TTK4250 Graded assignment 2:
ESKF

Group 19
Andreas Haugland
Emil Martens

NTNU | Department of Engineering Cybernetics

Introduction and a brief explaination for choice of tuning

The parameters that were tuned are:

- The standard deviation of the IMU (accelerometer and gyroscope) measurement noise, driving noise of bias process and bias time denoted as a_n, a_{bn} and p_{ab} ω_n, ω_{bn} and $p_{\omega b}$
- The GNSS position noise denoted as $p_{std,NED}$. Here the variance in north and east are assumed to be equal.
- The initial covariance of the prediction. $\bar{\mathbf{P}}_{\mathbf{0}}(x)$

As there were 13 different parameters to tune, and the simulation time was significant, the unconstrained Nelder-Mead algorithm was chosen for the tuning. A drawback with the us of an optimization algorithm is that the values found may be more mathematically desired than physically plausible due to no constrains on the optimization[1]. An example of this is a larger uncertainty in the down-parameter of the GNSS measurement than those in the NE-plane, but where it in reality may be oposite. A compromise was made, and the values that both fitted well with the performance and what we deemed physically reasonable was chosen.

The rounded values used for task 3 and 4 can be seen in table 1 and 2 and are computed from the handout formulas.

Task 3 - INS on simulated data

Tuning

For the generated data, we had the ground truth. This was used during the tuning through the Nelder-Mead algorithm with the cost function seen in eq. The optimization were done in several steps. Firstly only the initial covariance was tuned over 30s simulated datase. Then all the variables were tuned, over the same time frame. Finally all variables were tuned over a 300s time window.

$$f(\mathbf{v}) = \frac{1}{N}\sum_{i=1}^{N} \log^2(\mathbf{v}_i)$$

$$\text{cost} = 20x_e^T x_e + f(\text{NEES}_{\text{pos}}) + f(\text{NEES}_{\text{vel}}) + f(\text{NEES}_{\text{att}}) + f(\text{NEES}_{\text{acc}_{\text{bias}}}) + f(\text{NEES}_{\text{gyro}_{\text{bias}}}) + f(\text{NIS}_{\text{pos}}) \tag{1}$$

This cost function minimized the error, through the first term, and the NEES and NIS values through the other terms. The function $f(\mathbf{v})$ made sure the cost of too low NEES and NIS values was proportional to the cost if they were too high. Using NEES_{all} did not converge to a better tuning. This cost function gave values close to ground truth and good results for the NIS and NEES values. No unreasonable values were observed.

Parameter	σ_{a_n}	$\sigma_{a_{bn}}$	p_{ab}	σ_{ω_n}	$\sigma_{\omega_{bn}}$	$p_{\omega b}$	$p_{std,N}$	$p_{std,E}$	$p_{std,D}$
First tuning	0.0058	0.0024	1e-16	2.180e-04	1.667e-06	1e-16	0.3	0.3	0.5
Final optimization	3.924e-03	1.486e-03	1e-9	2.744e-04	5.625e-04	1e-9	0.389	0.389	0.511

Table 1: Tuning parameters for task 2 - Tuning the INS to simulated data.

$$\bar{P}_0(\mathbf{x}) = diag(\begin{bmatrix}0.5845\mathbf{I_3} & 0.7812\mathbf{I_3} & 0.0011\mathbf{I_3} & 0.0217\mathbf{I_3} & 0.0094\mathbf{I_3}\end{bmatrix}) \in \mathbb{R}^{15x15} \tag{2}$$

Attitude

As can be seen in figure , the attitude estimation error is quite small, with RMSE_{att} = [3.85e-1, 3.78e-1 and 1.05e+0] for roll, pitch and heading.

Even though there are no direct measurements of the attitude, the attitude becomes observable through the GNSS measurements due to the movement, or rather acceleration, of the UAV. As long at the UAV is not in free fall, then the roll and pitch angle can be estimated by aligning the acceleration of the accelerometer with the gravitational acceleration. But when the UAV is in uniform motion or standing still, the heading is not observable. To observe the heading, one can imagine fitting cubic splines to the GNSS measurement points and aligning the measurements from the accelerometer with the estimated acceleration, the second derivatives,

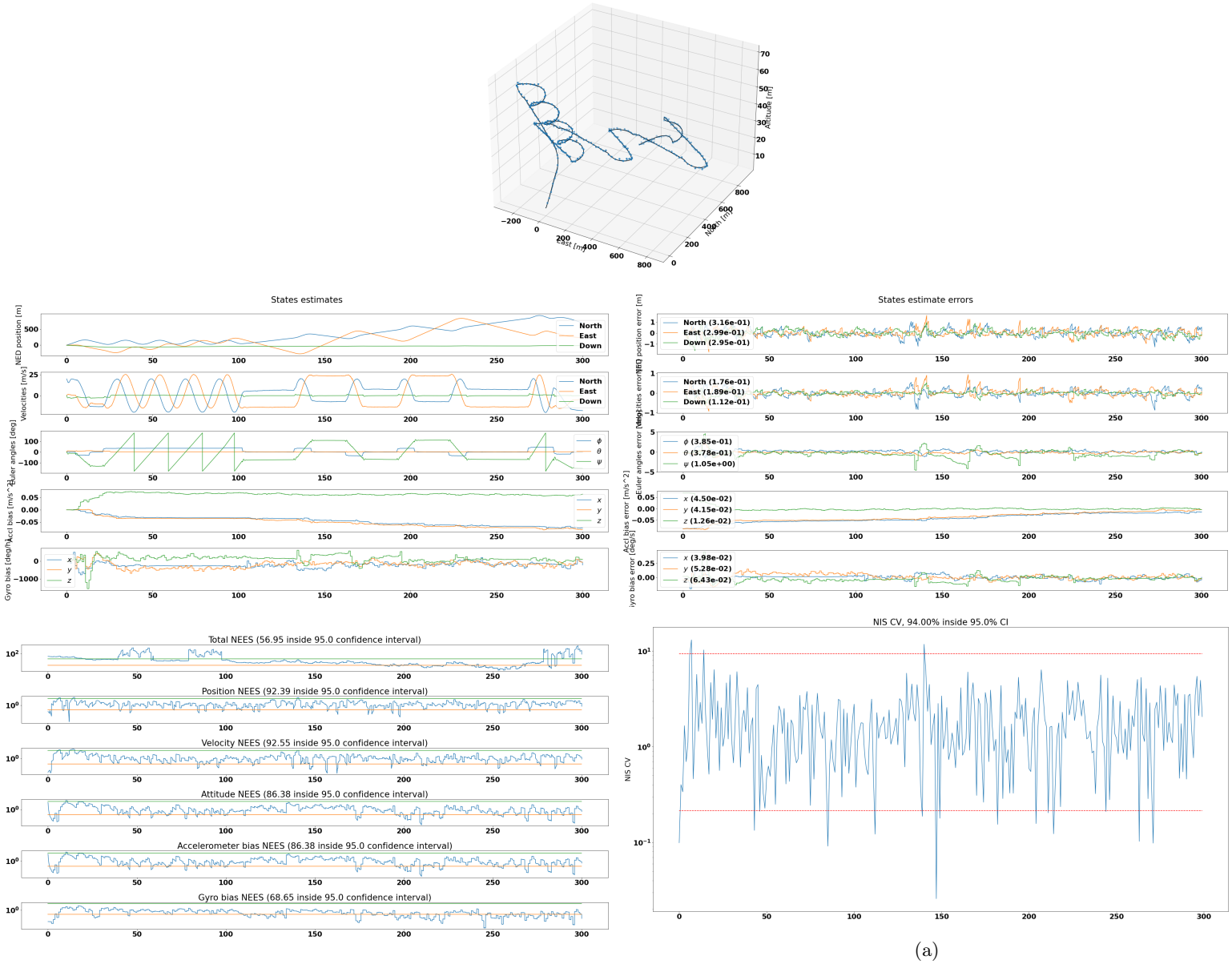
[1]Except for task 4 where a restriction on $p_std(3)$ was set. This should probably be a stricter constraint than what was used.

of these splines. During the a turn there is always centripetal acceleration. To estimate the heading during uniform motion, the model of the plane could have been taken into account, as planes are stable in the lateral plane. In other words, planes tend to not fly backwards.

Misalignment matrices

After changing the misalignment matrices to the identity matrix and thus disregarding misalignment and scale error, the ESKF still managed to follow the trajectory without altering its parameters. However a significant decrease in performance is observed in the NEES ans NIS values, as the filter is to confident. The misalignment matrix might disregarded in real life, if is insignificant, but as observed the matrix might improve the filters performance if it is taken into account. It is also plausible it might be possible to tune the filter to compensate for a slightly wrong misalignment matrix. The misalignment matrix can be estimated, for each of the principal axes, by mounting the IMU to a stationary mount and measuring the gravitational acceleration over a long period of time.

Resulting figures for the real data



(a)

Figure 1: Resulting plots for task 3 - INS simulation

Task 4 - INS on real data

Tuning

On this dataset, the reported accuracy data from the GNNS was used. The tuning parameters from the simulated data set were used as initial guess when tuning for the real data set, only adjusted for the new sampling time. As there were no ground truth available, the cost function was initially only based on the NIS value, cost $= f(\text{NIS})$, where f is the function descibed in eq. 1. The optimization ran over 30s of the dataset, with an offset of 206s as the UAV was stationary for approximately the first 205 seconds. The optimization appeared to give good results with 72.67% of the NIS values within the 95% confidence bound.

However, in reality the filter was useless. Since NIS is a measurement of the ratio between the magnitude of the innovation, and the covariance yielded by the filter, and the cost function was only based on the NIS, we got a filter with good NIS values, but large innovations and also large covariances.

Among many other problems, the filter estimated the UAV to have an angular velociy in the heading direction of almost one rotation per second, as can be seen in figure 3, due to to high bias driving noise. To resolve this issue a new cost function was used where the position covariance and bias was taken into account. This shows the importance of choosing a good cost function when using automated tuning, and the need to evaluate if the results are reasonable, when using an automated tuner.

Finally we updated the cost function to eq. 4, which aimed to keep the covarience as well as the bias values the position low. We observed significant improvement in performance, but it can be argued the optimized accelrometer noise is unreasonable, as it is significantly higher than the specifications of the IMU.

$$\text{cost} = f(\text{NIS}_{\text{pos}}) + 10^{-4}\boldsymbol{\rho}^T\boldsymbol{\rho} + 10^4\mathbf{a}_b^T\mathbf{a}_b + 10^4\boldsymbol{\omega}_b^T\boldsymbol{\omega}_b \tag{3}$$

Parameter	σ_{a_n}	$\sigma_{a_{bn}}$	p_{ab}	σ_{ω_n}	$\sigma_{\omega_{bn}}$	$p_{\omega b}$	$p_{std,N}$	$p_{std,E}$	$p_{std,D}$
Sim tuning	6.2038e-03	9.399e-04	1e-9	4.339e-04	3.557e-04	1e-9	0.389	0.389	0.511
First optimization	2.193e-02	5.323e-01	1e-9	1.076e-02	3.155e-01	1e-9	0.564	0.564	0.231
Final optimization	1.5988e-01	6.701e-04	1.420e-09	1.167e-03	1.804e-02	1.94e-09	0.106	0.106	0.107

Table 2: Tuning parameters for task 2 - Tuning the INS to simulated data.

$$\bar{P}_0(\mathbf{x}) = diag(\begin{bmatrix}0.5845\mathbf{I_3} & 0.7812\mathbf{I_3} & 0.0011\mathbf{I_3} & 0.0217\mathbf{I_3} & 0.0094\mathbf{I_3}\end{bmatrix}) \in \mathbb{R}^{15x15} \tag{4}$$

Resulting figures

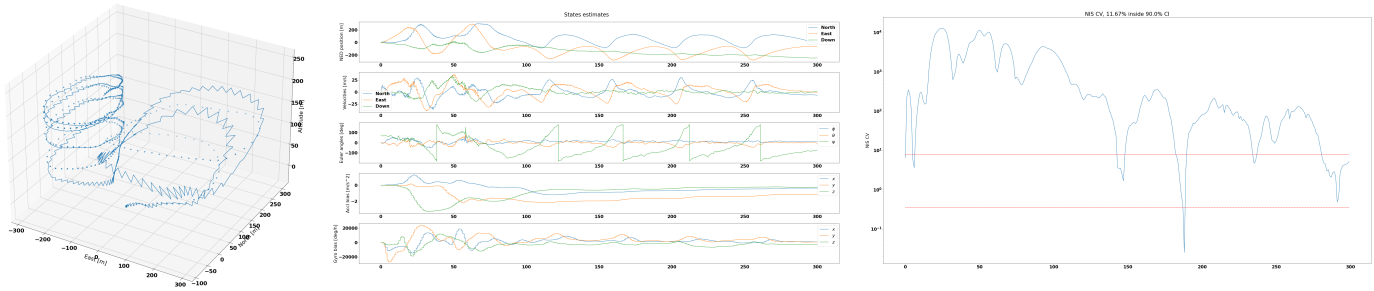


Figure 2: Real data with simulation tuning

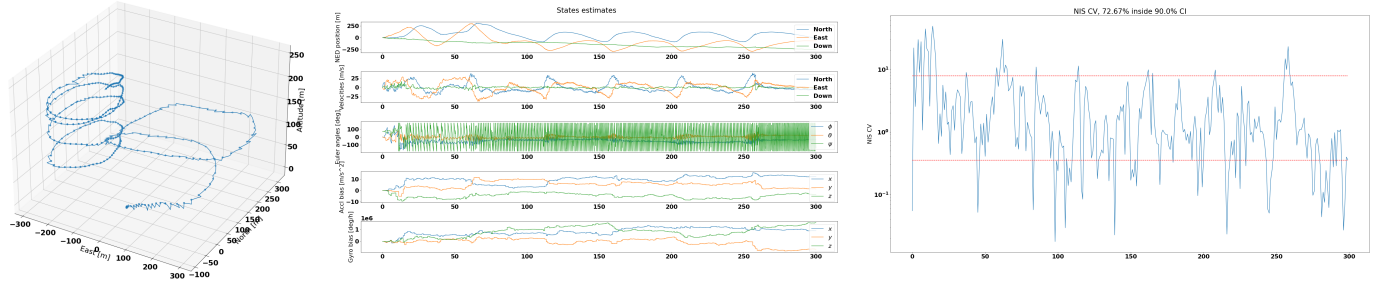


Figure 3: Real data with Nelder Mead optimization with focus on NIS only

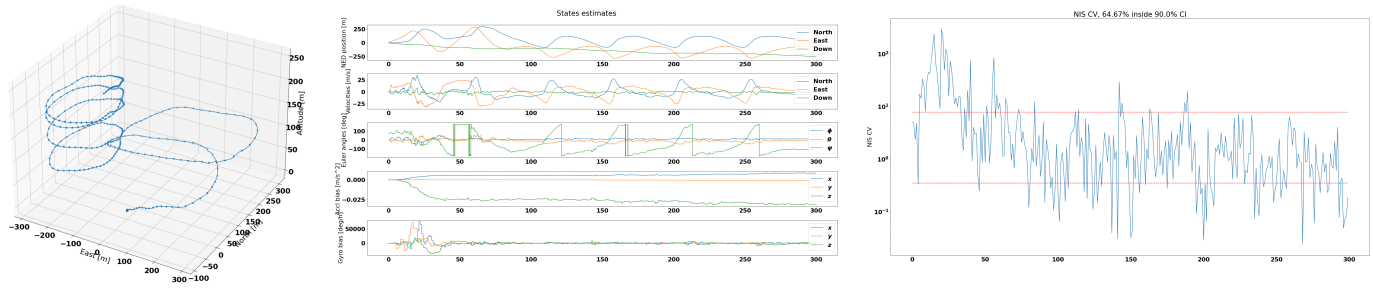


Figure 4: Real data with final Nelder Mead optimization

Misalignment matrices

Setting the misalignment matrix to the given simplified version had an unexpected effect. The performance of the filter appear to have improved, as the NIS values are closer to one and there were no observations of the trajectory that appeared unreasonable. The main difference was the accelerometer bias that ended up at a much higher value than previously, possibly to compensate for the missing misalignment matrix. We are not able to explain if these effects are due to unrealistic tuning parameters, or if the correction matrix was wrong. However this illustrates the difficulty of setting up a ESKF to be used in the real world. The number of parameters, and their interactions makes it difficult to know where a possible error might come from, and the filter has the ability to "hide" errors by compensating for them as if they were noise.

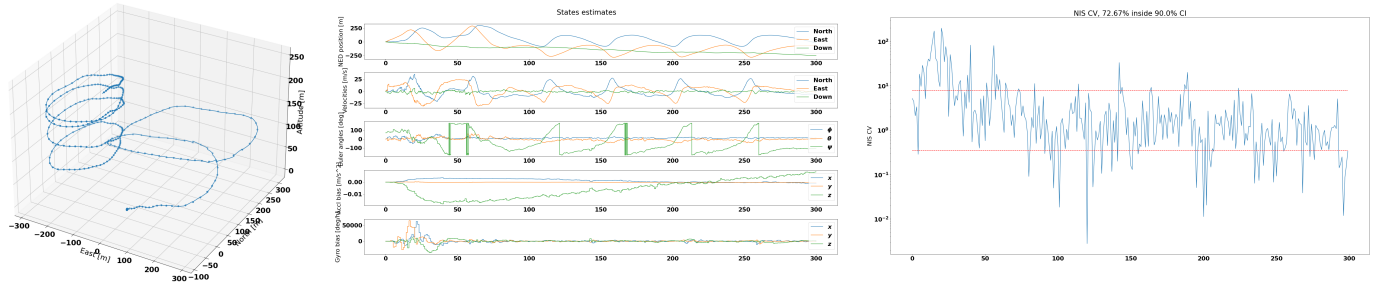


Figure 5: Real data with final tuning and no misalignment.

Final remarks

As a final remark we conclude that the ESKF is a great tool for state estimation and trajectory estimation for this control objective, but it is hard to implement. We also believe that our choice of tuning with an automatic tuner helped us a lot when tuning, as over 10 000 different parameter iterations were tested. But there are several pitfalls with this method. One is that this is a kind of brute-force method where you have to stop and think about what the algorithm is currently doing to guide it in the correct direction. Another pitfall is that unless there are implemented constraints on the mathematical values so they more represent values that are physically feasible, i.e ensuring that the UAV does not rotate around its own axis with close to 1 RPS and similar issues.

Throughout this project we had to stop several times and ask if "theese values are reasonable", check the performance and restart the autotuner with a push in a direction that was more reasonable. For further work we would tried again with more constraints and also spend more time on verifying the plausibility of the parametervalues.