TEL AVIV UNIVERSITY אוניברסיטת תל-אביב

THE IBY AND ALADAR FLEISCHMAN
FACULTY OF ENGINEERING

הפקולטה להנדסה
על שם איבי ואלדר פליישמן

Project2

Mapping and perception for an autonomous robot/ 0510-7591

By: Tomer Shimshi

Tel Aviv University

August 2023

Abstract:

This project aims to implement and analyze three algorithms, namely, the Kalman Filter, the Extended Kalman Filter, and the EKF-SLAM algorithm. The first section of the project involves implementing the classic Kalman Filter. The KITTI OXTS GPS trajectory from the recorded data **2011_09_26_drive_0061** will be used to extract latitude, longitude, and timestamps. The LLA coordinates will be transformed into the ENU coordinate system and Gaussian noise will be added to x and y of the ENU coordinates. The Kalman Filter will then be implemented on the constant velocity model with noisy trajectories to approximate the ground truth trajectory. The appropriate matrices and initial conditions will be calibrated and initialized to minimize the RMSE error and achieve a maximum error less than 7. The covariance matrix of the state vector and dead reckoning the Kalman gain after 5 seconds will be analyzed to observe their impact on the estimated trajectory. The x-y values and the corresponding sigma value along the trajectory will also be analyzed separately. In the second section, the Extended Kalman Filter will be implemented using the same data and noised trajectories as in the first section. The nonlinear motion model will be dealt with while still applying the Kalman Filter on it. The appropriate matrices will be initialized and computed, and the results will be plotted and analyzed similarly to section 1 to reduce the RMSE and maxE and observe how it deals differently with dead reckoning. In the third section, the EKF-SLAM algorithm will be implemented. The odometry motion model with Gaussian noised inputs will be run, and assumed measurements from the state with some Gaussian noise will be used. The predicted state of the motion will be implemented, and the correction of the state will be computed by considering the effect of each observed landmark on the Kalman gain, corrected mean, and uncertainty matrix for each time step. The estimated localization and mapping will be fully computed using observations at the relevant time steps and motion commands. The results will be analyzed to reach minimum RMSE and maxE values. The estimation error of X, Y, Theta, and two landmarks will also be analyzed.

Contents

Solutions

Kalman Filter

A

We Downloaded the dataset from the Kitti dataset website, the dataset that is going to be tested throughout this project is 2011_09_26_drive_0061

B

In this section we have extracted vehicle GPS trajectory from KITTI OXTS senser

packets which are treated as ground truth in this experiment:

- lat: latitude of the oxts-unit (deg)

- long: longitude of the oxts-unit (deg)

- Extract timestamps from KITTI data and convert them to seconds elapsed from

the first one

C

Here we transformed the GPS trajectory from [lat, long, alt] to local [x, y, z] ENU coordinates in order to enable the Kalman filter to handle them and plotted the GT LLA and ENU coordinates.

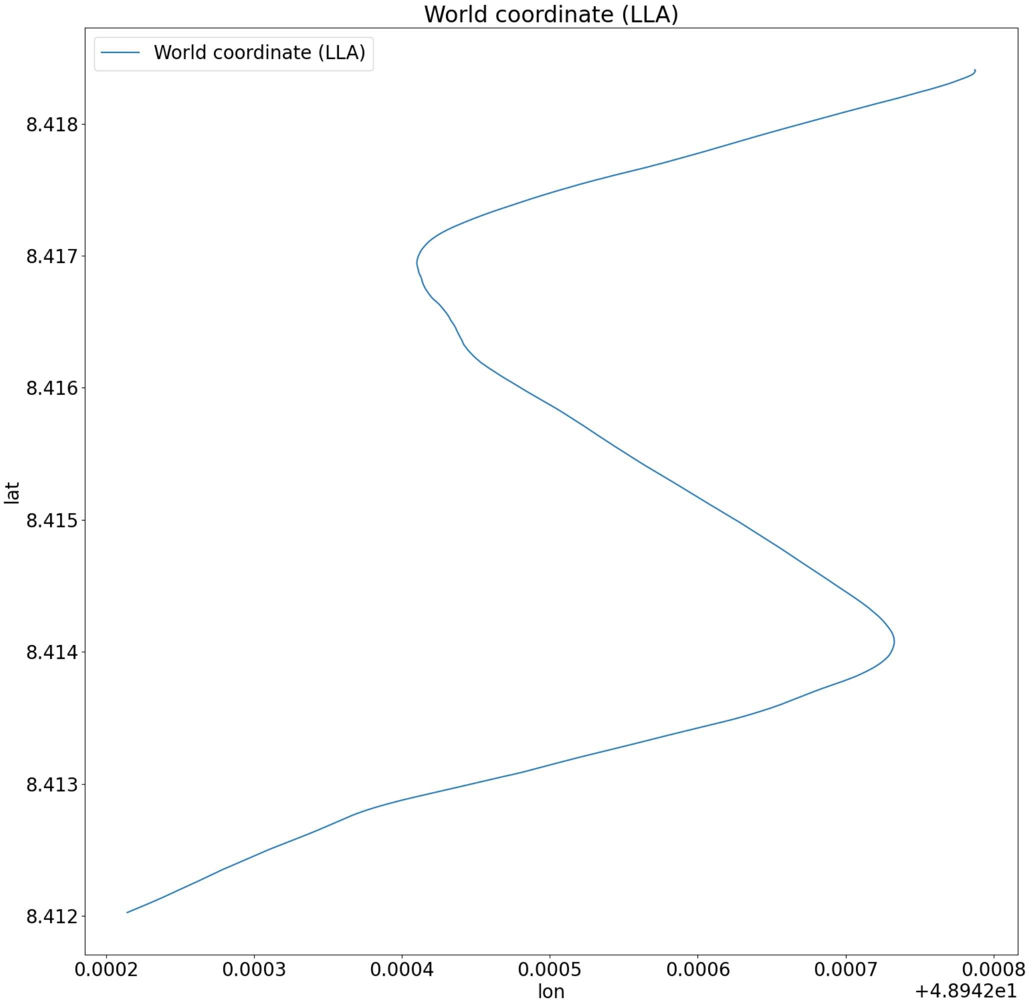

Figure 1 World coordinate (LLA)

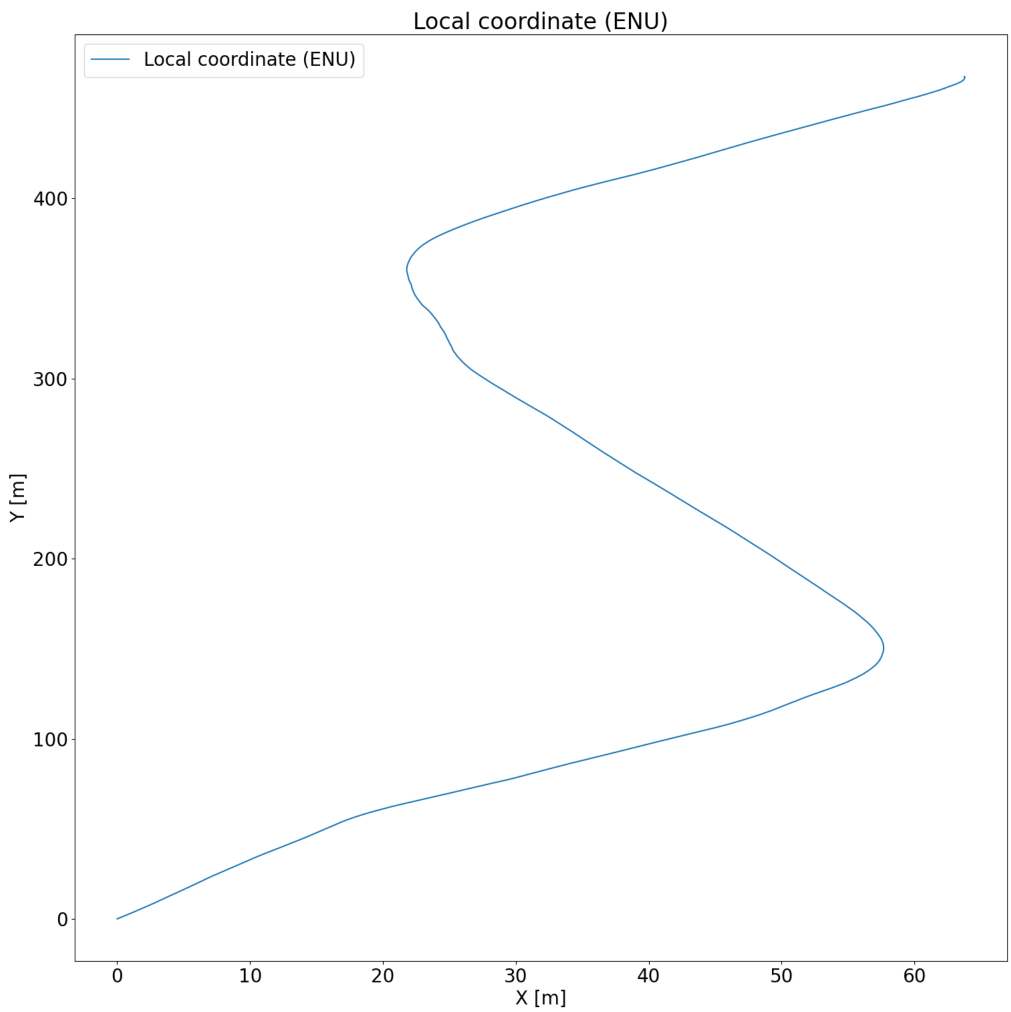

Figure 2 Local coordinate (LLA)

We can see that the vehicle drove North ~150m and east for ~50m than made a curve and drove west for ~20m and north ~200m did another curve and drove east ~30m and north ~80M

D

Here we added Gaussian noise to the ground-truth GPS data which will be used as noisy observations fed to the Kalman filter. Noise added with standard deviation of observation noise of x and y in meter (σx = 3 , σy = 3). In the next figure we can see the original GT and observations noise

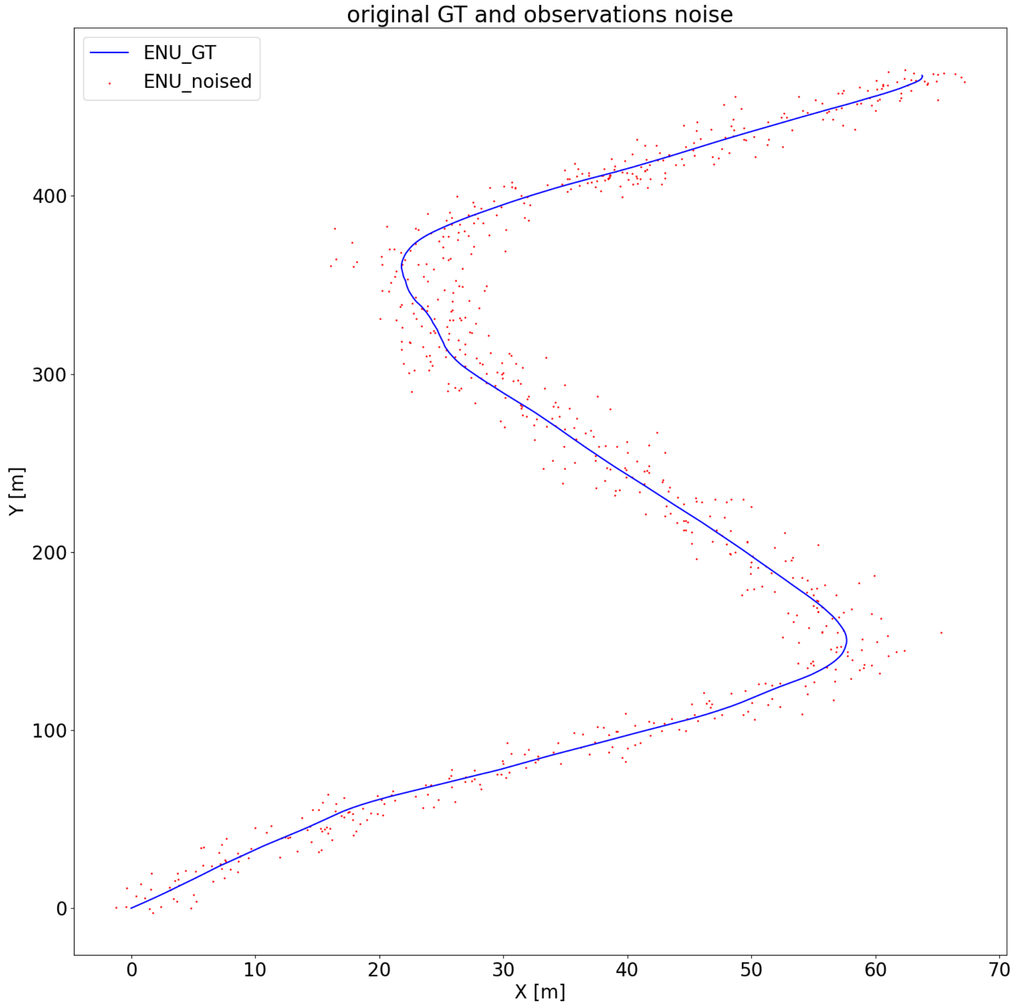

Figure 3 original GT and observations noise

E

Here we will apply a linear Kalman filter to the GPS sequence in order to estimate vehicle 2D pose based on constant velocity model Will suppose initial 2D position [x, y] estimation starts with the first GPS observation (the noised one), GPS observation noise of X and Y is known (σx =3, σy =3). Our goal will be to minimize the RMSE which is defined as:

$$RMSE \triangleq \sqrt{\frac{1}{N-100}\sum_{i=100}^{N}[e_x^2(i)+e_x^2(i)}$$

$$e_x(i) \triangleq \; x_{GT} - x_{Estimate}$$

$$e_y(i) \triangleq \; y_{GT} - y_{Estimate}$$

$$maxE \; \triangleq \max\{|e_x(i)| + \; |e_y(i)|\}$$

1. Initial conditions: according to your first observation the values of standard deviations initialized:
   X = $\begin{bmatrix} x_0 \\ v_{x0} \\ y_0 \\ v_{y0} \end{bmatrix} = \begin{bmatrix} x_{est0} \\ 1 \\ y_{est0} \\ 10 \end{bmatrix}$

   This is because we can see from the initial observation that at the beginning of the drive is north east so and more to the north so we set $vx0$ to be 3 and assumed a value for $vy0$ to 10.

$$P_0 = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & \sigma_y^2 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}$$

   This is because we want our first covariance uncertainties of our state to be relatively high at the beginning as we have not yet got corrections of our state so we are less certain of our initial condition after we run the algorithm this uncertainty covariance's will converge to contain 66% of the error if it contains more than 66 percent, we can decree the appropriate uncertainty. hence, we chose the uncertainty of X and Y according to the variance of the measurements and for the velocities we choose a high enough number to be able to handle the unknown velocities.
2. Matrixes A,B,C:

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad B = None\ (constant\ velocity), c = \begin{bmatrix} 1 & 0 & 00 \\ 0 & 0 & 10 \end{bmatrix}$$

   Corresponding to const velocity model $x_t = A \cdot x_{t-1} + B_t \cdot x_t$ while we only observe x and y form here matrix C
3. Measurement covariance (Q):
   $Q = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}$
   Corresponding to the measurement noise, we only measure x and y hence matrix of size 2x2.

4. transition noise covariance R:
   Containing the process noise in the const velocity model this is the source of the change in speed making it dynamic hence after analysing different values.

$$R = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Delta t \end{bmatrix} \cdot \sigma_n \ ; \ \sigma_n \ = 1.0$$

You can see in the next graphs the values RMSE an maxE compared to values of σ_n :

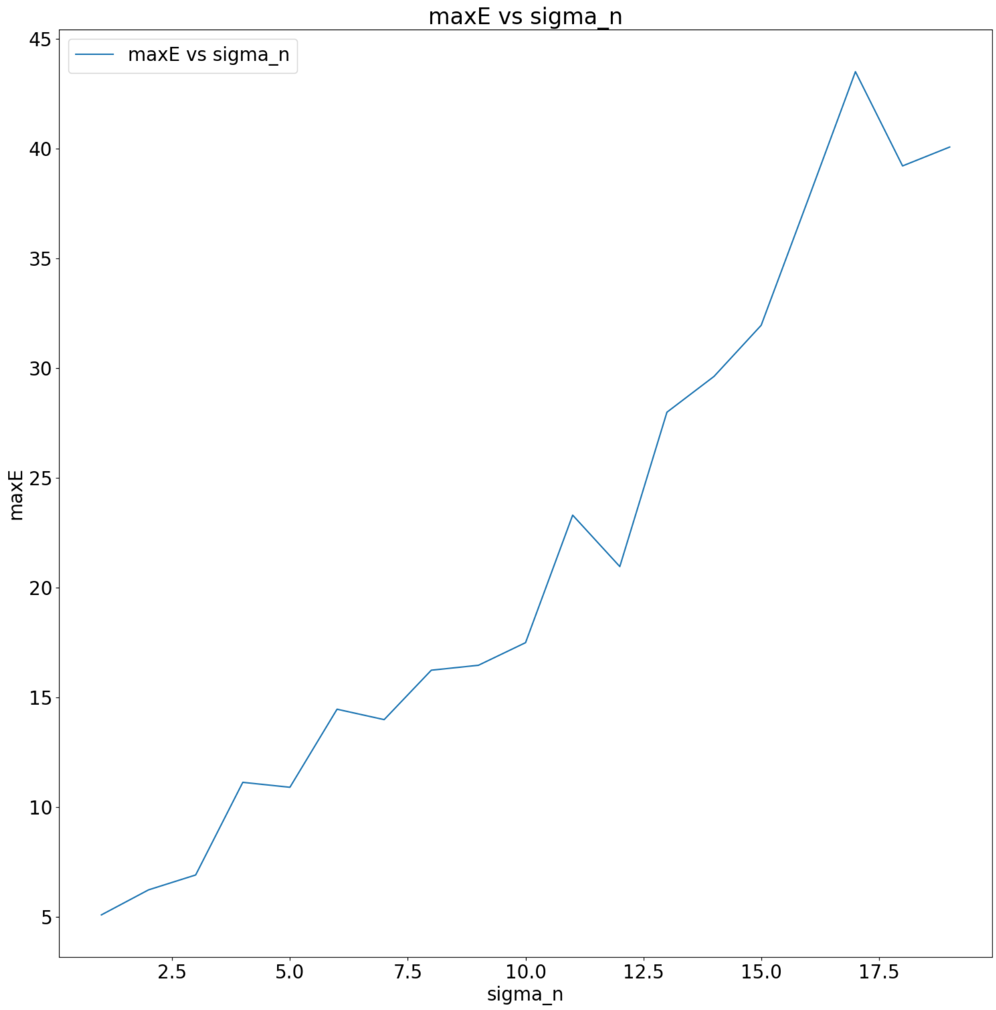

Figure 4 maxE vs sigma_n

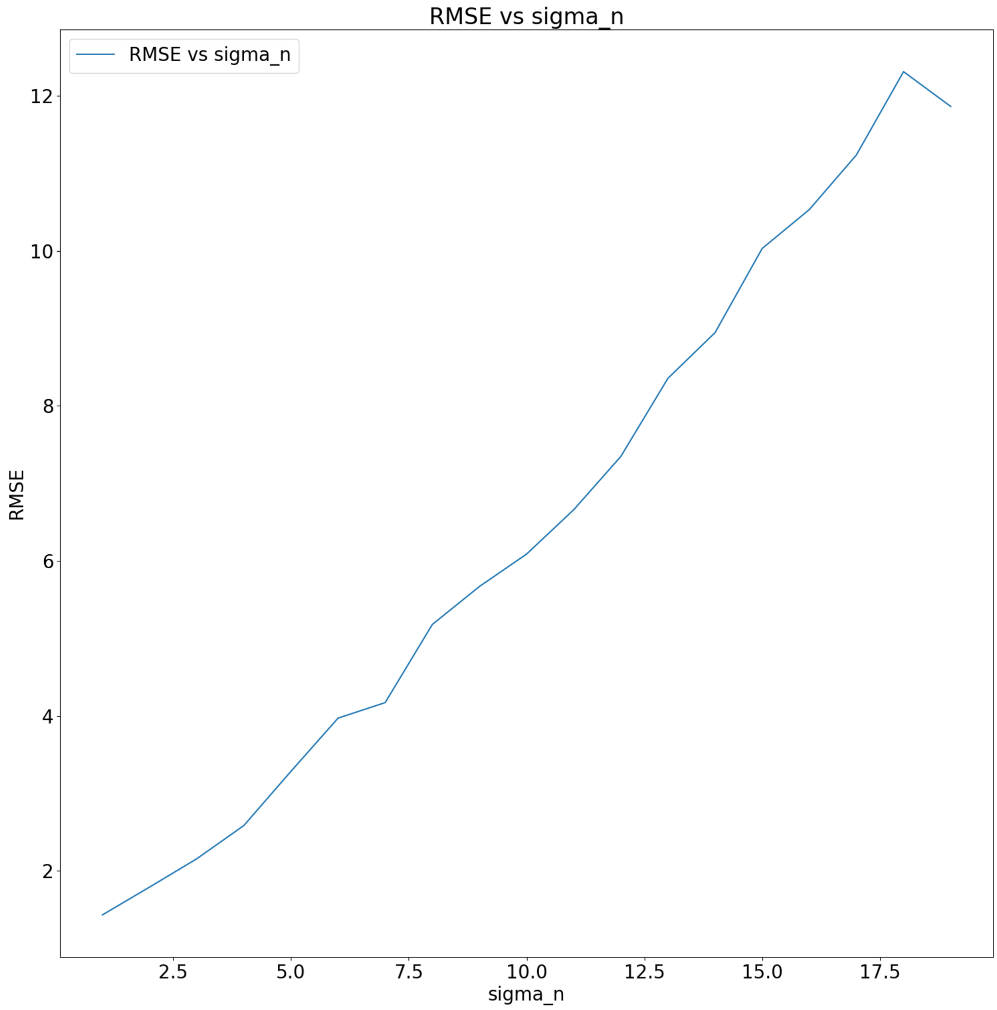

Figure 5 RMSE vs sigma_n

As we can see the lower the sigma the better the Kalman Filter performance, on top of that we reach a desired results with σ_n that is set to 1 hence that is the sigma that was used

<u>Kalman filter main routine:</u>

The state mean and uncertainty covariance matrix was initialized as above sections, using the KalmanFilter object. We used the KalmanFilter.run() function was called this function iterated over all time steps and ran the Kalman filter on all the ENU noised measurements. the KalmanFilter.run() contains the prediction of the location and uncertainty covariance by calculating:

$$(1)\ \bar{x}_t = A_t \cdot x_{t-1} + B_t \cdot x_t$$

$$(2)\overline{P_t} = A_t \cdot P_{t-1} \cdot A_t^T + R_t$$

And the correction srep:

$$(3) K_t = \bar{P}_t \cdot C_t^T (C_t \cdot \bar{P}_t \cdot C_t^T + Q_t)^{-1}$$

$$(4) x_t = \bar{x}_t + K_t (z_t - C_t \cdot \bar{x}_t)$$

$$(5) xP_t = (I - K_t \cdot C_t) \bar{P}_t$$

step 1: we predict our mean location based on our model (contained in A and B)

step 2: we predict how our uncertainty carries on to the next time step by our motion (in A) and add process noise R.

step 3: we compute our Kalman gain which controls the emphasis on the deviation between what we predicted and what was measured. Where C_t maps our predicted state to the observed state.

Step 4: computes the corrected mean based on the kalman gain computed and deviation of expected location and observation.

Step 5: computes the corrected covariance matrix also based on the kalman gain. because our model is linear all transformations can be done by matrix multiplication and assumption of Gaussian distributions hold throw all transforms.

F

Results analysis:

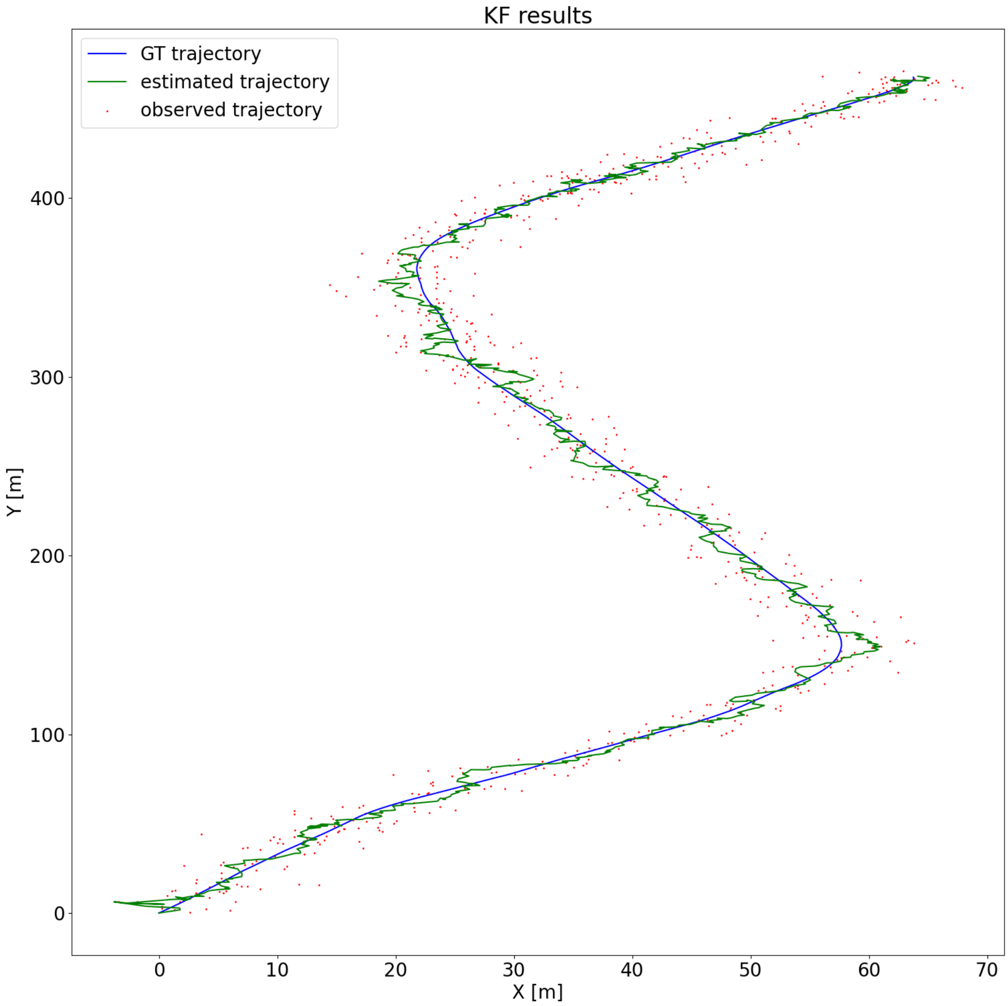

Figure 6 Ground-truth and estimated results

We can see that even with relatively high noise and constant velocity model which is not the most exact motion model for our case we still get pretty good performance for the Kalman filter. However, we do see slight deviations in the curves (namely the first curve) which is probably due to deviation from constant speed in a different direction.

2) minimum values of maxE and RMSE achieved:

RMSE reached: 1.74

maxE reached : 5.87

we could possibly even have gotten better results trying sigma_n smaller than 1

3) In the animation we can see the Trajectory of GT and KF results and the estimate the trajectory based on the prediction without observation of after ~5 seconds (dead reckoning, Kalman gain=0): we can see that at the beginning of the animation the covariance is large corresponding to the large initialization and decreases as it starts converging to its variance of location. In addition, we see that when we set kalman gain to 0 the trajectory continues in a straight line this is because it depends now only on the motion model which is constant

velocity and not on the measurements hence we get a straight line in the direction that we were when we set K to zero.

V

Here we will plot and analyse the estimated x-y values separately and corresponded sigma value along the trajectory

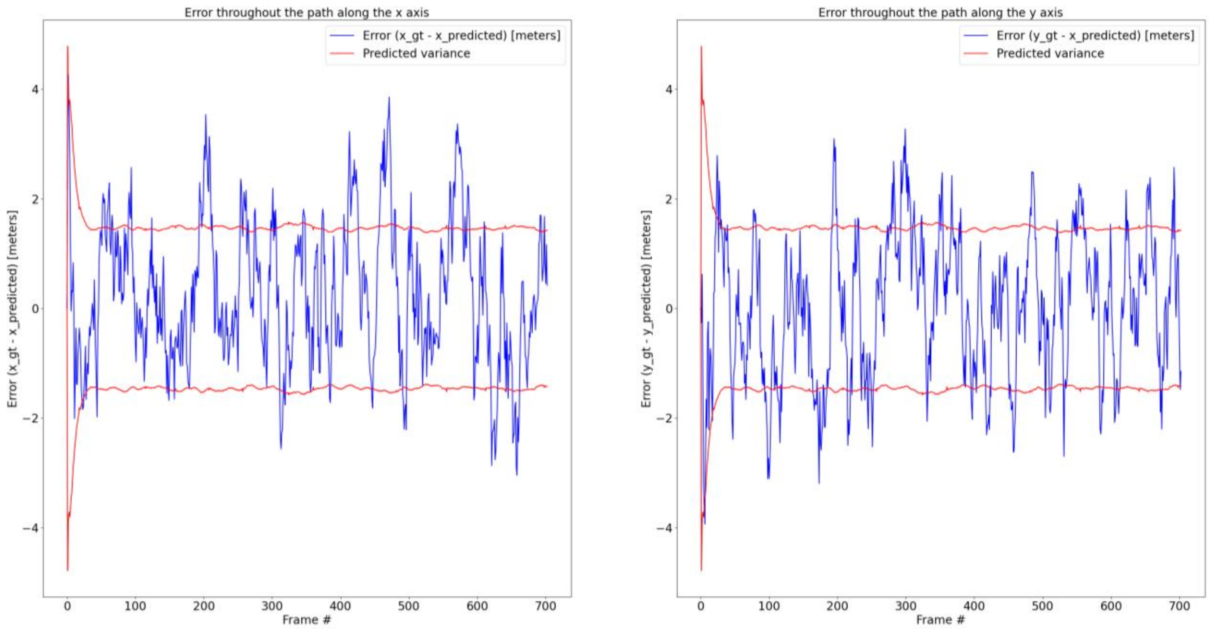

Figure 7 sigma errors

Here we can see that in both X and Y estimation errors get good performance, around 66% of the error is contained between the corresponding sigma meaning our selection of initialization process noise and measurement noise where correct. Possibly trying even smaller sigma_n could have given even better result because it mostly controls the converged sigma which could be a bit smaller. Moreover, we see that because where using a linear model the sigmas stay relatively linear (converge to straight lines) and don't change as much as will see in the EKF

Extended Kalman Filter

A-C

We will use same KITTI GPS/IMU sequence from last section.

we will extract vehicle GPS trajectory, yaw angle, yaw rate, and forward velocity from KITTI senser packets (OXT).

- lat: latitude of the oxts-unit (deg)

- lon: longitude of the oxts-unit (deg)

- yaw: heading (rad) - vf: forward velocity, i.e. parallel to earth-surface (m/s)

- wz: angular rate around z (rad/s) - Extract timestamps from KITTI data and convert them to seconds elapsed from the first one These are treated as GT in this experiment

We will now plot the ground truth of yaw angles, yaw rates, and forward velocities.

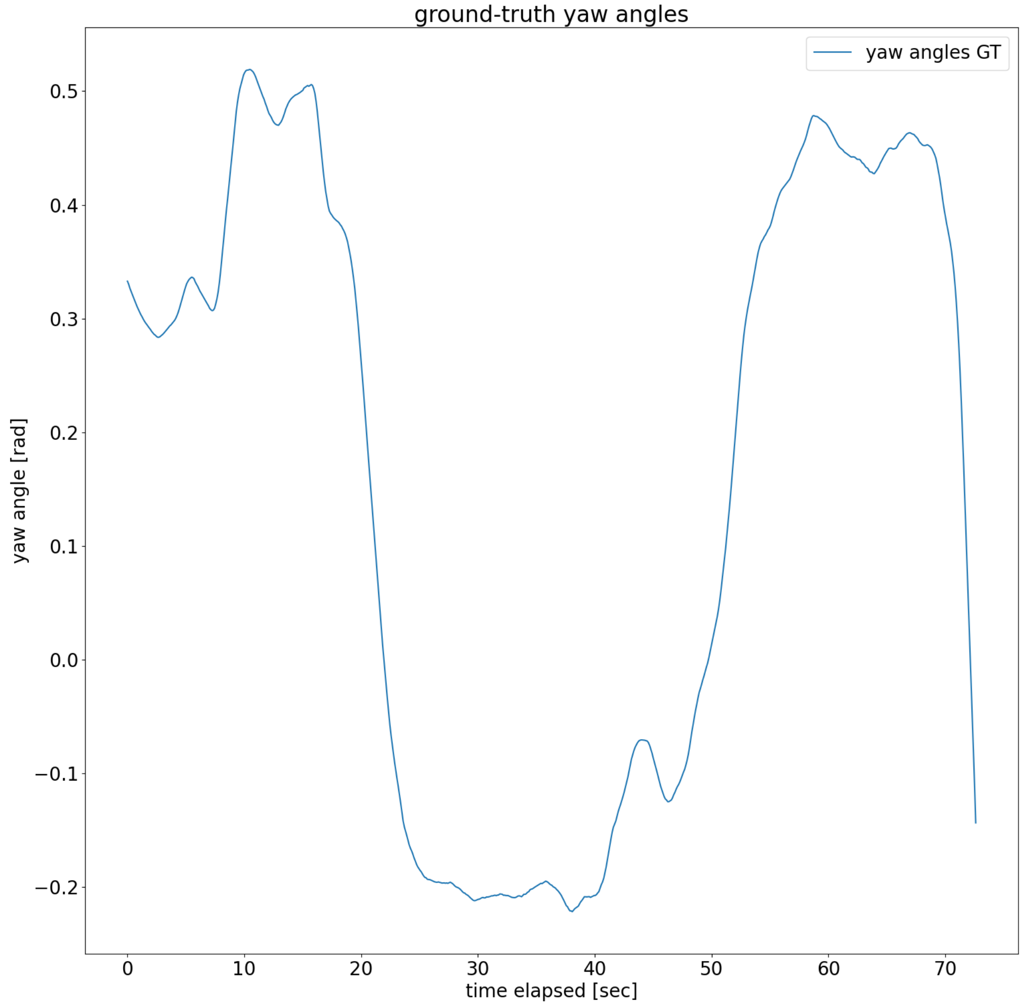

Figure 8 ground-truth yaw angles

We can see that for the most part the vehicle angle does not change much (biggest delta of ~0.7 degrees) which corresponds to the cars ENU graph that shows that for the most part it did not made any drastic changes in its path (no sharp turns)

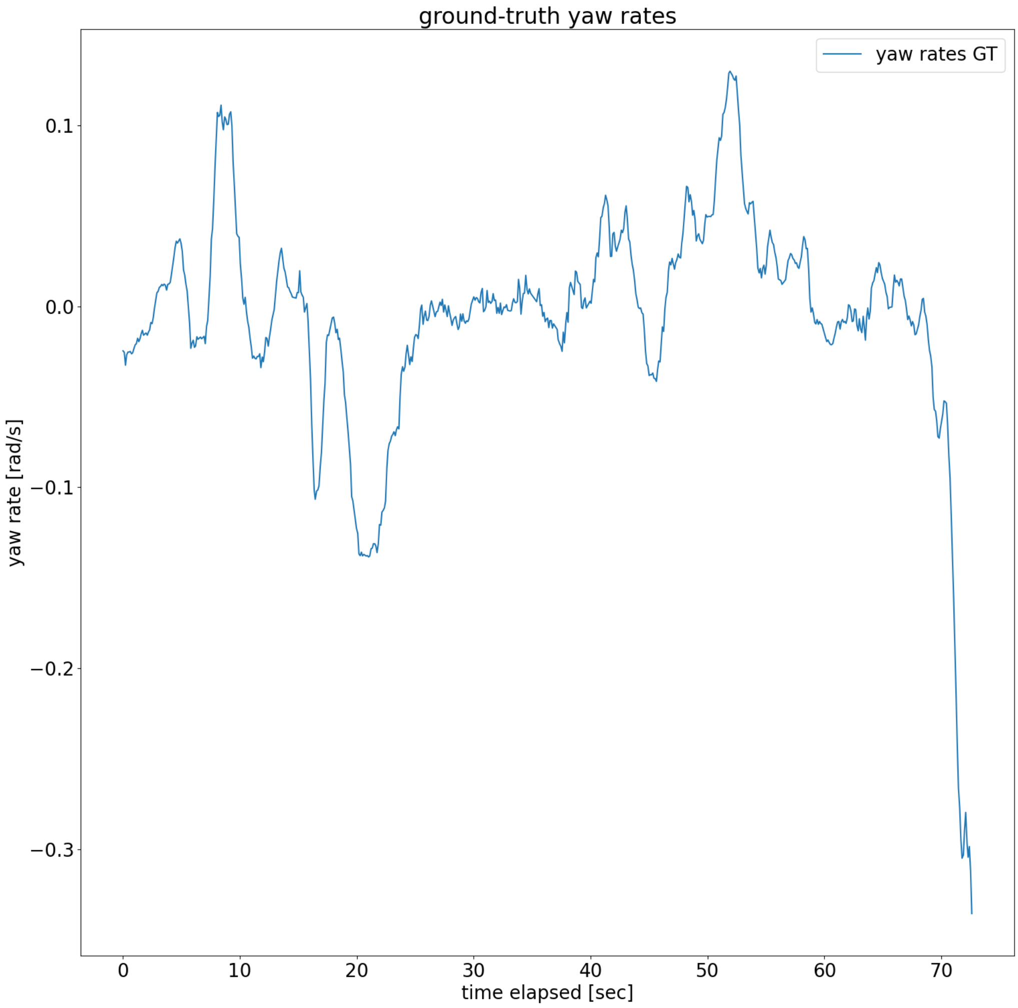

Figure 9 ground-truth yaw rates

Here we can see the change in angle change rate. As we can see for the most part it is quite static (with overall max delta of ~0.4 rad/sec) which corresponds to the ENU/LLA graphs where we can see that indeed there were no sharp turns in the graph.

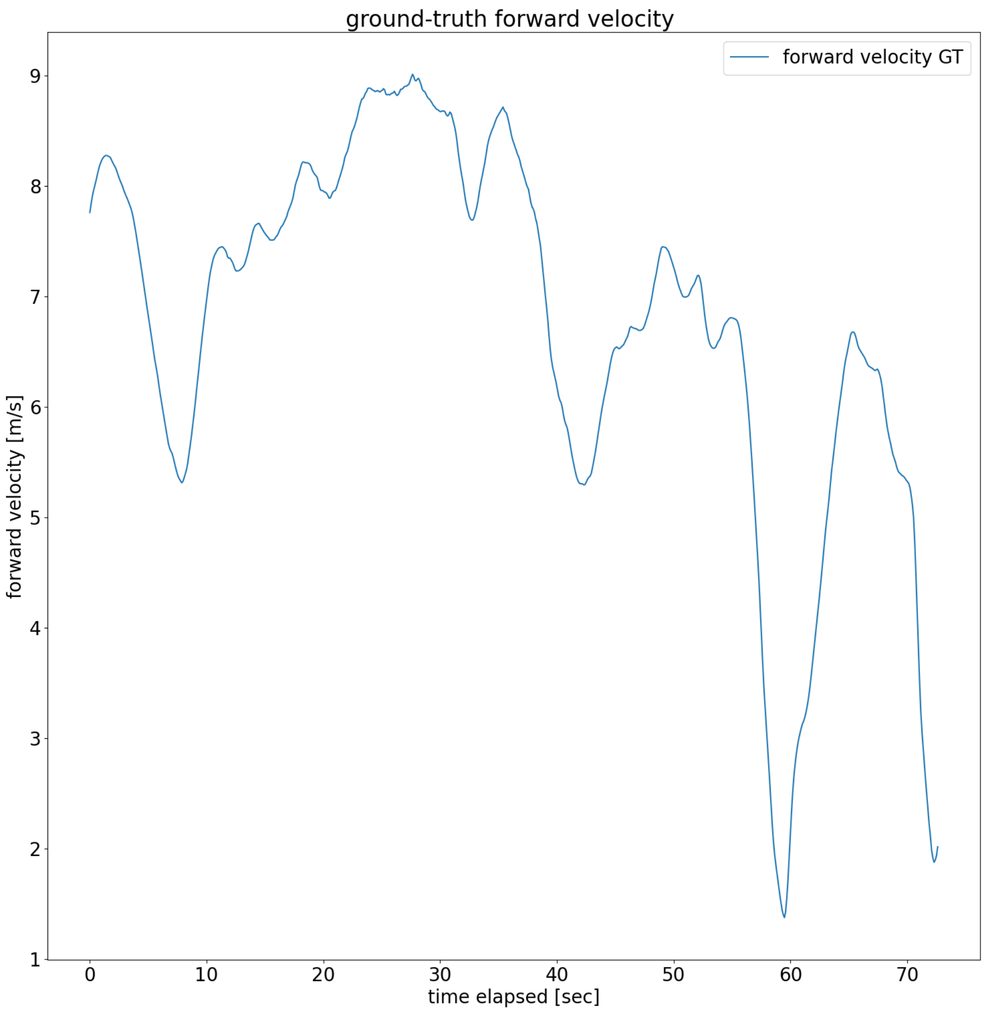

Figure 10 ground-truth forward velocity

Here we see that when driving straight the vehicle increased its speed and slowed down before the turns.

D-E

Here we added Gaussian noise to the ground-truth GPS data which will be used as noisy observations fed to the Extended Kalman filter. Noise added with standard deviation of observation noise of x and y in meter (σx = 3 , σy = 3).

Now we will apply an Extended Kalman filter to the GPS sequence in order to estimate the vehicle's 2D pose velocity-based model (non-linear model):

-we will suppose initial 2D position [x, y] estimation begins with the first GPS observation

- GPS observation noise of X and Y is known (σx =3, σy =3)

 -will Implement an EKF based on velocity-based model and compare results to the constants-velocity model (set the same initial conditions). In the next figure we can see EKF results with no noise in the commands

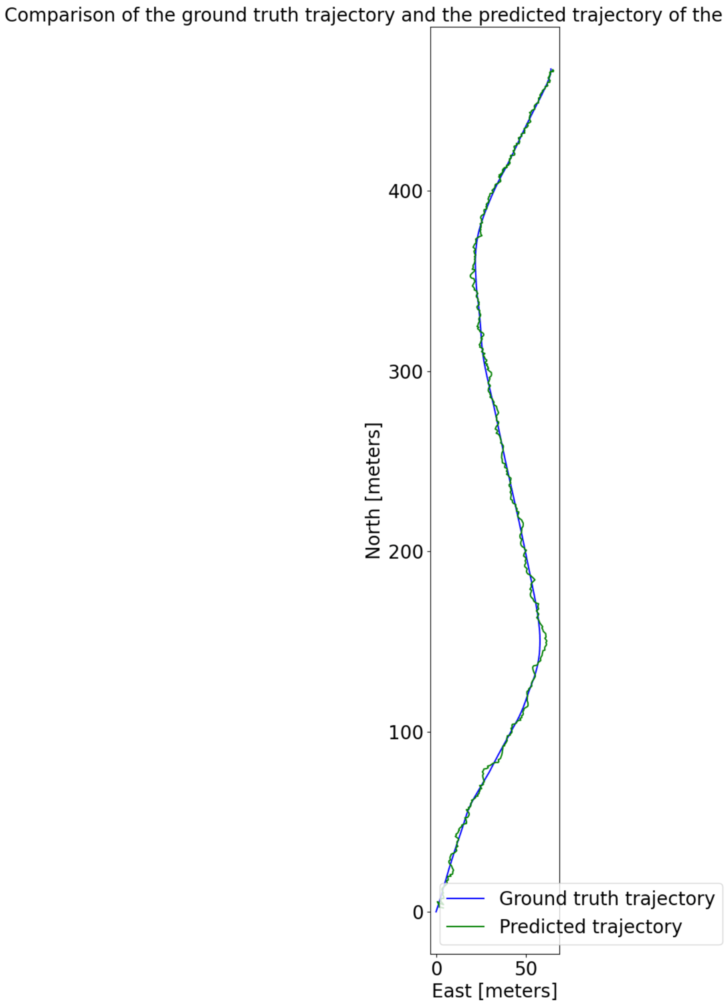

Figure 11 Comparison of the ground truth trajectory and the predicted trajectory of the EKF no noise in commands

In this case we can achieve better results compared to the constant velocity model especially if we look at the turns, when using the EKF on this data we used $\sigma_v = \sigma_w$ =0 since there is no noise added to this measurments. Moreover, in this case we got **RMSE = 1.18** and **maxE = 4.74** (compared to RMSE = 1.74 maxE = 5.87 in constant velocity model) on top of that we can see that the filter fitted itself quite good to the ground truth trajectory especially in all the turns that we can a smaller divergence than the one seen in the previous application of the regular Kalman Filter. This is because the model is much more dynamic and contains nonlinear components that more precisely model the real trajectory (these non linarites in the model are linearized so they will be able to use them in matrix form of Kalman filter and to uphold the Gaussian assumption).

F

Adding gaussian noise to the IMU data: Will add noise to yaw rates standard deviation of yaw rate in rad/s (σw = 0.2) and plot graphs of GT+ noise yaw rate:

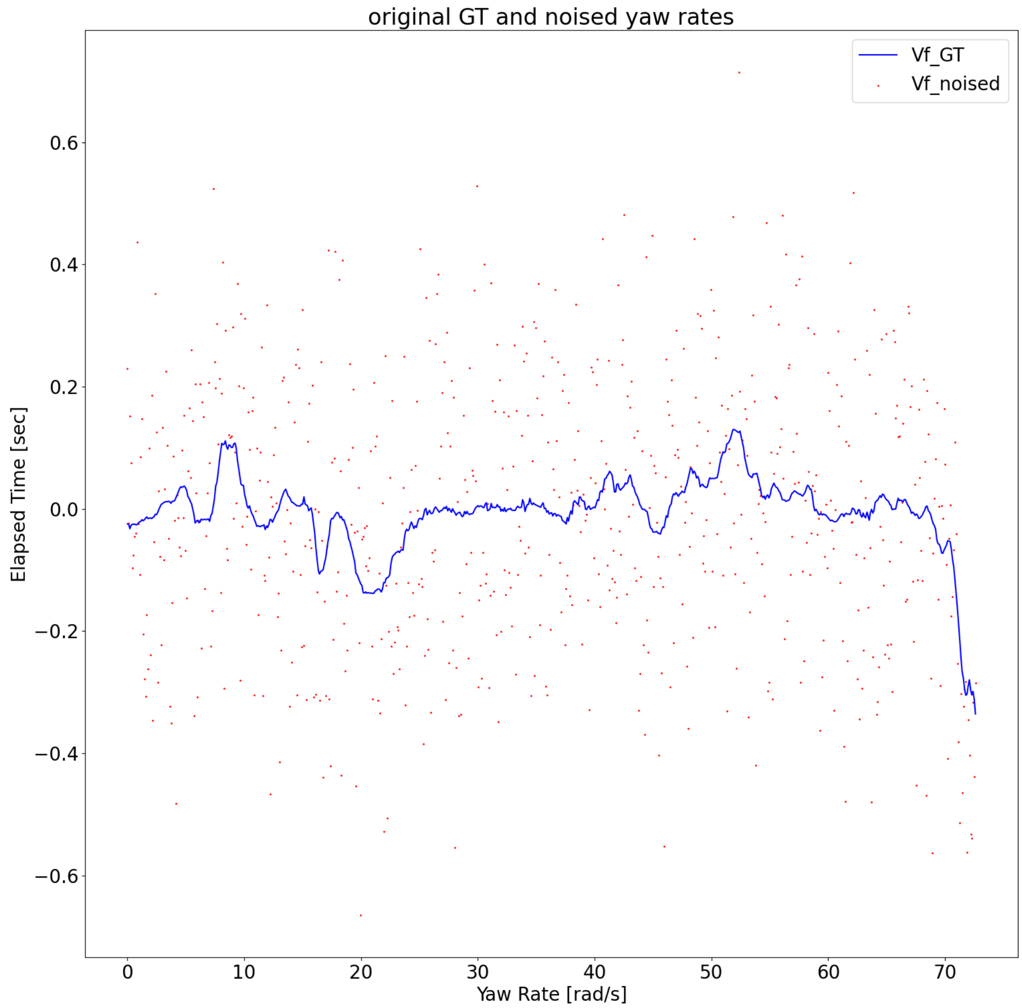

Figure 12 original GT and noised yaw rates

Since the Yaw rate values are pretty small we can see that adding white noise with σ_w= 0.2 has a big impact on the values.

Will Add noise to forward velocities adding standard deviation of forward velocity in m/s (σ_{vf} =2) plot graphs of GT+ noise velocities

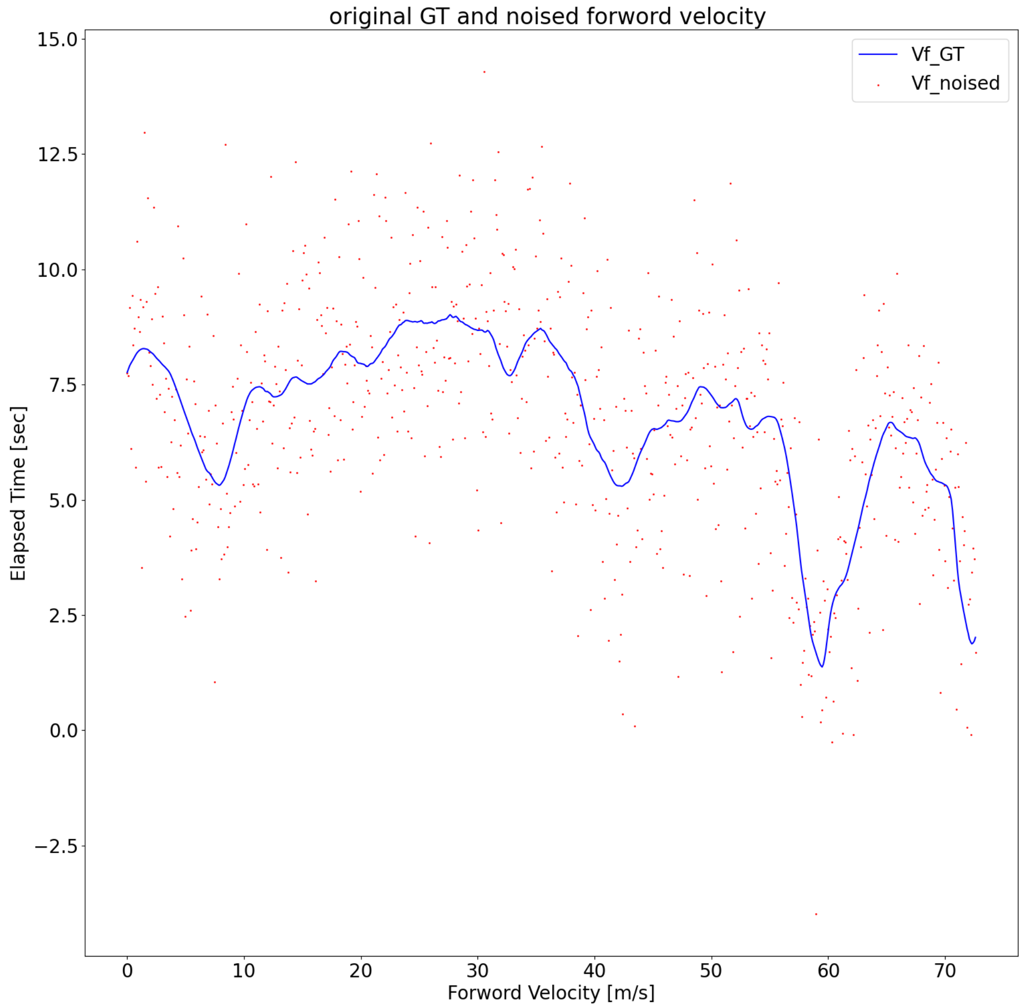

Figure 13 original GT and noised forword velocity

G

Our goal was to minimze RMSE while maxE <5:

Find which calibration has the best performance according to the above criteria.

1) Initial conditions: according to first observation

$$x_{initial} = \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \end{bmatrix}$$

We initialized the mean of our state to be our first observation and the assumed angel of heading north pi/2

$$P_0 = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\emptyset^2 \end{bmatrix} * K$$

These values are chosen as such because they are linearized with their Jacobean $V_t$to fit the uncertainty of x y and theta at the at the beginning of the path.

Where the value of K was set to be 2.0 although empirical tests showed that due to the convergence f the EKF the initial values of P (sigma) do not affect the EKF results as it converge rather fast

2) 2) Jacobians G, V and C:

$$G_t = \begin{bmatrix} \frac{dg_1}{dx_1} & \frac{dg_1}{dy_1} & \frac{dg_1}{d\theta_1} \\ \frac{dg_2}{dx_1} & \frac{dg_2}{dy_1} & \frac{dg_2}{d\theta_1} \\ \frac{dg_3}{dx_1} & \frac{dg_3}{dy_1} & \frac{dg_3}{d\theta_1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{v_t}{w_t}\cos(\theta_{t-1}) + \frac{v_t}{w_t}\cos(\theta_{t-1} + w_t\Delta t) \\ 0 & 1 & -\frac{v_t}{w_t}\sin(\theta_{t-1}) + \frac{v_t}{w_t}\sin(\theta_{t-1} + w_t\Delta t) \\ 0 & 0 & 1 \end{bmatrix}$$

$$V_t = \begin{bmatrix} \frac{dg_1}{dv_1} & \frac{dg_1}{dw_1} \\ \frac{dg_2}{dv_1} & \frac{dg_2}{dw_1} \\ \frac{dg_3}{dv_1} & \frac{dg_3}{dw_1} \end{bmatrix} =$$

$$= \begin{bmatrix} -\frac{1}{w_t}\sin(\theta_{t-1}) + \frac{1}{w_t}\sin(\theta_{t-1} + w_t\Delta t) & \frac{v_t}{w_t^2}\sin(\theta_{t-1}) - \frac{v_t}{w_t^2}\sin(\theta_{t-1} + w_t\Delta t) + \frac{v_t}{w_t}\cos(\theta_{t-1} + w_t\Delta t)\Delta t \\ \frac{1}{w_t}\cos(\theta_{t-1}) - \frac{1}{w_t}\cos(\theta_{t-1} + w_t\Delta t) & -\frac{v_t}{w_t^2}\cos(\theta_{t-1}) + \frac{v_t}{w_t^2}\cos(\theta_{t-1} + w_t\Delta t) + \frac{v_t}{w_t}\sin(\theta_{t-1} + w_t\Delta t)\Delta t \\ 0 & \Delta t \end{bmatrix}$$

$$H_t = \begin{bmatrix} \frac{dh_1}{dx_1} & \frac{dh_1}{dy_1} & \frac{dh_1}{d\theta_1} \\ \frac{dh_2}{dx_1} & \frac{dh_2}{dy_1} & \frac{dh_2}{d\theta_1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

These Jacobeans linearize the nonlinear model around our state vector so they will be able to be fed into the kalman filter and still retain the Gaussian assumption.

3) Covariance (Q and R):

$$\tilde{R}_t = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_w^2 \end{bmatrix}$$

$$Q_t = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}$$

$$R_t = V_t \widetilde{R_t} V_t + R_n$$

$$R_n = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \sigma_n \ , \sigma_n \ = 0.01 \gg \mathit{gives\ best\ results\ acourding\ to\ impirical\ tests}$$

We have seen empirically that in general adding this specific R_n to our predicted estimate yielded better than the results (in terms of maxE and RMSE) without adding this noise

EKF main routine:

We preformed the Extended Kalman Filter as learned in class:

Extended_Kalman_filter$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:

$\bar{\mu}_t = g(u_t, \mu_{t-1})$

$\bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^T + R_t$

$K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$

$\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$

$\Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$

return μ_t, Σ_t

step 1: we predict our mean location based on our nonlinear model (contained in non linear motion model: g)

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{bmatrix} + \begin{bmatrix} -\frac{v_t}{\omega_t} sin\theta_{t-1} + \frac{v_t}{\omega_t} sin\left(\theta_{t-1} + \omega_t \Delta t\right) \\ \frac{v_t}{\omega_t} cos\theta_{t-1} - \frac{v_t}{\omega_t} cos\left(\theta_{t-1} + \omega_t \Delta t\right) \\ \omega_t \Delta t \end{bmatrix}$$

step 2: we predict how our uncertainty carries on to the next time step by our point linearized motion (of nonlinear model) calculated in our Jacobean of "g" (meaning G_t) and process noise R which contains the process noise transformed by V and general process noise added to contain the angle in the sigma 1.

step 3: we compute our Kalman gain which controls the emphasis on the deviation between what we predicted and what was measured. Where H_t maps our predicted state to the observed state.

Step 4: computes the corrected mean based on the kalman gain computed and deviation of expected location and observation.

Step 5: computes the corrected covariance matrix also based on the kalman gain.

because our model is non linear all transformations most be point state linearized by the jacobian so the assumption of Gaussian distributions will hold throw all transforms.

H

Results analysis:

I. Ground-truth and estimated results:

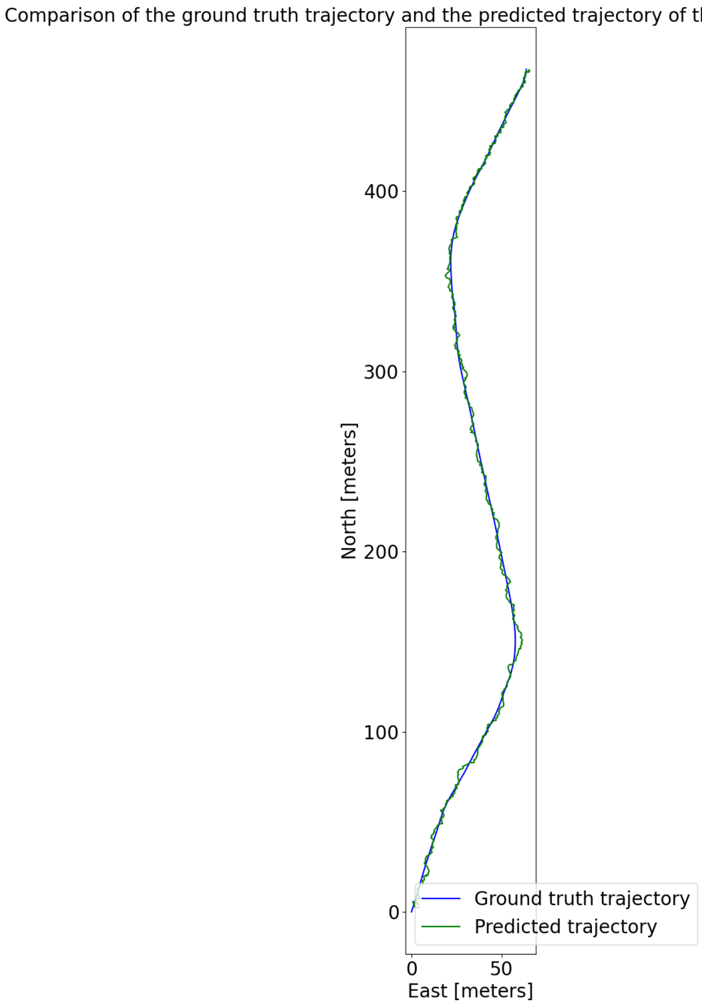

Figure 14 Comparison of the ground truth trajectory and the predicted trajectory of the EKF_with_vf_wz_noise

II. Show Kalman filter performance: We can see that even with noise in location and commands (this time we also accounted for the added noise in our model, namely $\sigma_v = 2.0$ σ_w =0.2) and managed to receive a relatively good estimate of the GT and a less noisy trajectory compared with regular Kalman Filter, we can that the biggest divergence between the ground truth and the trajectory is happening in the first curve which is to be expected as it is harder to fit the motion model when there are big changes applied to the trajectory, as such the turns are the weak point of the Kalman Filters but we can also see that as the algorithm continue to fit itself with the car movement and even on the second curve we get relatively low divergence between the ground truth and the predicted trajectory .

III. The minimum values of maxE and RMSE achieved after running different values of sigma_n and finding the one that will give the best results :
RMSE = 1.39 , maxE = 4.8

Hence even with noise in the commands and measurements we still get better results than the regular Kalman filter.

IV. In the animation we can see the Trajectory of GT and EKF results and the estimate the trajectory based on the prediction without observation of after ~5 seconds (dead reckoning, Kalman gain=0): We can see in the animation that indeed our predicted trajectory is very close to the ground truth trajectory, and again as mentioned before the biggest seen divergence is in the first curve. In addition, we see that when we set Kalman gain to 0 the trajectory becomes skew this is because we are depending only on our motion model (which is nonlinear) and not on the measurements hence we get a trajectory that is not aligned with the observations but is still similar in way to the original trajectory and very different than the dead reckoning seen in the regular linear Kalman Filter where it is seen as a straight line.

V. Plot and analyse the estimated x-y-θ values separately and corresponded sigma value along the trajectory:

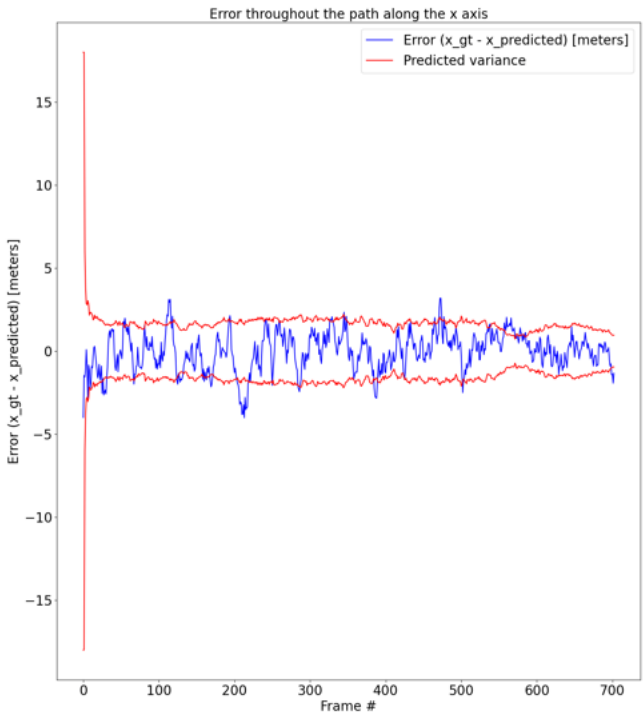

Figure 15 sigma error x

We can see that more then 66% of the errors are contained in the estimated 1 sigma interval and that the covariance is dynamic decaying when there a small difference between observations and predictions.

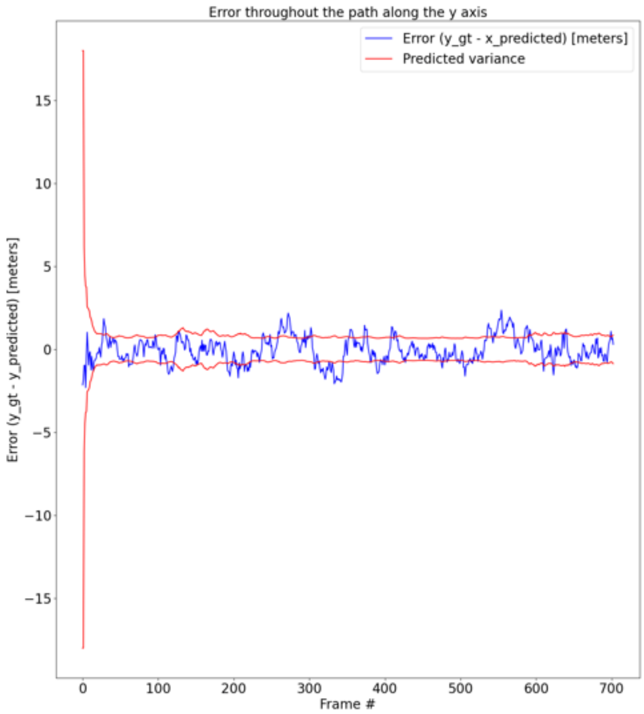

Figure 16 sigma error y

Again we see that more then 66% error is contained within the 1 sigma interval and how the sigma is dynamic

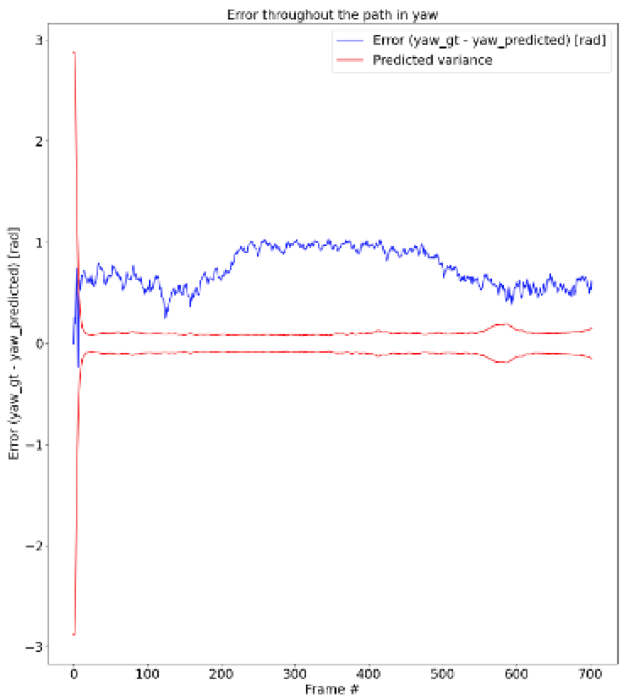

Figure 17 sigma error yaw

Here we see that there is a very big difference between the predicted yaw and the actual yaw. We believe that this big of a divergence is due to the fact that for yaw we do not get any real measurement and thus cannot really predict correct its location, but still, we can see that the difference between the ground truth and the actual yaw location is indeed rather small. however, adding more variance to the theta (via sigma_n) might be able to contain a bit more of the error inside the sigma's and give better representation of our uncertainties regarding the yaw.

EKF-SLAM

A

Here we load attached inputs and code Python files.

- Landmarks location

- Odometry and sensor data

- filled in missing parts inside the attached code.

B

Here we will run Odometry data according to odometry model and plot the GT trajectory.

$$\begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{pmatrix} + \begin{pmatrix} \delta_{trans} \cos(\theta_{t-1} + \delta_{rot1}) \\ \delta_{trans} \sin(\theta_{t-1} + \delta_{rot1}) \\ \delta_{rot1} + \delta_{rot2} \end{pmatrix}$$

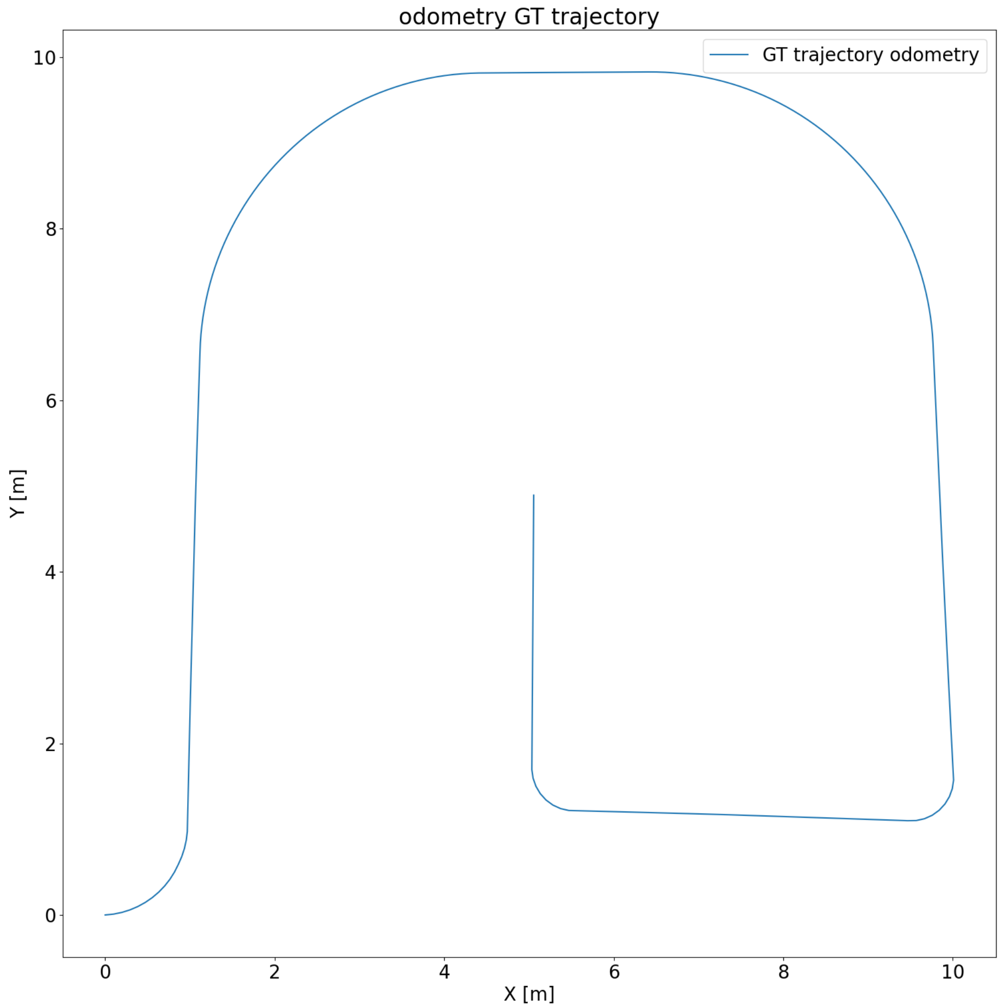

Figure 18 odometry GT trajectory

We start from 0,0 heeding east and start to turn north then wrap around to get to center.

C

Here we added Gaussian noise in the motion model assume ($\sigma_{\mathrm{rot1}} = 0.01, \sigma_{\mathrm{trans}} = 0.1, \sigma_{\mathrm{rot2}} = 0.01$).

Now we will Apply Extended Kalman SLAM filter: The goal: minimize RMSE while maxE < 1.5:

$$RMSE \triangleq \sqrt{\frac{1}{N-20}\sum_{i=20}^{N}[e_x^2(i) + e_x^2(i)}$$

$$e_x(i) \triangleq x_{GT} - x_{Estimate}$$

$$e_y(i) \triangleq y_{GT} - y_{Estimate}$$

$$maxE \triangleq \max\{|e_x(i)| + |e_y(i)|\}$$

D

Initialize initial conditions μ_0, Σ_0:

$$\mu_0 = \begin{bmatrix} x_0 \\ y_0 \\ \emptyset_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$, 2N +3 dimensions

Containing the mean of the pose and landmark locations [x,y : for each landmark hence 2N] and pose = [0,0,0]

$$\Sigma_0 = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_y^2 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_\phi^2 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 100 & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots & 0 & \vdots \\ 0 & 0 & 0 & 0 & & 100 \end{bmatrix}$$, inf is set to 100

sigma_x_y_theta = [3,3, 1]

all landmarks are set to uncertainty of infinity because they have not yet been observed

E

Here we Implement the prediction step of the EKF SLAM algorithm in the function "predict" Use the odometry motion model. We compute the predicted mean:

$$F_x = \begin{pmatrix} 1 & 0 & 0 & 0\cdots 0 \\ 0 & 1 & 0 & 0\cdots 0 \\ 0 & 0 & 1 & 0\cdots 0 \end{pmatrix}$$

$$\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} + \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \omega_t\Delta t \end{pmatrix}$$

Where: Meaning the new predicted mean is the old predicted mean plus the motion step timed by F_x so it won't affect the land marks

F

We compute its Jacobian $Gt\ x$ to construct the full Jacobian matrix Gt :

$$G_t^x = I + \begin{pmatrix} 0 & 0 & -\delta_{trans}\ \sin(\theta_{t-1} + \delta_{rot1}) \\ 0 & 0 & \delta_{trans}\ \cos(\theta_{t-1} + \delta_{rot1}) \\ 0 & 0 & 0 \end{pmatrix}$$

$$G_t = \begin{pmatrix} G_t^x & 0 \\ 0 & I \end{pmatrix}$$

G_t is the Jacobean of the nonlinear motion model g with respect to u_{t-1}, that impacts only the pose uncertainty's

G

We compute its Jacobian V to construct the full Jacobian matrix GR_t^x and R_t

$$V_t = \begin{bmatrix} \frac{dg_1}{dx_1} & \frac{dg_1}{dy_1} & \frac{dg_1}{d\theta_1} \\ \frac{dg_2}{dx_1} & \frac{dg_2}{dy_1} & \frac{dg_2}{d\theta_1} \\ \frac{dg_3}{dx_1} & \frac{dg_3}{dy_1} & \frac{dg_3}{d\theta_1} \end{bmatrix} = \begin{bmatrix} -\delta_{trans}\sin(\theta + \delta_{rot1}) & \cos(\theta + \delta_{rot1}) & 0 \\ \delta_{trans}\cos(\theta + \delta_{rot1}) & \sin(\theta + \delta_{rot1}) & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$\tilde{R} = \begin{bmatrix} \sigma_{rot1}^2 & 0 & 0 \\ 0 & \sigma_{trans}^2 & 0 \\ 0 & 0 & \sigma_{rot2}^2 \end{bmatrix}$$

$$R_t^x = V_t \tilde{R}_t V_t^T + R_n$$

$$R_n = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \sigma_n, \qquad \sigma_n = 0.1$$

$$R_t = F_x^T R_t^x F_x$$

Where V_t is the Jacobean local linearization of g with respect to u_t also impact the pose only.

The whole prediction step impacts the mean pose and pose uncertainty and does not effect the landmarks.

R_t is the process noise from commands.

R_n is general process noise and after applying many empirical test it was set as seen above and with $\sigma_n = 0.1$ which again gave us the best results in terms of RMSE and maxE.

To get full uncertainty covariance we compute :

$$\bar{\Sigma} = G_t \Sigma_{t-1} G_t + R_t$$

H

Here we Implement the correction step in the function "update": The argument $\boldsymbol{z}$ of this function is a struct array containing m landmark observations made at time step t. Each observation z(i) has an id z(i).id, a range z(i).range, and a bearing z(i). bearing. We will Iterate over all measurements (i= 1…m) and compute the Jacobian H_t^i

We compute a block Jacobian matrix H_t by stacking the H_t^i matrices corresponding to the individual measurements. The landmark measurement is:

$$z_t^i = (r_t^i, \phi_t^i)^T$$

The update to the predicted measurement if it is already initialized is:

$$\delta = \begin{pmatrix} \delta_x \\ \delta_y \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{j,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{j,y} - \bar{\mu}_{t,y} \end{pmatrix}$$

If landmark hasn't been seen before it is initialized to:

$$\begin{pmatrix} \bar{\mu}_{j,x} \\ \bar{\mu}_{j,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \end{pmatrix} + \begin{pmatrix} r_t^i \ \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ r_t^i \ \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \end{pmatrix}$$

The expected observation is then:

$$\hat{z}_t^i = \begin{pmatrix} \sqrt{q} \\ \operatorname{atan2}(\delta_y, \delta_x) - \bar{\mu}_{t,\theta} \end{pmatrix}$$

When q is defined as:

$$q = \delta^T \delta$$

In order to treat each landmark separately we use vector $F_{x,j}$to decouple them which is defined:

$$F_{x,j} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & \underbrace{0 \cdots 0}_{2j-2} & 0 & 1 & \underbrace{0 \cdots 0}_{2N-2j} \end{pmatrix}$$

H_t^i Is then:

$$H_t^i = \frac{1}{q} \begin{pmatrix} -\sqrt{q}\delta_x & -\sqrt{q}\delta_y & 0 & +\sqrt{q}\delta_x & \sqrt{q}\delta_y \\ \delta_y & -\delta_x & -q & -\delta_y & +\delta_x \end{pmatrix} F_{x,j}$$

In general, we are computing the difference between the predicted location and the assumed location of the landmark to achieve the predicted measurement and the Jacobean of the observation with respect to the mean prediction. The Kalman gain is then:

$$K_t^i = \bar{\Sigma}_t \ H_t^{iT} (H_t^i \ \bar{\Sigma}_t \ H_t^{iT} + Q_t)^{-1}$$

While Q_t is the noise in the sensor model is a diagonal matrix with alternating values on its diagonal of σ_r = 0.3 , σ_θ = 0.0335.

The corrected mean and uncertainty matrix is no calculated by:

$$\bar{\mu}_t = \bar{\mu}_t + K_t^i (z_t^i - \hat{z}_t^i)$$
$$\bar{\Sigma}_t = (I - K_t^i \; H_t^i) \; \bar{\Sigma}_t$$

This is done for each land mark that has been observed in that time step.

1) Analyze results:
    1) Here will show the trajectory of EKF-SLAM results.

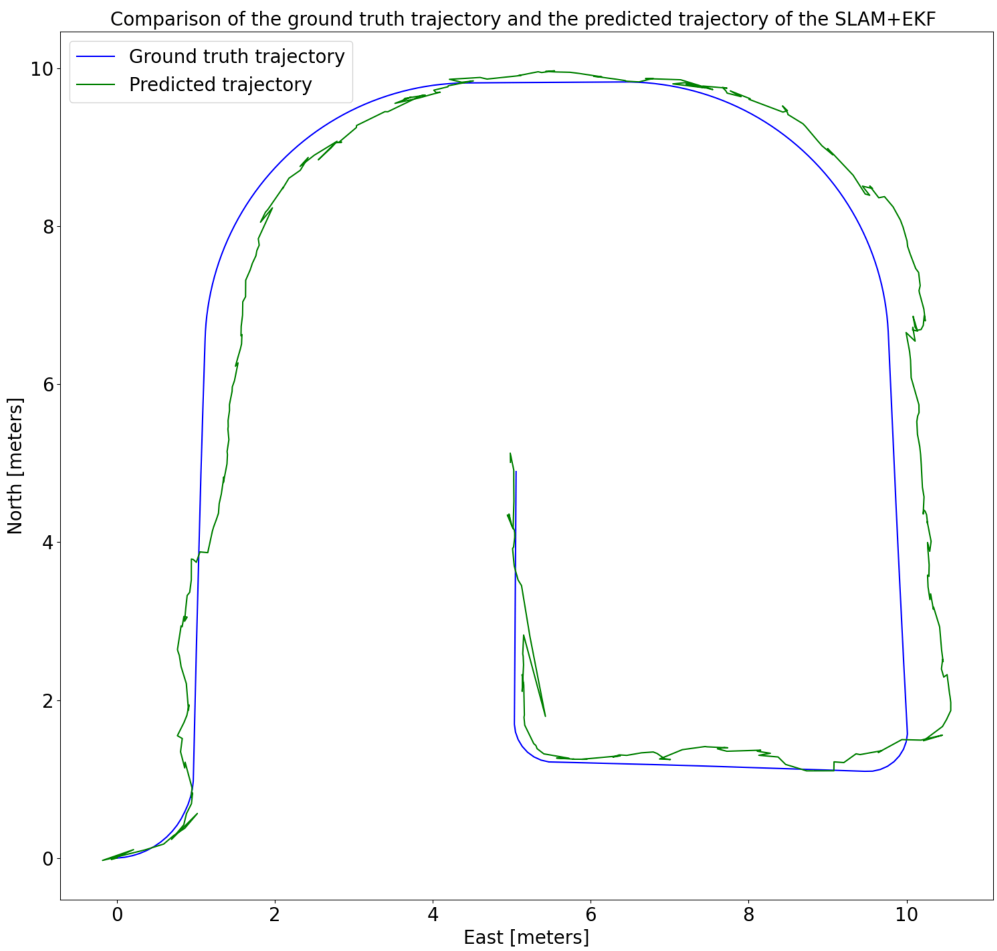

Figure 19 Comparison of the ground truth trajectory and the predicted trajectory of the SLAM+EKF

We can see from the above image that overall, we managed to predict the trajectory rather good, we can see we have slight drift in our predicted trajectory, especially near the sharp turns (namely the sharp turn to the left) , but overall we can see that we got pretty good results.

On top of the above figure, we have also added an animation, plot covariance matrix of state vector as ellipse. We can see in the animation that the covariance matrix of the pose starts out large and goes down as it recognizes its position from the landmarks the landmarks covariance also goes down as it matches the expected observation. Moreover we can see that the predicted location of each of the landmarks is close to the ground truth landmark.
the minimum values of maxE and RMSE achieved is:
RMSE 0.496
maxE 1.36
We have noticed during the calibration of the algorithm that the added noise to the R_n had been very beneficial and helped the model gain better results, as seen above.

2) Analyse estimation error of X, Y and Theta:

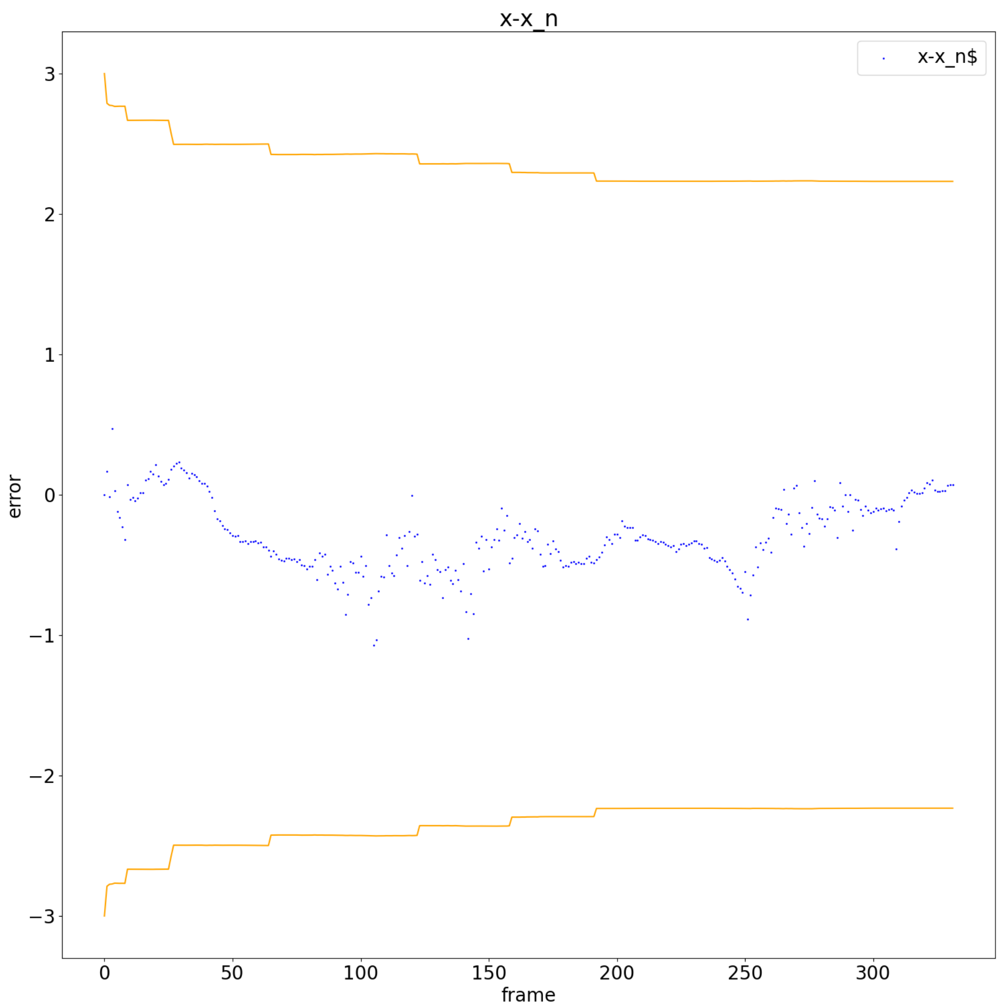

Figure 20 x-x_n

We can see that more than 66% is in the 1 sigma hence the uncertainty could have been smaller however we can see that the error is quite small.

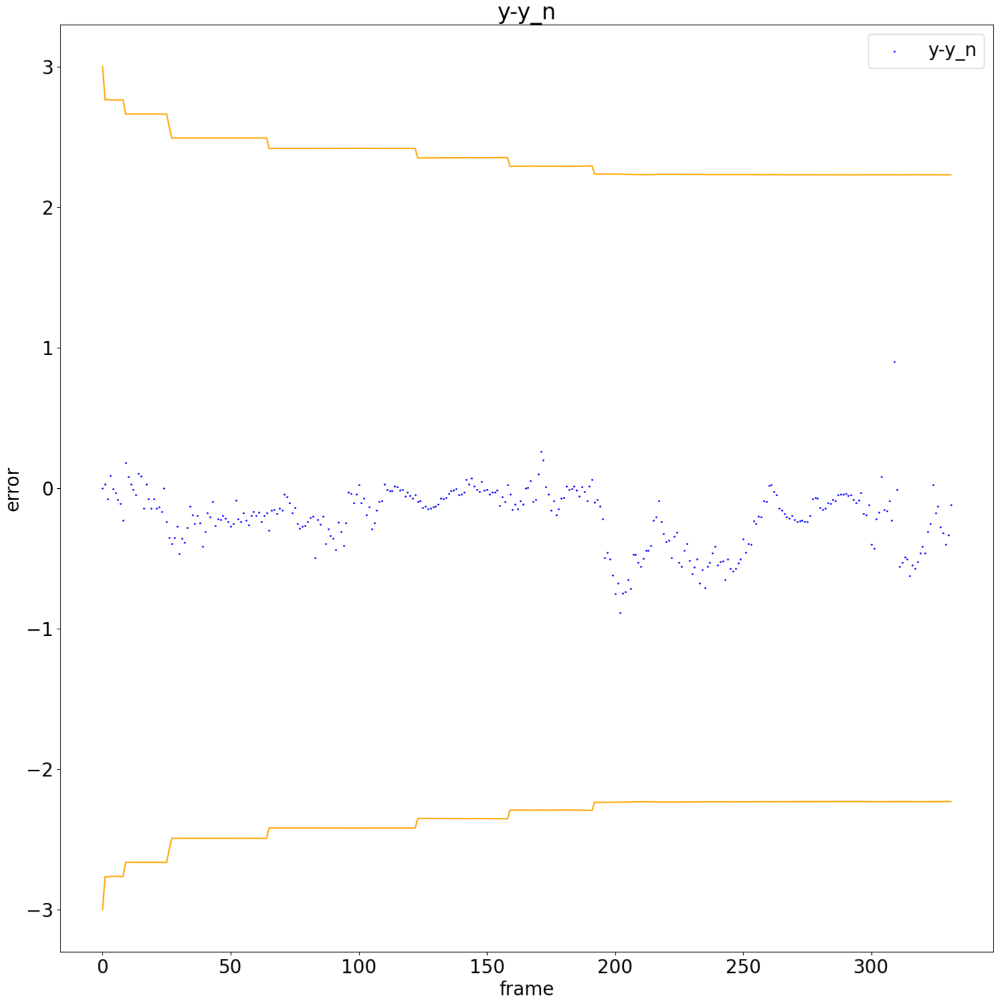

Figure 21y -y_n

Again we can see that more than 66% is in the 1 sigma hence the uncertainty could have been smaller however we can see that the error is still quite small.

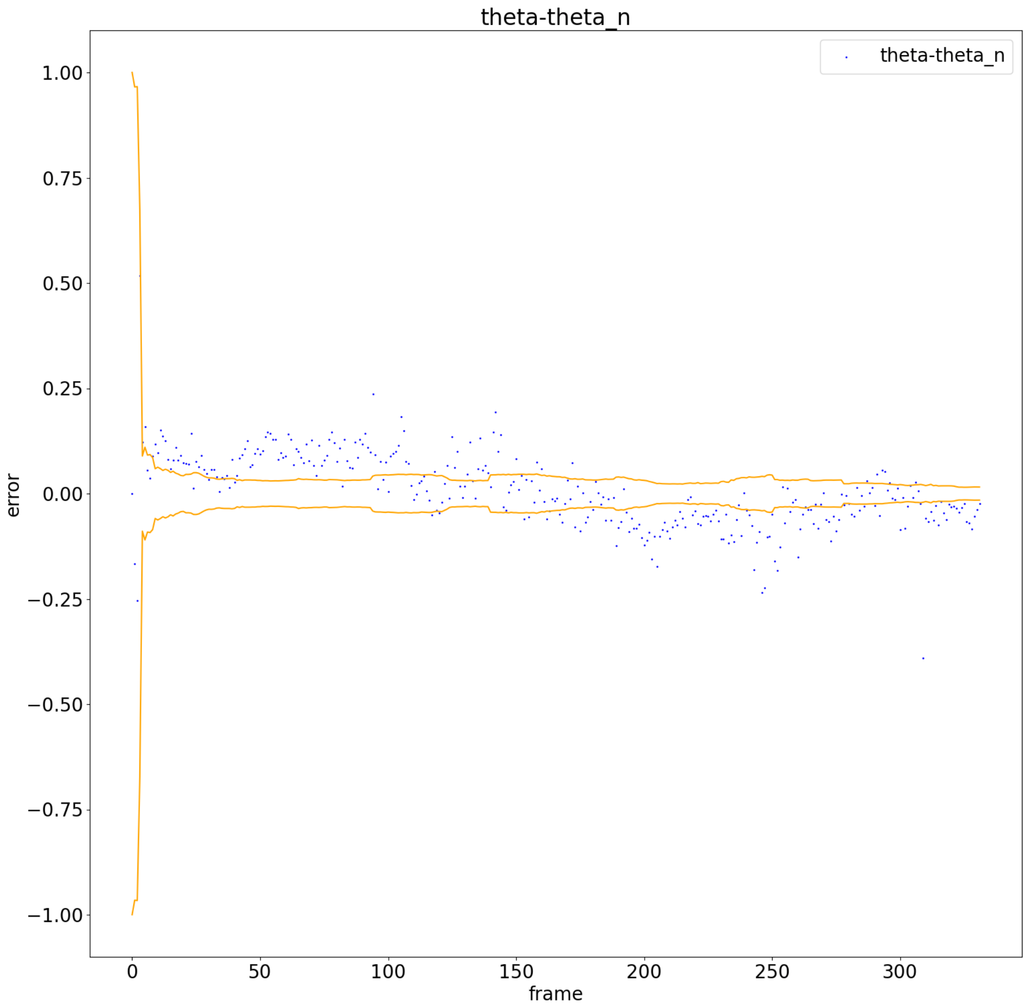

Figure 22 theta-theta_n

Here we can see that our uncertainty was too small when dealing with the angle perhaps adding additional uncertainty to the angle would have gotten better results, however we do see that the error itself is quite small.

3) Here we picked 2 landmarks and analysed them:

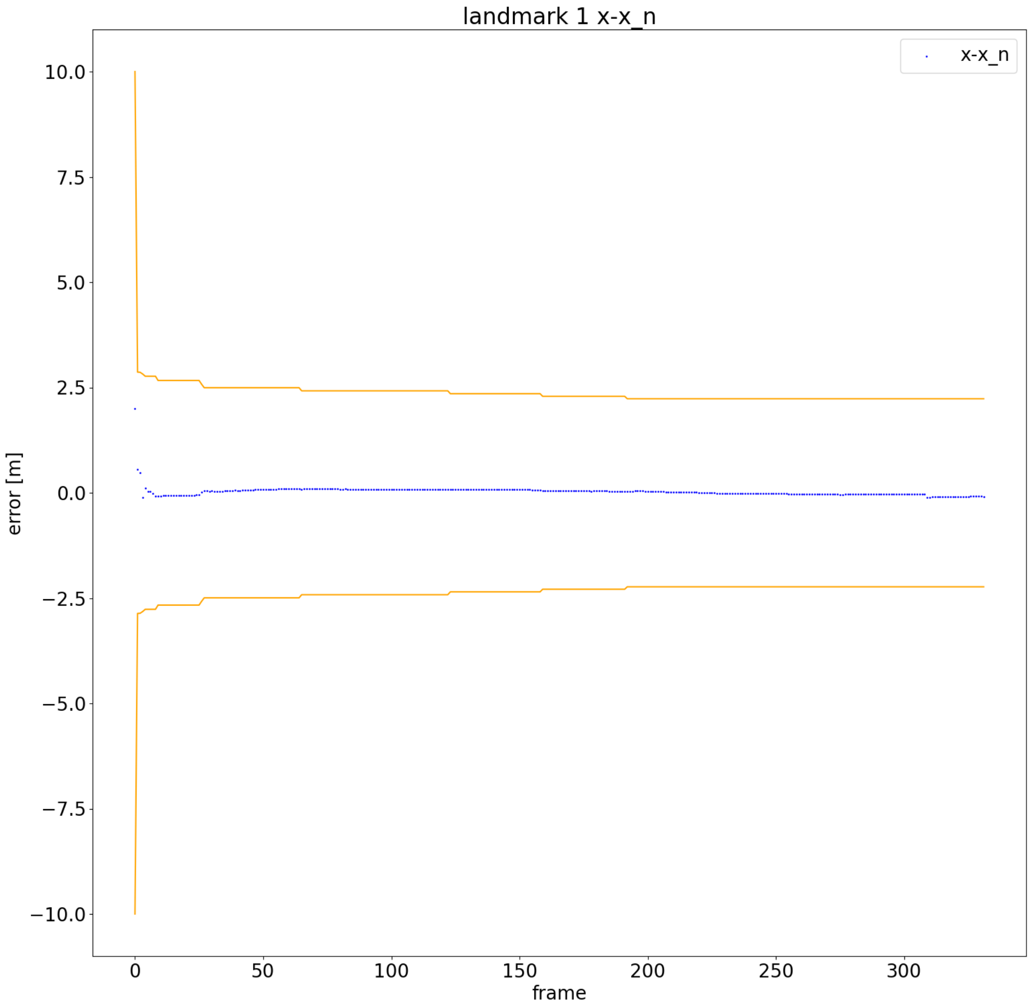

Figure 23 landmark_1_x-x_n

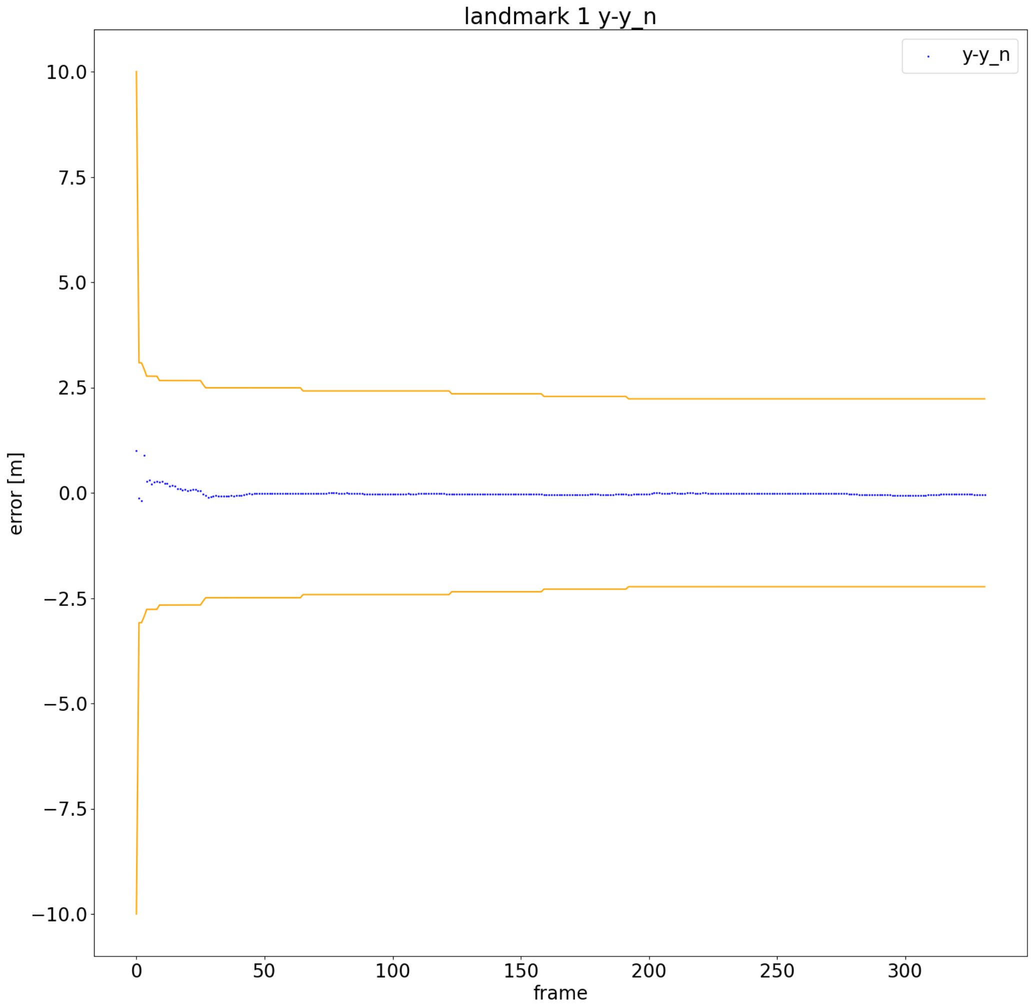

Figure 24 landmark_1_y-y_n

Here we see that both x and y errors are 100% contained meaning there covariance's could have been smaller, more over you can see that the uncertainty converges to the uncertainty of the pose and that the error itself in both the x and the y is very small.

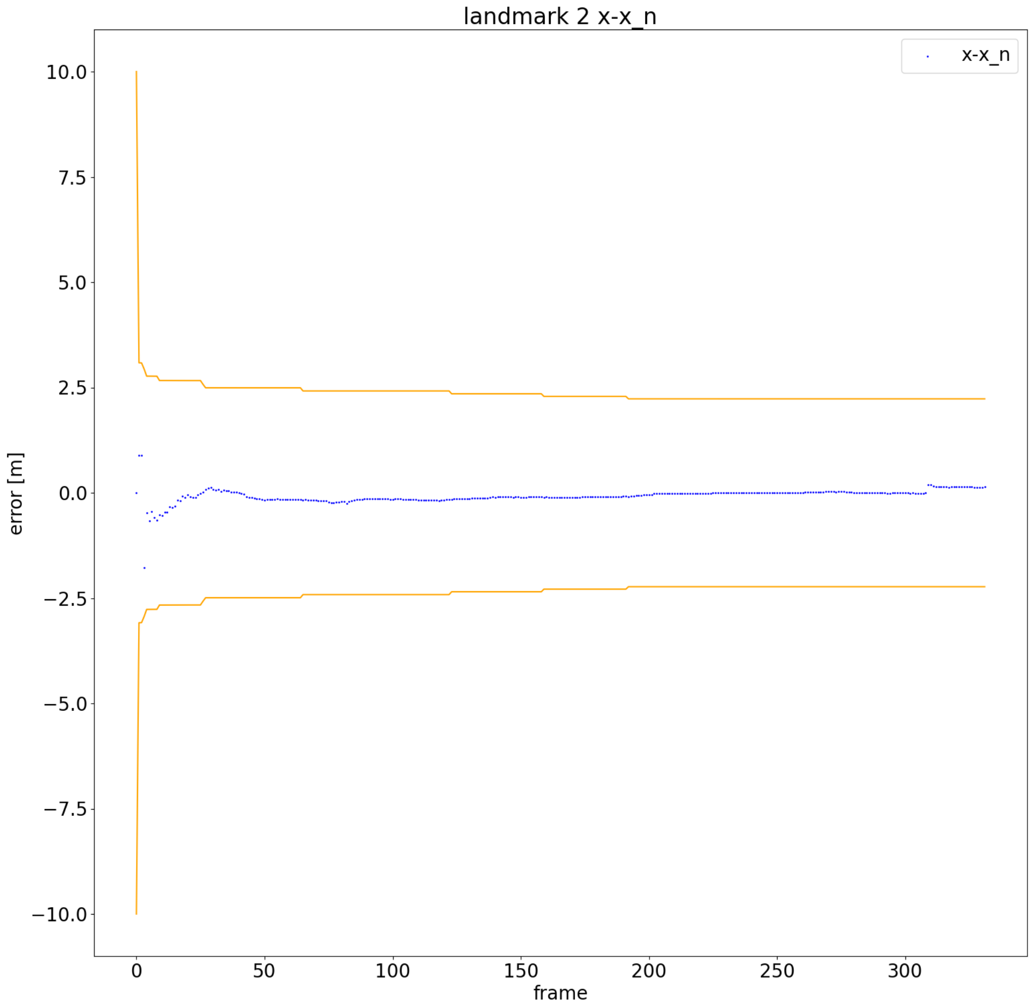

Figure 25 landmark_2_x-x_n

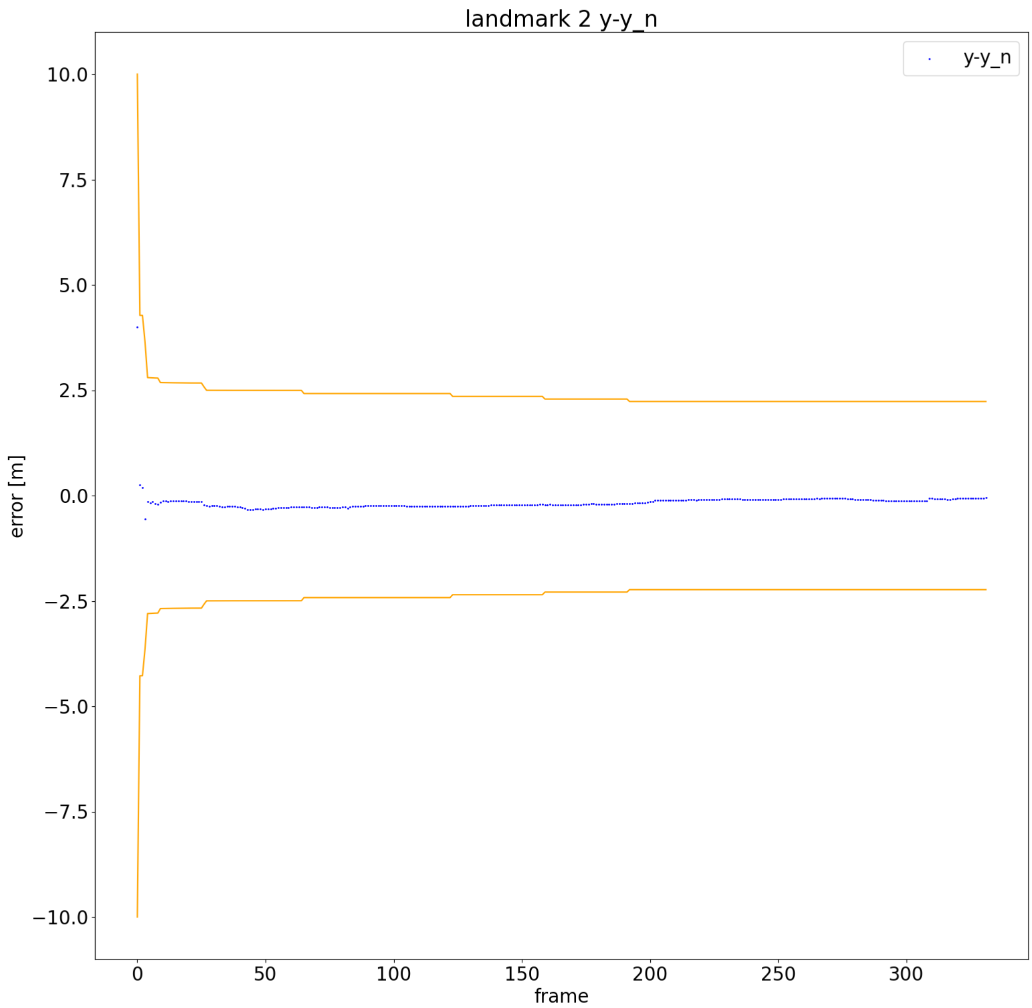

Figure 26 landmark_2_y-y_n

Here we can see again that the uncertainty could have been smaller since the error between the estimated location the real location is very close to 0, on x we can see that in the beginning the error was larger and then is reduced until the last “jump” in the end, this corresponds with the drift seen in the comparison of the ground truth trajectory and the estimated trajectory.

Summary

In this project, we conducted an implementation of the Kalman Filter(KF) and the Extended Kalman Filter (EKF) for estimating the trajectory of a moving vehicle using noisy GPS measurements from the KITTI dataset. Through this implementation, we observed that despite the presence of noisy inputs, the filters were able to provide a reasonably accurate estimation of the ground truth trajectory.

Furthermore, we extended the EKF framework to incorporate Simultaneous Localization and Mapping (SLAM) capabilities. By leveraging the odometry model and noisy measurements of landmarks, we simultaneously estimated both the vehicle's pose and the positions of the landmarks. Despite the nonlinear nature of the motion model and the presence of measurement noise, the EKF SLAM approach yielded satisfactory results in terms of trajectory estimation and landmark positioning.

During our investigation, we explored the impact of various parameters such as process noise (R), measurement noise (Q), and initial conditions on the performance of the EKF SLAM. By adjusting these parameters, we were able to analyze their influence on metrics such as Root Mean Square Error (RMSE) and maximum error (maxE), aiming to minimize these error measures. Overall, we achieved notable accuracy in our estimations.

This project has highlighted the effectiveness and power of the Kalman Filter (KF), Extended Kalman Filter (EKF), and EKF SLAM techniques. By employing these estimation methods, we obtained reliable trajectory estimates, witnessed the benefits of handling nonlinear models, and experienced the capabilities of simultaneous localization and mapping in the presence of noisy measurements.