Compute_Attitude_Norm` function). Results are as follows:

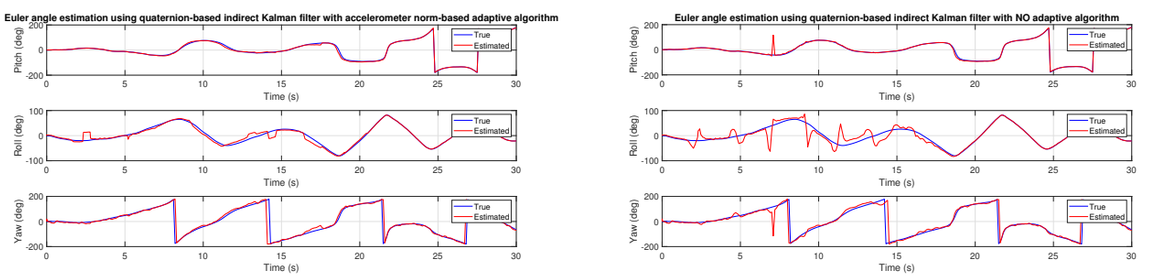

Figure 7: Results of the accelerometer norm-based adaptive (left) and standard algorithm (right).

We can see that the proposed method works better than the accelerometer norm-based adaptive algorithm. This is because in the mentioned norm-based method, having $\hat{Q}_{a_b,k} = sI$ (here $s = 10$) is equivalent to giving less weights to all three-axis accelerometer output whenever there is external acceleration (checked by the norm-based condition). If we do so when external acceleration is only applied to one axis, e.g., the x-axis, giving smaller weights on the other two (the y- and z-axis accelerometer output) will make us lose the information stored in these and we observe a phenomenon similar to that in Figure 8 of the paper. Lastly, with the standard Kalman filter, the results are the worst and coherent with those in Figure 7 of the paper. Therefore, the proposed strategy for $\hat{Q}_{a_b,k}$ has proven its effectiveness compared to the other two.

5 Conclusion

This lab has helped us understand attitude estimation using indirect Kalman filtering with two-step measurement updates and strategic compensation of external acceleration by directly programming and comparing it with the accelerometer norm-based adaptive and the standard algorithm. One limitation of this method is that the magnetic disturbances are not adaptively compensated.