# Sensor Fusion Graded Assignment 2

O. Kirkerud and B. Holm

October 2020

## 1 Introduction

This report is the second graded assignment in TTK4250, and explores the implementation and tuning of an Error State Extended Kalman Filter (ESKF) applied to a fixed wing Unmanned Aerial Vehicle (UAV.) The goal of the exercise is to make the estimated trajectory match the real trajectory. This entails tuning the filter parameters, then running the ESKF on simulated data, and then on real data. Performance metrics are collected to compare the efficacy of the ESKF on different estimation scenarios.

## 2 ESKF on simulated data

Tuning was initially carried out by choosing plausible first guesses for the tuning parameters. Then the values were adjusted until performance was satisfactory. By mistake, we were provided with a set of tuning parameters which produced good results. These were used as a benchmark, but not taken for granted.

Wrong initializing the vehicle states resulted in poor convergence time to the real trajectory. Bad first guesses for the predicted covariances seemed to have disastrous consequences for the whole performance, resulting in jittery, off-track behavior throughout. Great care was therefore taken when choosing these values. Proper initialization can be difficult in a real-life scenario, and in cases where good first guesses are hard to come by, ESKF might be less reliable than other schemes.

For the measurement noise and bias covariance matrix $Q_{IMU}$, the provided "solution" was calculated by looking at measurement noise standard deviation for the STIM300 IMU produced by Sensonor. This was investigated as a first guess, even though the measurements are simulated. After unit conversion, the standard deviations were found to be $\sigma_{a,n} = 5.835 \cdot 10^{-3}$, $\sigma_{g,n} = 2.18 \cdot 10^{-4}$, $\sigma_{a,b} = 2.4 \cdot 10^{-2}$ and $\sigma_{g,b} = 1.67 \cdot 10^{-6}$ for acceleration noise, gyro noise, acceleration bias and gyro bias, respectively.

For the noise parameters $\sigma_{a,n}$ and $\sigma_{g,n}$, scaling them up from here yielded poor results. This was expected, as the data looked to be relatively low in noise. Scaling them all the way down to zero yielded no visible change in the tracking accuracy for short timescales, but adversely affected the performance with more than ten thousand steps. This is probably due to the sharp turning that occurs some ways out in the set. However, we suspected that the simulated sensor was less noise-prone than the real one, so $\sigma_{a,n}$ and $\sigma_{g,n}$ were scaled down by 0.6. These new values improved the NEESes significantly, so were kept, and can be seen in table 1.

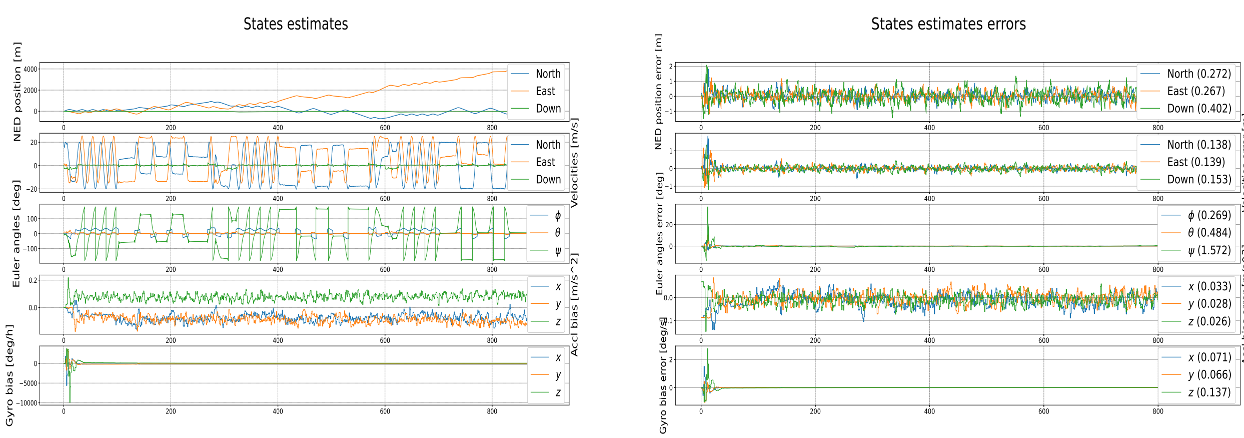
Like the case for the IMU noise parameters, the bias parameters were also expected to be too pessimistic. After some trial and error, scaling of $\sigma_{g,b}$ and $\sigma_{a,b}$ did not seem to improve the NEESes or NIS significantly. The original bias parameters seem to reflect the simulated IMU bias fairly well, and so were kept.

For the measurement noise covariance matrice $R_{GNSS}$, the variance was found empirically by comparing ground truth to the measurements. It was observed to be approximately $0.4m$ for the planar uncertainties, and a bit higher, $0.5m$, for the vertical uncertainty. The reason for the discrepancy between planar and vertical uncertainty is simply that inaccuracies caused by satellite geometry plays a bigger role in height estimation. After some tweaking, the values which yielded the best performance were $r_x = r_y = 0.35^2$ and $r_z = 0.51^2$.

As for the IMU bias time constants, these were assumed to be very large, as the bias varies very slowly, on a scale of hours. By setting the reciprocal time constants $p_{acc}$ and $p_{gyro}$ to 0, the bias time constants were effectively infinite, making them random walk processes. As this dataset is only on a scale of minutes, this idea seems to hold up well, producing the best results.

After tuning the measurement noise and bias matrices, as well as the bias time constants, the ESKF manages to track the simulated UAV rather well. The estimates keep close to the true path nearly everywhere, and

fig. 1a seems to capture the generally eastbound flight path, along with the small biases.
The estimates errors appear to be rather small, as seen in fig. 1b. Both roll and pitch only have negligible errors most of the run, but we can see that the heading error is significantly larger. Most of this stems from the beginning of the run, when the heading error experiences a large spike. Observing the Euler angle estimates for heading in this period reveals a heading that "jumps" and shakes frequently, with an accompanying increased error in speed and position. After this, the graph appears smoother for much of the rest of the run. This is likely due to the fact that the vehicle does not undergo much aggressive maneuvering in the beginning. Roll and pitch always have good observability due to the fact that they depend on gravitational force on the accelerometer, and gravity is of course always present. Heading estimation, however, is dependent on acceleration about the yaw axis, not on the gravitational force. It is therefore more difficult to estimate heading during periods of little maneuvering. A remedy for this might be an on-board, high-resolution magnetometer.
By the performance metrics in fig. 2a, the NEESes and NIS are observed to be quite good, with positional, velocity, attitude and accelerometer NEES, as well as the NIS, all falling within the 95% confidence interval over 90% of the time. However, the total NEES and the gyro bias NEES are both in the 70s, although this does not seem to hinder the performance much, and was deemed acceptable. The RMSE errors seen in fig. 2b of approximately $0.554m$ for position and $0.249m/s$ for speed are also very good when considering the large distances that are involved in the flight path.
Setting the IMU misalignment matrices $S_a$ and $S_g$ to the identity matrix in essence means we neglect mounting errors, scale errors and orthogonality errors. As expected, this does not appear to affect the GNSS aided estimates much, and these estimates are still able to follow the UAV. However setting the matrices to unity seems to cause the heading estimation, as well as both biases, to become quite unreliable. Ignoring correction also seems to cause the performance metrics to deteriorate, with a total NEES of less than 1% inside the CI. Likewise, the attitude NEES, accelerometer NEES and gyro bias NEES also become terrible, falling in the low 10s.
This goes to show that forgetting to take "trivial" errors such as mounting errors into account can have disastrous results on reliability. In real life, there will always be some misalignment, due to improper sensor mounting and alignment, as well as the fact that a moving vehicle is not a perfect rigid body.



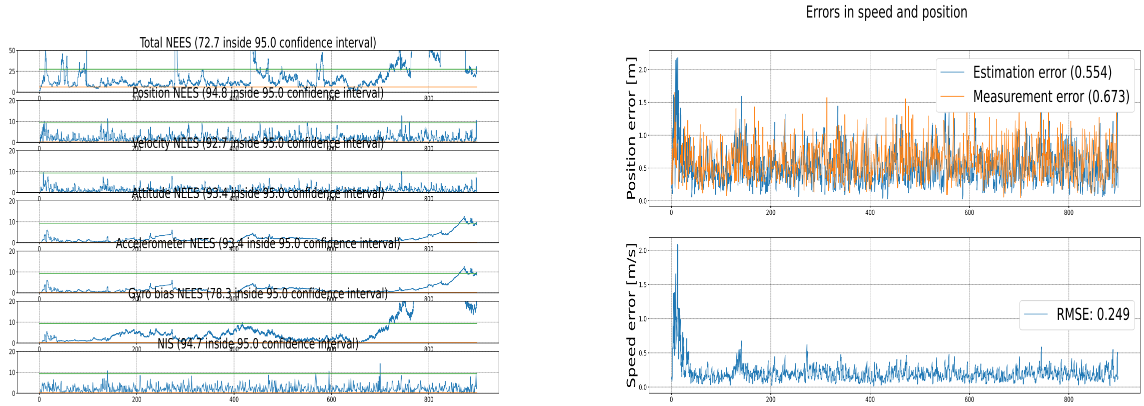

(a) States estimates for simulated data

(b) States estimates errors for simulated data

Figure 1: Estimates and estimates errors for simulated data

# 3 ESKF on real data

We are now provided with a dataset collected by a real UAV, and are tasked with tuning as in the previous exercise. However, this time we do not have access to ground truth. What we do have is access to is the datasheet of the onboard STIM300 IMU and a Ublox-8 GNSS receiver, from which we can extract good first guesses for some of the tuning parameters. These guesses will be used in combination with some parameter values from the previous task, and this will hopefully provide a good starting point.




(a) Performance metrics for simulated data

(b) RMSE for simulated data

Figure 2: Performance metrics and RMSE for simulated data

The datasheet of the Ublox-8 specifies an accuracy in velocity of $0.05m/s$. From this, combined with the known sampling frequency of $1Hz$, the positional noise was estimated to be $\frac{0.05m/s}{1Hz} = 0.05m$. For the IMU, the velocity random walk and the angular random walk were found in the datasheet as $0.06m/s/\sqrt{hr}$ and $0.15°/\sqrt{hr}$. Through some simple work with the units, they can be rewritten as $\sigma_{gyro} = 6.90 \cdot e^{-5} rad/s/\sqrt{Hz}$ and $\sigma_{acc} = 1.60 \cdot e^{-4} m/s^2/\sqrt{Hz}$.

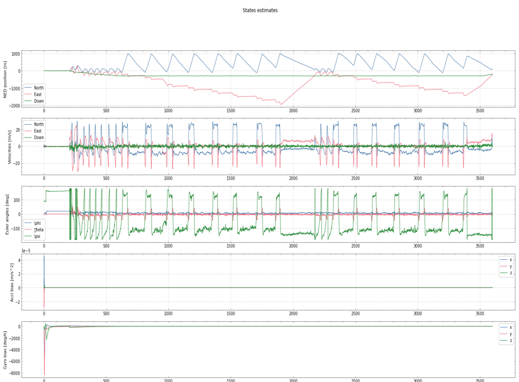
In the datasheet for the STIM300 IMU, there were no numerical value for the bias. However considering the quality of the IMU and the relative short time scale, one can initially neglect the bias by setting it to one thousand of the white noise. And set the time constants for the Gauss-Markov processes to 1 to minimize any development of bias.

By using the parameters described above in a combination with what we found in section 2, we get an ok result. The estimation drift out of position, only to be reset to the trajectory when the GNSS are updated. This results in a zigzag movement, where there are some deviations in x and y direction, and but better in z direction. The deviations in x and y direction are quite understandable, since the lateral positions are dependent on the yaw angle of the UAV which are only observable from integrating the gyro measurements. To make the ESKF weigh the updated gyro measurements less we increase the measurement noise a little. The system contains more bias than assumed, especially for the gyroscope. We therefore decrease the time constant significantly. Since there is no significant bias for the acceleration we increase the time constant. This leads to a very good estimation of the trajectory.

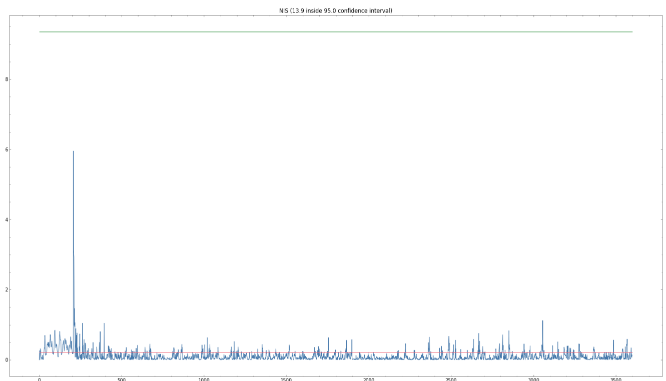
The gyro bias does however have a big drop in y direction in the beginning, visible in fig. 3a. This is probably a result of the UAV standing still at the beginning, as well as poor initialization of the filter when our model assumes that it is in motion. When the UAV is standing still, the filter is dominated by small signals such as white noise and bias. The filter then learns that the UAV is not in motion and the biases stabilizes eventually around 0. NIS can be found in fig. 3b, with an average about 2.3, meaning that the uncertainty the filter assigns the prediction, is about 1/2.3 of the actual error. The NIS is also not that often inside the confidence interval (13.9), meaning that we have an under-confident system. This does not seem to be an disadvantage in this scenario, as the estimated trajectory is following the ground truth exceptionally well. One can however make the system less under-confident by decreasing the measurement noise. It might be possible that we have wrongly calculated the amount the NIS is inside the confidence interval as well, and that this is the reason we get a somewhat small result for this.
As in the previous section, we attempt to set the IMU misalignment matrices $S_a$ and $S_g$ to the one given as:

$$S_a = S_g = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

As stated earlier when setting these to an simplified matrix, you might neglect mounting errors, scale errors and orthogonality errors. We can see that $S_a \approx R_z(-\pi/2)$ and $S_g \approx R_z(-\pi/2)$. Meaning that the measurements have to be rotated $-90°$, when going from sensor frame to body frame. When implementing an ESKF algorithm, one must take the position of the sensor relative to the body frame into consideration. The new matrix 1 are however $S_a = S_g = R_z(\pi/2)$. Meaning that we rotate it $+90°$. It is possible to see that something is off from the plots, when comparing the Euler angles before and after changing the misalignment matrices. The heading angle is rotated 180 degrees, when the UAV is flying south the estimated heading is pointing north. This isn't that easy to observe just by looking at the data, and it comes to show how important it is to either align sensors with the body frame, or remember this when implementing the state estimation algorithms. The new misalignment matrix does however not break the system, it actually does not have a big impact on its performance at all. This happens because the filter eventually learns that the IMU is orientated wrong, and compensates for that.



(a) State estimates for real data



(b) NIS for real data

Figure 3: State estimates and NIS for real data

# 4 Conclusion

An ESKF scheme has been shown to be superior to the standard KF when dealing with nonlinear systems such as UAVs. As long as the error states remain relatively small throughout the flight, a GNNS-aided inertial trajectory tracker performs very well, even when affected by noise and disturbances. Estimating bias has also been demonstrated to be an invaluable asset to perform corrections throughout the runs. Also, the importance of proper sensor mounting and alignment has become clear, as neglecting this showed a deterioration in certain state estimates. Improper initialization of the ESKF was shown to have catastrophic results for convergence time and tracking performance, meaning great care should be taken when picking this scheme for tracking applications. Certain limitations of purely inertial navigation have also come to light, particularly when it comes to the poor heading estimation that occurs with little active maneuvering.

| $\sigma_{a,n}$ | $\sigma_{a,b}$ | $\sigma_{g,n}$ | $\sigma_{g,b}$ | $r_x$ | $r_y$ | $r_z$ | $p_{acc}$ | $p_{gyro}$ |
|---|---|---|---|---|---|---|---|---|
| $3.5 \cdot 10^{-3}$ | $2.4 \cdot 10^{-2}$ | $1.308 \cdot 10^{-4}$ | $1.167 \cdot 10^{-3}$ | $0.35^2$ | $0.35^2$ | $0.51^2$ | 0 | 0 |

Table 1: Tuning parameters for simulated UAV

| $\sigma_{a,n}$ | $\sigma_{a,b}$ | $\sigma_{g,n}$ | $\sigma_{g,b}$ | $r_x$ | $r_y$ | $r_z$ | $p_{acc}$ | $p_{gyro}$ |
|---|---|---|---|---|---|---|---|---|
| $1.60 \cdot 10^{-4}$ | $1.60 \cdot 10^{-7}$ | $4.50 \cdot 10^{-3}$ | $6.90 \cdot 10^{-6}$ | $0.35^2$ | $0.35^2$ | $0.51^2$ | 1 | $5 \cdot 10^{-3}$ |

Table 2: Tuning parameters for real UAV