NTNU
Kunnskap for en bedre verden

Department of Engineering Cybernetics

TTK4250 - Sensor Fusion

Graded Assignment 3

Authors:
Even Eliassen
Rendell Cale
(Group 64)

November, 2020

1 Introduction

For a robot to navigate in an unknown environment it needs to know its pose relative to its local environment. Simultaneous Localization and Mapping (SLAM) solves this problem by building a map and localizing the robot within the map at the same time. The goal of this assignment is to implement SLAM using an Extended Kalman Filter (EKF-SLAM) and Joint Compatibility Branch and Bound (JCBB) for data association, and evaluate it on a simulated data set and the Victoria Park data set (VP) [2]. The implementation is done according to [1].

2 Simulated data

The simulated data consists of odometry measurements, range-bearing detections of landmarks, and ground truth information for position, orientation and landmarks.

The EKF-SLAM algorithm is characterized by a small set of parameters. Following the notation in [1], we have a (discrete) odometry noise matrix parametrized as $\mathbf{Q} = \text{diag}(\sigma_x, \sigma_y, \sigma_\psi)^2$, and a range-bearing measurement noise matrix parametrized as $\mathbf{R} = \text{diag}(\sigma_r, \sigma_\theta)^2$. In addition to these there are two parameters for the JCBB compatibility test, they are $\alpha_{\text{individual}}$ and α_{joint}.

2.1 Tuning

Our tuning strategy was to first find reasonable values for $\mathbf{Q}$ and then tune $\mathbf{R}$ by looking mainly at NIS. Then we went back to $\mathbf{Q}$ and tuned by looking at NEES. $\mathbf{R}$ does affect NEES and $\mathbf{Q}$ affects NIS, so this was an iterative process. From the theory and previous experience, too high NEES or NIS means we should increase the uncertainties and vice versa.

The simulated data did not come with a physical description, so we used ground truth information for initial tuning. To tune the process noises σ_x, σ_y, and σ_ψ, we computed odometry error. From this, we set $\sigma_x = \sigma_y = 0.05$ and $\sigma_\psi = 1\,\text{deg}$. With these parameters fixed, we were able to tune $\mathbf{R}$ and obtain consistent NIS and decent position and heading RMSE at 0.7 m and 0.6 deg respectively. However we noticed that the (A)NEESes were not within their respective confidence intervals. We then tuned $\mathbf{Q}$ to improve NEES, as this is an important measure of the filter performance. Poor NEES values means that the filter has poor understanding of its own error.

To get more detailed information we decomposed pose NEES into position and heading, and NIS into range and bearing. This allowed us to spot that even though the total NEES/NIS was consistent, the decomposed NEES/NISes were not, which we could fix by further tuning.

After finding a tuning we were content with, we tried adjusting the alphas. To begin with, they were quite low, which seemed to work. By increasing $\alpha_{\text{individual}}$ from 10^{-5} to 10^{-3} we managed to lower the RMSE value, without affecting consistency too much. Unfortunately this also increased the number of duplicate landmarks and simulation time. This fits with the theory as higher alphas means fewer measurements are compatible with a landmark, leading to more landmarks, leading to more work for the JCBB algorithm.

2.2 Results and Discussion

The parameters we arrived at are given in table 1, and the results are shown in fig. 1. Note that to compute ANIS we have taken the mean of the (unnormalized) NISes, so ANIS should be scaled χ^2-distributed with degrees of freedom equal the total number of associations. Looking at the NEES plots in fig. 1a, we see that they are consistent with their confidence intervals, but the NIS is a bit too much outside as can be seen in fig. 1b. This indicates that the filter's estimate of its uncertainty in pose matches the error, while the uncertainty in landmark locations does not. The filter is underconfident in its position estimate since ANEES for position is a little low. Similarly, it is too uncertain about its bearing measurement since the bearing NIS is a little low. It should be possible to tune away these inconsistencies, but since the values are quite close to their confidence intervals we think this is not necessary.

Looking at the position error and comparing to the position of the robot, we see that the error increases when the robot visits new areas. When the robot gets close to the initial position at around timestep 550 and 800, the error becomes very small. It makes sense that the error would increase when exploring areas with few known landmarks. Since the new landmarks will be less accurate than the ones closer to the starting position, the estimates in regions further out will tend to be less accurate also when the robot returns to this area. The fact that the

Table 1: Tuned parameters for simulated data.

Parameter	Value
σ_{x}	0.025
σ_{y}	0.02
σ_r	0.06
$\alpha_{\mathrm{individual}}$	10^{-5}
α_{joint}	10^{-5}
σ_ψ	0.37deg
σ_θ	1.2deg

Table 2: Consistency evaluation for simulated data with 95 % CI.

	Inside	Averaged	CI
NEES pose	96.8%	2.934	(2.850, 3.154)
NEES pos	97.5%	1.818	(1.878, 2.126)
NEES psi	95.5%	1.034	(0.914, 1.090)
NIS	88.7%	19.197	(19.053, 19.826)
NIS range	90.5%	9.817	(9.448, 9.994)
NIS bearing	90.5%	9.406	(9.448, 9.994)

error drops when close to the origin indicates that the robot is able to recognize previously visited areas. If the filter is consistent, it should know that this happens and we should expect that the pose covariance increases when further away from the origin. To test this hypothesis we computed the norm of the pose covariance and error from ground truth at each time step and plotted them as heatmaps over the trajectory. These plots confirmed to us that error and pose covariance norms increase as we go further away from the starting point.

There does not seem to be any duplicate landmarks added to the estimated map. This indicates that the data association works as it should.

3 Real data

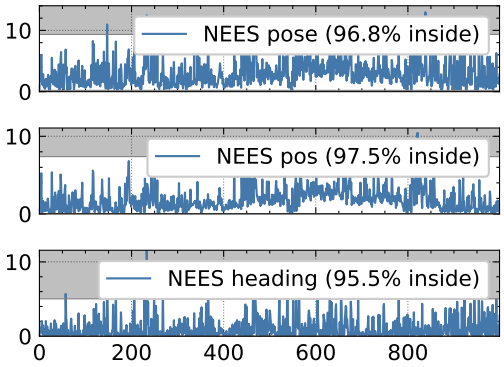
The VP dataset contains data from a lidar mounted on a vehicle as it drives around in a park. The rear wheel is equipped with an encoder which together with the steering angle is converted into an odometry measurement. For our purposes the setup is identical to the simulated data setup, except without ground truth measurements. The vehicle has a GPS, which is useful for evaluating the performance, but it is not fused into EKF-SLAM.

3.1 Tuning

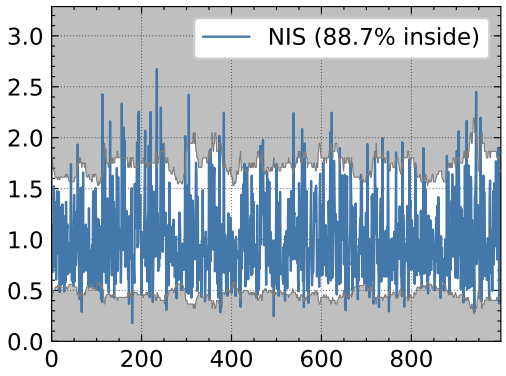
To start off the tuning process on the VP data, we used the tuning from the simulated data. This resulted in poor performance, which wasn't surprising given that, as far as we know, the simulated data was not made to resemble the VP data. To proceed, we set $\mathbf{Q}$ and $\mathbf{R}$ by using physical reasoning and the sensor descriptions. The laser returns 361 measurements divided evenly in a half circle, giving a resolution of 0.5 deg, so we expect σ_θ to be close to this value. Odometry measurements come every 25 ms, so a vehicle moving at 20 km/h would give odometry measurements around 0.1 m, so we expect σ_x and σ_y to be significantly smaller than this. We did not get very good consistency this way. We also tried to tried to tune the odometry parameters separately by disabling SLAM and only integrating the odometry measurements, and computing NIS using GPS measurements. To do this we used a scaled 2x2 identity matrix to represent the GPS noise covariance, $\mathbf{R}_{\mathrm{GNSS}} = 2^2\mathbf{I}_2$, and limited the run to 3000 steps so that odometry integration errors wouldn't be too large. Following this process gave us very big uncertainties for the odometry and did not give better results. We also tried to look at NIS using SLAM position estimates and GPS measurements, but the difference between estimates and GPS measurements were quite large and very sensitive to the initial pose, so we decided not to use this strategy for tuning.

Since the above methods did not result in good performance, we decided to see if we could get better results by just looking at total and decomposed NIS, while not restricting us too much to the physical interpretation of the parameters. This strategy yielded the best performance on the VP data.

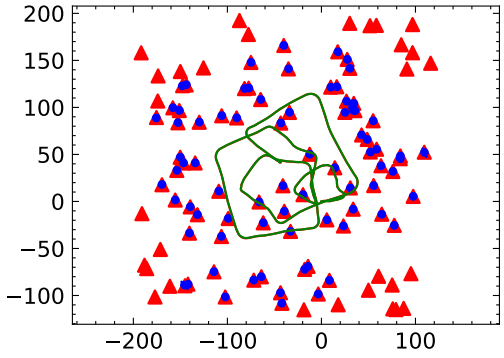
We adjusted the alphas of the JCBB algorithm and found that, as with the simulated data, they affected both NIS values and run time. Small alphas give less strict associations and resulted in fewer landmarks being added. This in turn made the covariance matrices smaller, giving faster run time, but only to a point. Too small alphas means more measurements needs to be checked by JCBB to find the best set of measurements, which could lead to longer run time. So reducing alpha could both reduce and increase run time. With higher alphas, we got more landmarks being created in close proximity, which are most likely duplicates of the same landmark. This could perhaps explain variations in NIS. For example if five measurements of a landmark result



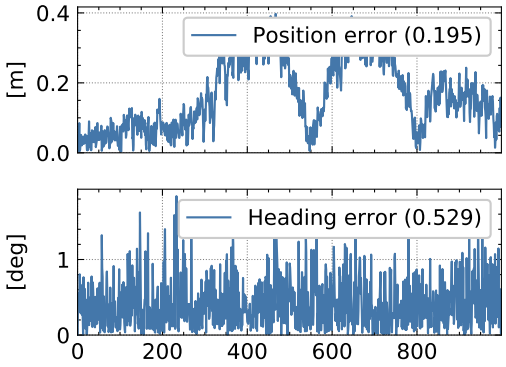
(a) NEES over time over simulated data. Grey region is outside 95% confidence interval.



(b) Normalized NIS over time in simulated data. Grey region is outside 95% confidence interval.



(c) Estimated and ground truth trajectory and map.



(d) Error from ground truth over time. Numbers in legends are RMSE values.

Figure 1: Figures for simulated data.

Table 3: Tuned parameters for VP data.

Parameter	Value
σ_{x}	0.01
σ_{y}	0.01
σ_r	0.1
$\alpha_{\mathrm{individual}}$	10^{-5}
α_{joint}	10^{-7}
σ_ψ	0.2deg
σ_θ	0.2deg

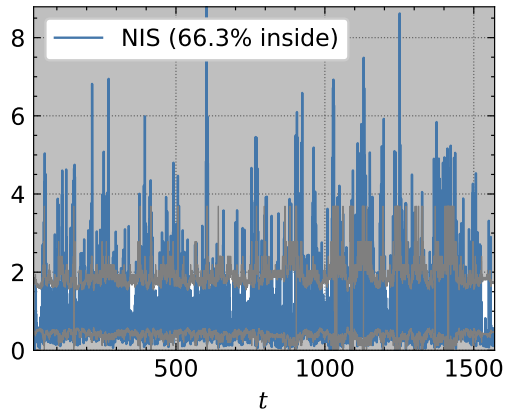
in five different landmarks estimates, these estimates will each be more uncertain than if all five measurements were associated to the same landmark. This corresponds with our results, where higher alpha gave lower ANIS.

3.2 Results and Discussion

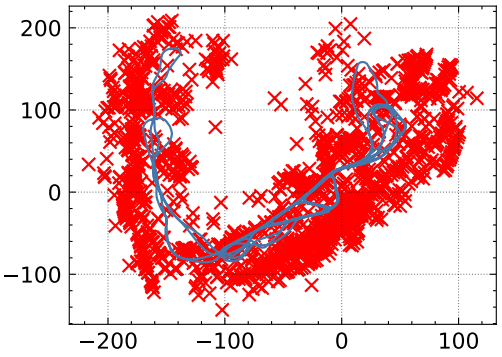
The result of the tuning given in table 3 are shown in table 4 and fig. 2.

In general we seem to have a lot of landmark duplication, big deviations from GPS measurements, and inconsistency. This is different from the simulated data set results. One important reason for this is that the simulated data is generated using the same modeling assumptions we use in the filter. The real data might have bias, colored noise, and clutter measurements, which could lead to excessive landmark creation and challenging data association. There are a lot more iterations in VP, so errors accumulate over longer time. This makes it interesting to investigate how the covariance changes during the run. We computed the norm of the pose covariance at each time step and plotted it over trajectory, as shown in fig. 2d. We observed that the pose covariance is in general larger when further away from the starting point, which is consistent with what was found on the simulated data.

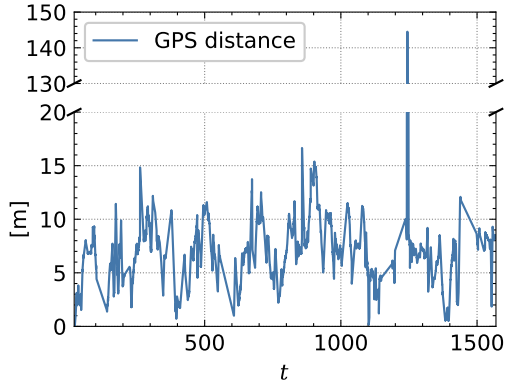
A limitation of the above analysis is that it assumes the filter is consistent, but table 4 indicates that we have a consistency issue, meaning the filter is not able to give a good description of its errors. Internally, this could lead to poor data association because the innovation covariance is used in JCBB. In general the ANIS values are low, which is preferable to large, since



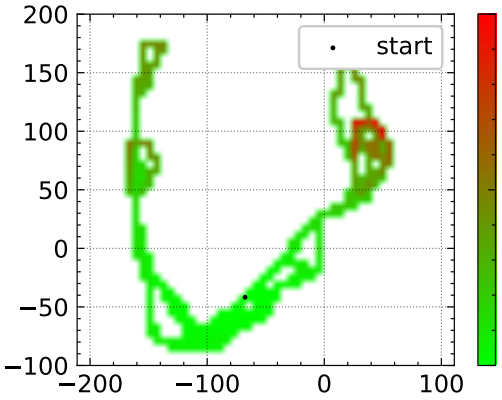
(a) Normalized NIS for landmarks over time. Grey region is outside 95% confidence interval.



(b) Estimated trajectory and map. steps 61944, laser scans 7247, landmarks 1686, measurements 10, num new 0



(c) Distance between GPS measurement and position estimate



(d) Norm of predicted pose covariance, $\|P_{xx}\|$, over trajectory. Red means larger covariance.

Figure 2: Results for VP data.

Table 4: Consistency evaluation for VP data with 95 % CI.

	Inside	Averaged	CI
NIS	66.3%	12.366	(14.027, 14.272)
NIS range	80.3%	5.531	(6.988, 7.162)
NIS bearing	73.7%	6.068	(6.988, 7.162)

large values indicate that the filter is overconfident, which could result in dangerous maneuvers when used in a control system.

While the inconsistencies could be explained as poor tuning, we think it is a shortcoming of EKF-SLAM. The literature [1] suggests that it is connected to the unobservability of the pose, and errors when linearizing about estimated state instead of true state.

In addition to the improvement strategies mentioned in [1], we think detecting and merging duplicate landmarks could improve performance.

4 Conclusion

We tuned the EKF-SLAM algorithm on both a simulated and the Victoria Park data set. In the simulated data set there were fewer duplicate landmarks and less inconsistent results than in the VP data set. This could be because the simulated data was generated with the assumptions that are used in EKF-SLAM, while VP is real data. We think the consistency issues are a shortcoming with EKF-SLAM. We also noticed that errors and uncertainties increased when the robot travels away from the starting position. We think there are two main reasons for this; (1) error accumulates over distance traveled since erroneous pose estimates lead to erroneous interpretations of measurements, and (2) when exploring new regions the robot has fewer known landmarks to associate with. Since SLAM does not incorporate global data like GPS, we don't see a way around this.

Bibliography

[1] Edmund Brekke. *Fundamentals of Sensor Fusion*. 2020.

[2] Jose Guivant and Eduardo Mario Nebot. 2001. URL: http://www-personal.acfr.usyd.edu.au/nebot/victoria_park.html.