16-833: ROBOT LOCALIZATION AND MAPPING

FALL 2024

Homework 2- SLAM using Extended Kalman Filter (EKF-SLAM)

Done by:

Name: Ashwin Biju Nair

Andrew ID: abijunai

Homework 2 - Extended Kalman Filter

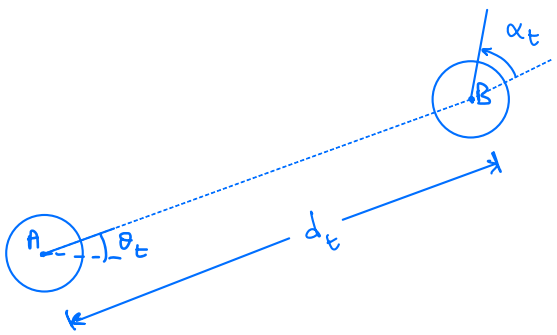
1. Theory

2D ground plane Control input: d_t meters forward, α_t radian rotation.

Pose of robot in global coordinates:

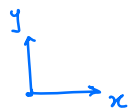
$$p_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

1.



A: (x_t, y_t)

B: (x_{t+1}, y_{t+1})



global coordinate frame

$$\left.\begin{aligned} x_{t+1} &= x_t + d_t \cos\theta \\ y_{t+1} &= y_t + d_t \sin\theta \\ \theta_{t+1} &= \theta_t + \alpha_t \end{aligned}\right\} \quad \underbrace{\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix}}_{p_{t+1}} = \underbrace{\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}}_{p_t} + \underbrace{\begin{bmatrix} d_t \cos\theta \\ d_t \sin\theta \\ \alpha_t \end{bmatrix}}_{B}$$

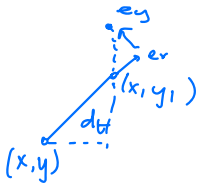
$$p_{t+1} = p_t + B \quad -(1)$$

Equation (1) describes the next pose p_{t+1} as a non-linear function of current pose p_t and control inputs d_t, α_t.

2. Assuming there exist errors:

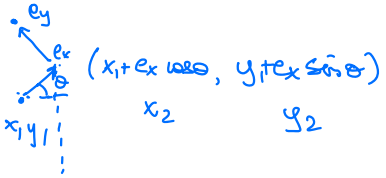
$e_x \sim N(0, \sigma_x^2)$ x-direction
$e_y \sim N(0, \sigma_y^2)$ y-direction
$e_a \sim N(0, \sigma_a^2)$ rotation.

} In robot coordinates.



Uncertainty of robot at time t: $N(0, \Sigma_t)$

Uncertainty of robot at time $(t+1)$: ??

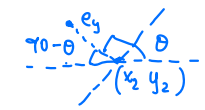


Incorporating the errors into the motion model:

$$x_{t+1} = x_t + (d_t + e_x)\cos\theta - e_y \sin\theta$$

$$y_{t+1} = y_t + (d_t + e_x)\sin\theta + e_y \cos\theta$$

$$\theta_{t+1} = \theta_t + \alpha_t + e_a$$



$180 - 90 - \theta$
$(90 - \theta)$

$(x_2 - e_y \sin\theta, y_2 + e_y \cos\theta)$

∴ The matrix form of the equation becomes:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} + \begin{bmatrix} d_t \cos\theta \\ d_t \sin\theta \\ \alpha_t \end{bmatrix} + \begin{bmatrix} e_x \cos\theta - e_y \sin\theta \\ e_y \sin\theta + e_y \cos\theta \\ e_a \end{bmatrix}$$

Linearizing the model using 1st order Taylor expansion:

$y = f(x)$

$$y \approx \left.\frac{\partial f}{\partial x}\right|_{x_0} (x - x_0) + f(x_0)$$

Computing Jacobian:

$$J = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial x_t} & \frac{\partial x_{t+1}}{\partial y_t} & \frac{\partial x_{t+1}}{\partial \theta_t} \\ \frac{\partial y_{t+1}}{\partial x_t} & \frac{\partial y_{t+1}}{\partial y_t} & \frac{\partial y_{t+1}}{\partial \theta_t} \\ \frac{\partial \theta_{t+1}}{\partial x_t} & \frac{\partial \theta_{t+1}}{\partial y_t} & \frac{\partial \theta_{t+1}}{\partial \theta_t} \end{bmatrix}$$

$$W = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial e_x} & \frac{\partial x_{t+1}}{\partial e_y} & \frac{\partial x_{t+1}}{\partial e_a} \\ \frac{\partial y_{t+1}}{\partial e_x} & \frac{\partial y_{t+1}}{\partial e_y} & \frac{\partial y_{t+1}}{\partial e_a} \\ \frac{\partial \theta_{t+1}}{\partial e_x} & \frac{\partial \theta_{t+1}}{\partial e_y} & \frac{\partial \theta_{t+1}}{\partial e_a} \end{bmatrix}$$

Linearization point : Current estimate of robot pose.

$$x_{t+1} = x_t + (d_t + e_x)\cos\theta - e_y \sin\theta$$

$$y_{t+1} = y_t + (d_t + e_x)\sin\theta + e_y \cos\theta$$

$$\theta_{t+1} = \theta_t + \alpha_t + e_a$$

$$\frac{\partial x_{t+1}}{\partial x_t} = 1 \quad , \quad \frac{\partial x_{t+1}}{\partial y_t} = 0 \quad , \quad \frac{\partial x_{t+1}}{\partial \theta} = -(d_t + e_x)\sin\theta - e_y \cos\theta$$

$$\frac{\partial x_{t+1}}{\partial e_x} = \cos\theta \quad , \quad \frac{\partial x_{t+1}}{\partial e_y} = -\sin\theta \quad , \quad \frac{\partial x_{t+1}}{\partial e_a} = 0$$

$$\frac{\partial y_{t+1}}{\partial x_t} = 0 \quad , \quad \frac{\partial y_{t+1}}{\partial y_t} = 1 \quad , \quad \frac{\partial y_{t+1}}{\partial \theta} = (d_t + e_x)\cos\theta - e_y \sin\theta$$

$$\frac{\partial y_{t+1}}{\partial e_x} = \sin\theta \quad , \quad \frac{\partial y_{t+1}}{\partial e_y} = \cos\theta \quad , \quad \frac{\partial y_{t+1}}{\partial e_a} = 0$$

$$\frac{\partial \theta_{t+1}}{\partial x_t} = 0 \quad , \quad \frac{\partial \theta_{t+1}}{\partial y_t} = 0 \quad , \quad \frac{\partial \theta_{t+1}}{\partial \theta_t} = 1$$

$$\frac{\partial \theta_{t+1}}{\partial e_x} = 0 \quad , \quad \frac{\partial \theta_{t+1}}{\partial e_y} = 0 \quad , \quad \frac{\partial \theta_{t+1}}{\partial e_a} = 1$$

∴ The Jacobian becomes :

$$J = \begin{bmatrix} 1 & 0 & -(d_t + e_x)\sin\theta - e_y\cos\theta \\ 0 & 1 & (d_t + e_x)\cos\theta - e_y\sin\theta \\ 0 & 0 & 1 \end{bmatrix} \qquad W = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

current state estimate

$$P_{t+1} \approx f(\hat{P}_t, u_t, 0) + J(P_t - \hat{P}_t) + W e_t$$

→ without error terms.

$\hat{P}_t$ → represents the current state estimate $(x_t \; y_t \; \theta_t)$

$$f(\hat{P}_t, u_t, 0) = \begin{bmatrix} x_t + d_t \cos\theta \\ y_t + d_t \sin\theta \\ \theta_t + \alpha_t \end{bmatrix}, \quad \hat{P}_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}$$

$$P_{t+1} = \begin{bmatrix} x_t + d_t \cos\theta \\ y_t + d_t \sin\theta \\ \theta_t + \alpha_t \end{bmatrix} + \begin{bmatrix} 1 & 0 & -d_t \sin\theta \\ 0 & 1 & d_t \cos\theta \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} x_t + (d_t + e_x)\cos\theta - e_y \sin\theta \\ y_t + (d_t + e_x)\sin\theta + e_y \cos\theta \\ \theta_t + \alpha_t + e_a \end{bmatrix} - \begin{bmatrix} x_t + d_t \cos\theta \\ y_t + d_t \sin\theta \\ \theta_t + \alpha_t \end{bmatrix} \right)$$

$$+ \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ e_a \end{bmatrix}$$

$$= \begin{bmatrix} x_t + d_t \cos\theta \\ y_t + d_t \sin\theta \\ \theta_t + \alpha_t \end{bmatrix} + \begin{bmatrix} 1 & 0 & -d_t \sin\theta \\ 0 & 1 & d_t \cos\theta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_x \cos\theta - e_y \sin\theta \\ e_x \sin\theta - e_y \cos\theta \\ e_a \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_x \\ e_y \\ e_a \end{bmatrix}$$

$$= \begin{bmatrix} x_t + d_t \cos\theta \\ y_t + d_t \sin\theta \\ \theta_t + \alpha_t \end{bmatrix} + \begin{bmatrix} e_x \cos\theta - e_y \sin\theta - e_a d_t \sin\theta \\ e_x \sin\theta - e_y \cos\theta + d_t \cos\theta \\ e_a \end{bmatrix} + \begin{bmatrix} e_x \cos\theta - e_y \sin\theta \\ e_x \sin\theta + e_y \cos\theta \\ e_a \end{bmatrix}$$

The covariance prediction (non-additive noise formulation):

$$\Sigma_{t+1} = J \Sigma_t J^T + W Q W^T$$

$$Q = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\alpha^2 \end{bmatrix}, \quad J = \begin{bmatrix} 1 & 0 & -d_t \sin\theta \\ 0 & 1 & d_t \cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$

$\Sigma_t \rightarrow$ covariance of robot pose @ time 't'

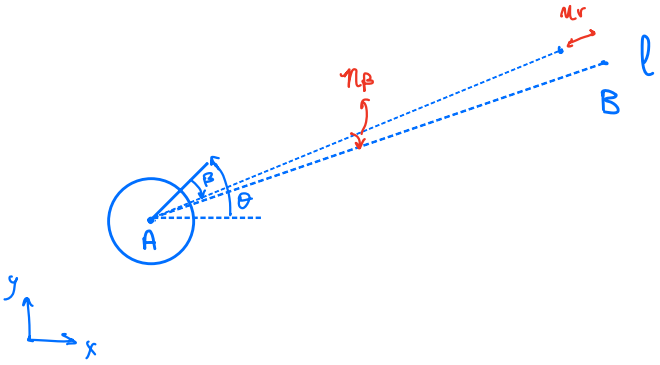
3. Laser Sensor measurement: Bearing angle β $(-\pi, \pi]$

Range r

Measurement noise: $n_\beta \sim N(0, \sigma_\beta^2)$

$n_r \sim N(0, \sigma_r^2)$

Landmark position: (l_x, l_y) → global coordinates



A: (x_t, y_t, θ_t)

B: (l_x, l_y)

β: Bearing angle (angle between robot heading & landmark, measured counterCW)

$$l_x = x_t + (r + n_r)\cos(\theta - (-\beta - n_\beta))$$

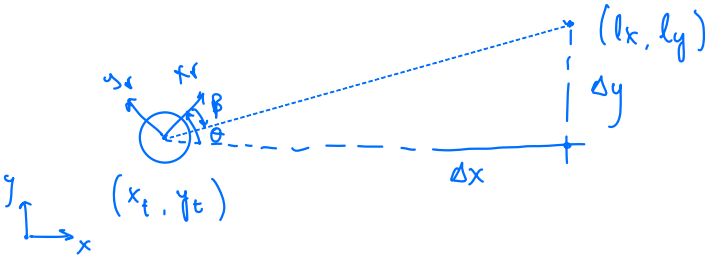
$$l_y = y_t + (r + n_r)\sin(\theta - (-\beta - n_\beta))$$

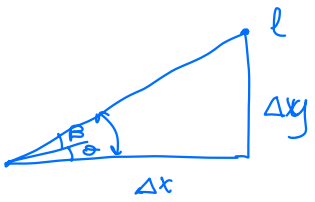
$$\therefore \quad l_x = x_t + (r + n_r)\cos(\theta + \beta + n_\beta)$$

$$l_y = y_t + (r + n_r)\sin(\theta + \beta + n_\beta)$$

The above equation expresses landmark position (l_x, l_y) in terms of $P_t : (x_t, y_t, \theta_t)$, β and noise terms.

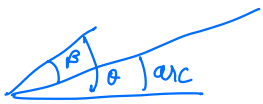
4.





$\arctan(\,) - \theta$

$$\left.\begin{aligned}\Delta x &= l_x - x_t \\ \Delta y &= l_y - y_t\end{aligned}\right\} \text{ global coordinates}$$



Landmark position from robot coordinates:

We need to rotate by angle θ in the counterclockwise direction.

∴ The rotation matrix for this rotation is:

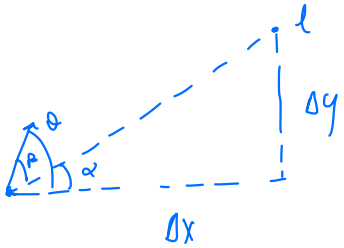
$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

$$\begin{bmatrix} x_r \\ y_r \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$

Range: $r = \sqrt{\Delta x^2 + \Delta y^2} + n_r$ ← noise

↳ Euclidean distance

Bearing angle: $\beta = \alpha - \theta + n_\beta$



$$\beta = \arctan2(\Delta y, \Delta x) - \theta_t + n_\beta$$ ← noise

'−' sign as β is negative in this case (ensures generalization)

arctan2( ) is used to avoid division by zero ($\Delta x = 0$)

& to ensure the correct sign

Final equations of measurement predictions:

$$r = \sqrt{\Delta x^2 + \Delta y^2} + n_r$$

$$\beta = \text{wrap2pi}\left(\text{arctan2}\left(\Delta y, \Delta x\right) - \theta_t + n_\beta\right)$$

$$r = \sqrt{\left(l_x - x_t\right)^2 + \left(l_y - y_t\right)^2} + n_r$$

$$\beta = \text{wrap2pi}\left[\text{arctan2}\left(\left(l_y - y_t\right), \left(l_x - x_t\right)\right) - \theta_t + n_\beta\right]$$

5.

$$H_p = \begin{bmatrix} \frac{\partial \beta}{\partial x} & \frac{\partial \beta}{\partial y} & \frac{\partial \beta}{\partial \theta} \\ \frac{\partial r}{\partial x} & \frac{\partial r}{\partial y} & \frac{\partial r}{\partial \theta} \end{bmatrix}$$

$$r = \sqrt{(l_x - x_t)^2 + (l_y - y_t)^2} + n_r$$

$$\beta = \text{wrap2pi}\left[\arctan2\left((l_y - y_t), (l_x - x_t)\right) - \theta_t + n_\beta\right]$$

$$\frac{\partial \beta}{\partial x} = \frac{(l_y - y_t)}{(l_x - x_t)^2 + (l_y - y_t)^2}$$

Using: $z = \tan^{-1}(y/x)$

$$\frac{\partial z}{\partial x} = -y/(x^2+y^2)$$

$$\frac{\partial z}{\partial y} = x/(x^2+y^2)$$

$$\frac{\partial \beta}{\partial y} = \frac{-(l_x - x_t)}{(l_x - x_t)^2 + (l_y - y_t)^2}$$

$$\frac{\partial \beta}{\partial \theta} = -1$$

$$\frac{\partial r}{\partial x} = \frac{1}{2\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}} \times \left[-2(l_x - x_t)\right]$$

$$= \frac{-(l_x - x_t)}{\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}}$$

$$\frac{\partial r}{\partial y} = \frac{1}{2\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}} \times \left[-2(l_y - y_t)\right]$$

$$= \frac{-(l_y - y_t)}{\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}}$$

$$\frac{\partial r}{\partial \theta} = 0$$

$$H_p = \begin{bmatrix} \dfrac{(l_y - y_t)}{(l_x - x_t)^2 + (l_y - y_t)^2} & \dfrac{-(l_x - x_t)}{(l_x - x_t)^2 + (l_y - y_t)^2} & -1 \\ \dfrac{-(l_x - x_t)}{\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}} & \dfrac{-(l_y - y_t)}{\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}} & 0 \end{bmatrix}$$

6.

$$H_\ell = \begin{bmatrix} \frac{\partial \beta}{\partial \ell_x} & \frac{\partial \beta}{\partial \ell_y} \\ \frac{\partial r}{\partial \ell_x} & \frac{\partial r}{\partial \ell_y} \end{bmatrix}$$

$$r = \sqrt{(\ell_x - x_t)^2 + (\ell_y - y_t)^2} + n_r$$

$$\beta = \text{wrap}2\pi \left[\arctan2\left((\ell_y - y_t), (\ell_x - x_t)\right) - \theta_t + n_\beta \right]$$

$$\frac{\partial \beta}{\partial \ell_x} = \frac{-(\ell_y - y_t)}{(\ell_x - x_t)^2 + (\ell_y - y_t)^2}$$

$$\frac{\partial \beta}{\partial \ell_y} = \frac{(\ell_x - x_t)}{(\ell_x - x_t)^2 + (\ell_y - y_t)^2}$$

$$\frac{\partial r}{\partial \ell_x} = \frac{(\ell_x - x_t)}{\sqrt{(\ell_x - x_t)^2 + (\ell_y - y_t)^2}}$$

$$\frac{\partial r}{\partial \ell_y} = \frac{(\ell_y - y_t)}{\sqrt{(\ell_x - x_t)^2 + (\ell_y - y_t)^2}}$$

$$H_\ell = \begin{bmatrix} \frac{-(\ell_y - y_t)}{(\ell_x - x_t)^2 + (\ell_y - y_t)^2} & \frac{(\ell_x - x_t)}{(\ell_x - x_t)^2 + (\ell_y - y_t)^2} \\ \frac{(\ell_x - x_t)}{\sqrt{(\ell_x - x_t)^2 + (\ell_y - y_t)^2}} & \frac{(\ell_y - y_t)}{\sqrt{(\ell_x - x_t)^2 + (\ell_y - y_t)^2}} \end{bmatrix}$$

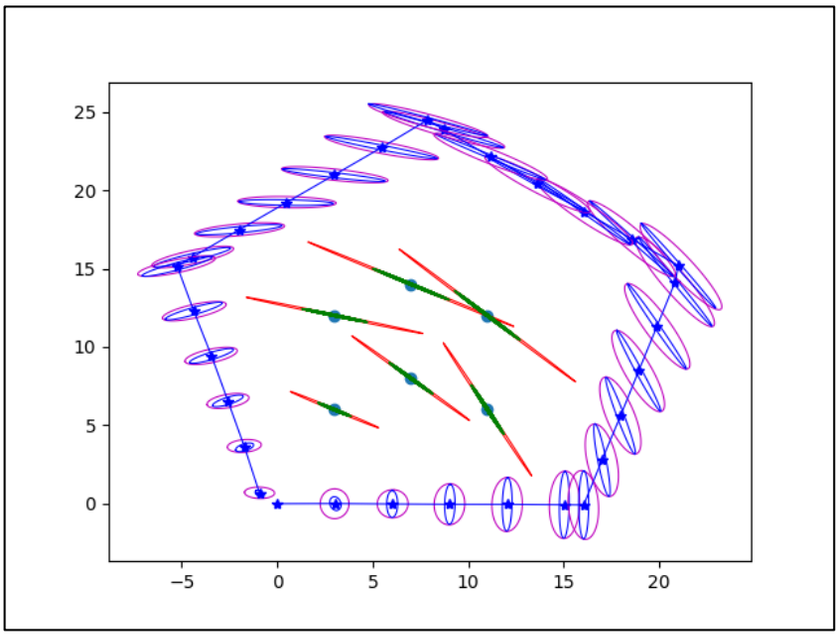
The reason we calculate the measurement Jacobian only with respect to itself is because we assume that the landmark observations are independent. One landmark does not directly affect the measurement of other landmarks.

Q2. Implementation and Evaluation

Q2.1.

The fixed number of landmarks being observed over the entire sequence is six.

Q2.2.



Q2.3.

EKF-SLAM improves the estimation of both the trajectory and map by progressively reducing uncertainties as more observations are made. For the robot's trajectory, the magenta ellipses (predicted uncertainties) are larger than the blue ellipses (updated uncertainties), indicating that measurements help refine the robot's position estimate. Similarly, for landmarks, the red ellipses (initial uncertainties) are larger than the green ellipses (updated uncertainties), showing that repeated observations of landmarks lead to more accurate position estimates. This iterative process of prediction and update allows EKF-SLAM to simultaneously refine both the robot's trajectory and the map of landmarks, with uncertainties generally decreasing over time as more information is incorporated.

Q2.4.

The program output for Euclidian, Mahalanobis distance and whether the final landmark is inside its 3-sigma ellipse is as follows:

```
Landmark 1:
  Euclidean distance: 0.0194
  Mahalanobis distance: 0.0796
Landmark 2:
  Euclidean distance: 0.0380
  Mahalanobis distance: 0.3091
Landmark 3:
  Euclidean distance: 0.0355
  Mahalanobis distance: 0.0915
Landmark 4:
  Euclidean distance: 0.0529
  Mahalanobis distance: 0.1098
Landmark 5:
  Euclidean distance: 0.0428
```

```
Landmark 1 is inside its 3-sigma ellipse
Landmark 2 is inside its 3-sigma ellipse
Landmark 3 is inside its 3-sigma ellipse
Landmark 4 is inside its 3-sigma ellipse
Landmark 5 is inside its 3-sigma ellipse
Landmark 6 is inside its 3-sigma ellipse
```

```
  Mahalanobis distance: 0.3713
Landmark 6:
  Euclidean distance: 0.0533
  Mahalanobis distance: 0.2796
```

- The small Euclidean distances (0.0194 to 0.0533) suggest that the EKF-SLAM algorithm has produced accurate estimates of the landmark positions.
- The fact that all landmarks are within their 3-sigma ellipses and have Mahalanobis distances less than 3 indicates that the algorithm's uncertainty estimates are consistent with the actual errors. The algorithm appears to be neither overconfident nor too conservative in its uncertainty estimates, as evidenced by the reasonable Mahalanobis distances.
- There is some variation in the accuracy of different landmark estimates, with Landmark 1 having the smallest Euclidean distance (0.0194) and Landmark 4 having the largest (0.0529). This variation could be due to factors such as the landmarks' positions relative to the robot's trajectory or measurement noise.
- The combination of small Euclidean distances and appropriate Mahalanobis distances suggests that the EKF-SLAM implementation is reliable and performing as expected.
- Landmarks with higher Mahalanobis distances relative to their Euclidean distances (Landmark 2,5 and 6) suggest more uncertainty, despite being physically close to the true positions.

Q3. Discussion

Q3.1.

The zero terms in the initial landmark covariance matrix become non-zero in the final state covariance matrix due to the nature of the Extended Kalman Filter (EKF) SLAM process. Here's why:

- Initially, the landmarks are assumed to be independent, which is why their covariances are set to zero in the off-diagonal blocks.
- As the robot moves and observes landmarks, the update step of the EKF introduces correlations between the robot's pose and the landmarks, as well as between different landmarks.
- These correlations are reflected in the off-diagonal elements of the covariance matrix, which become non-zero over time.
- This process captures the fact that errors in estimating one landmark's position can affect the estimates of other landmarks and the robot's pose.

The assumption here that is not necessarily correct is that the initial cross-covariances between the robot's pose and the landmarks are set to zero (the np.zeros((3, 2 * k)) and np.zeros((2 * k, 3)) blocks).

This assumption implies that the initial uncertainty in the robot's pose is completely independent of the initial uncertainty in the landmark positions. However, in reality, there might be some correlation between these uncertainties from the very beginning, especially since the initial landmark positions are estimated based on the initial robot pose.

The reason this assumption is made is often for simplicity and to provide a starting point for the filter. In practice, it's challenging to accurately specify these initial cross-covariances, so starting with zero and allowing the filter to develop correlations over time is a common approach. However, it's important to be aware that this is a simplification that might not perfectly reflect the true initial state of uncertainty in the system

Q3.2. Parameter Tuning

Comparing these results to the baseline for each parameter:

- How do the landmark positions change?
    - Are the uncertainty ellipses larger or smaller?
    - How do the Euclidean and Mahalanobis distances change?
    - Is the final robot pose estimate significantly different?

For a parameter (say sig_x), sig_x_large denotes the case where sig_x = sig_x*10 and sig_x_small denotes the case where sig_x = sig_x/10.

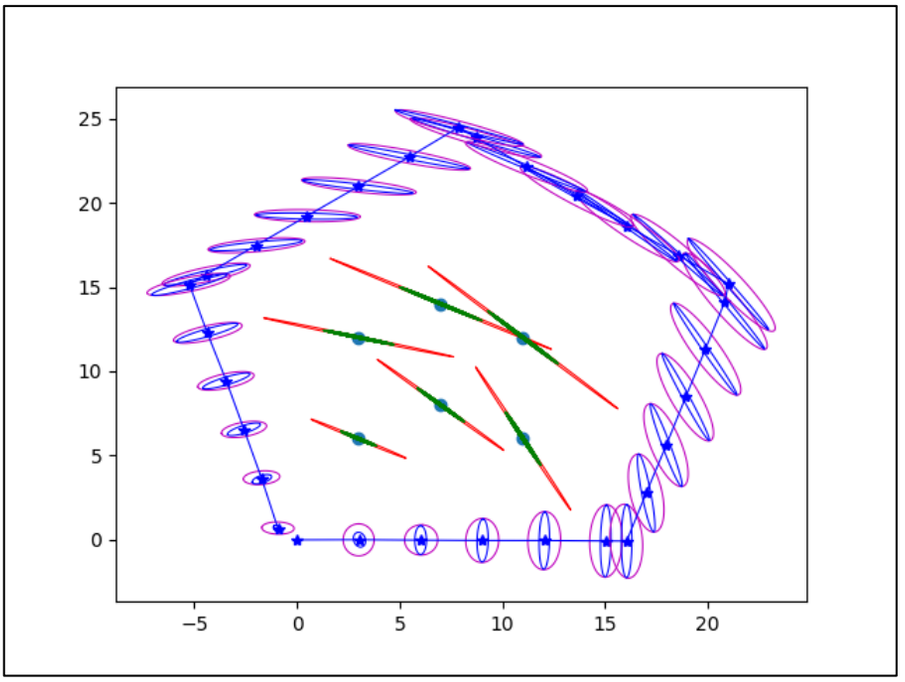


Figure 2- Baseline output (original parameters)

Sig_x

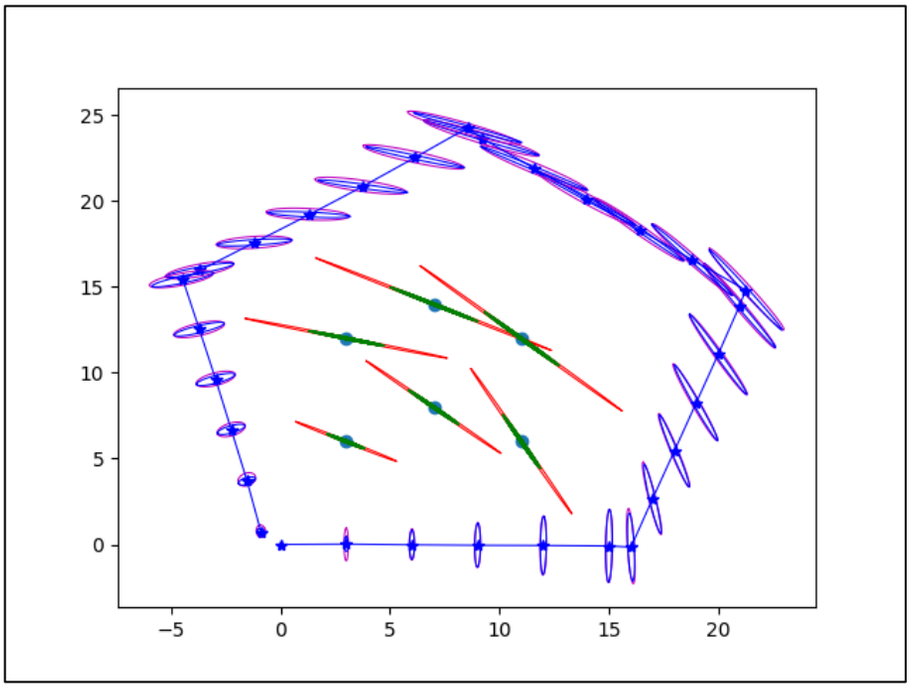


Figure 3- output of sig_x/10

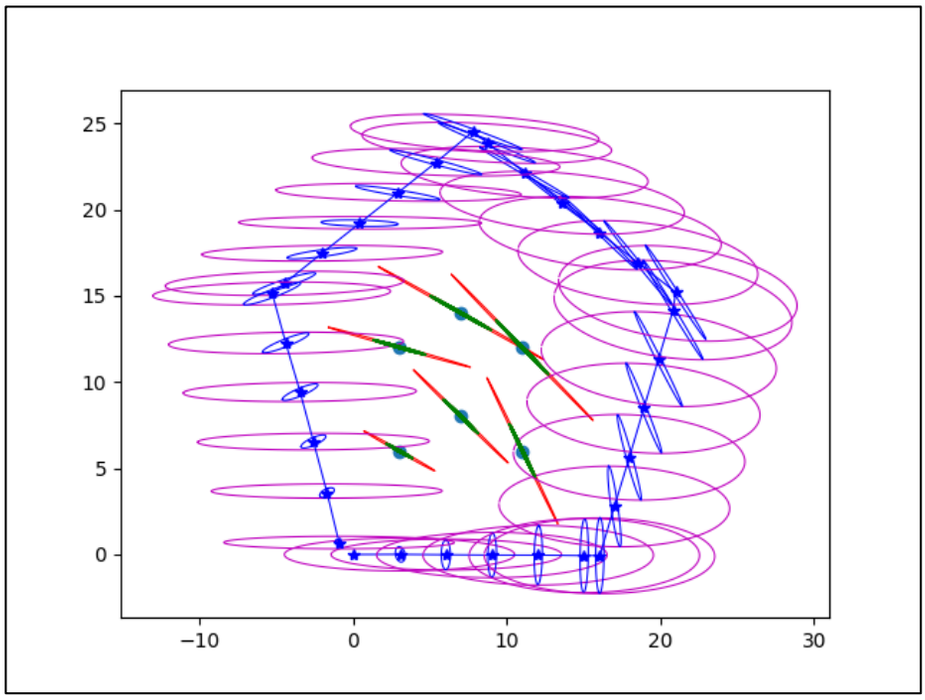


*Figure 4- output of sig_x*10*

- Increasing sig_x leads to larger uncertainty ellipses along the x-direction for the predicted values, greater Euclidean and Mahalanobis distances (not significant- maintains reasonable accuracy). The final robot pose is similar to the baseline, with minor differences. This suggests that while there is more noise, the system can still estimate the pose reasonably well.
- Decreasing sig_x results in smaller uncertainty ellipses which are closer to the updated ellipses. Distances are increase significantly, suggesting that reducing noise too much can lead to inaccurate estimates due to potential overfitting. The final landmark positions show greater deviation from the baseline, suggesting reduced accuracy due to underestimated motion noise

Sig_y

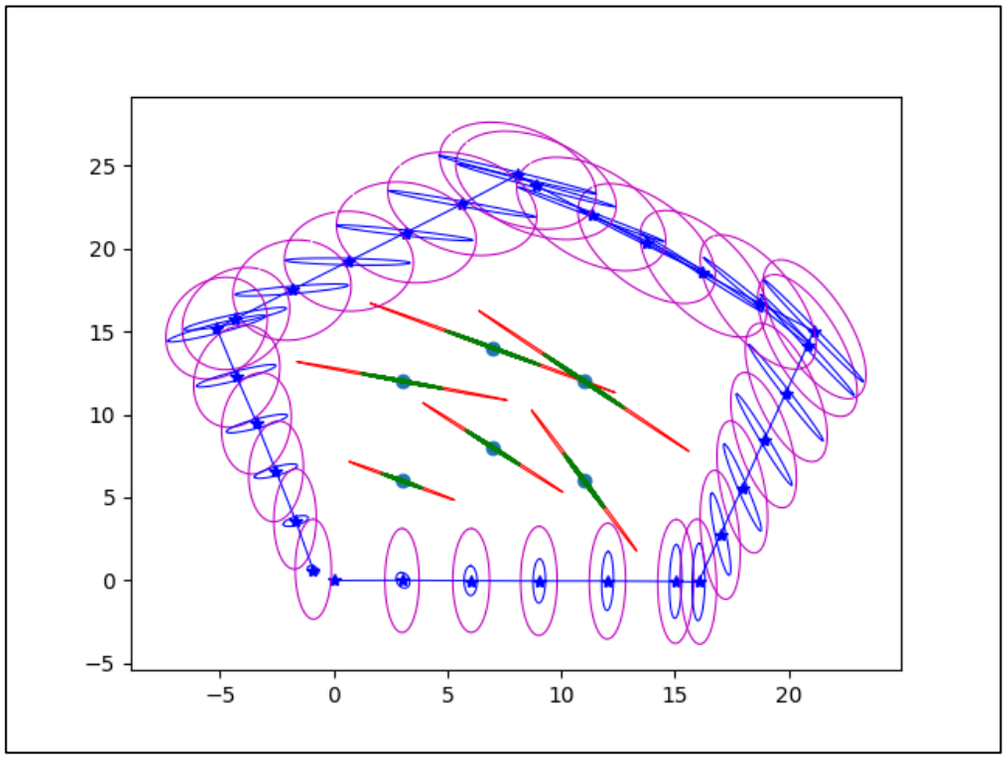


*Figure 5- output of sig_y*10*

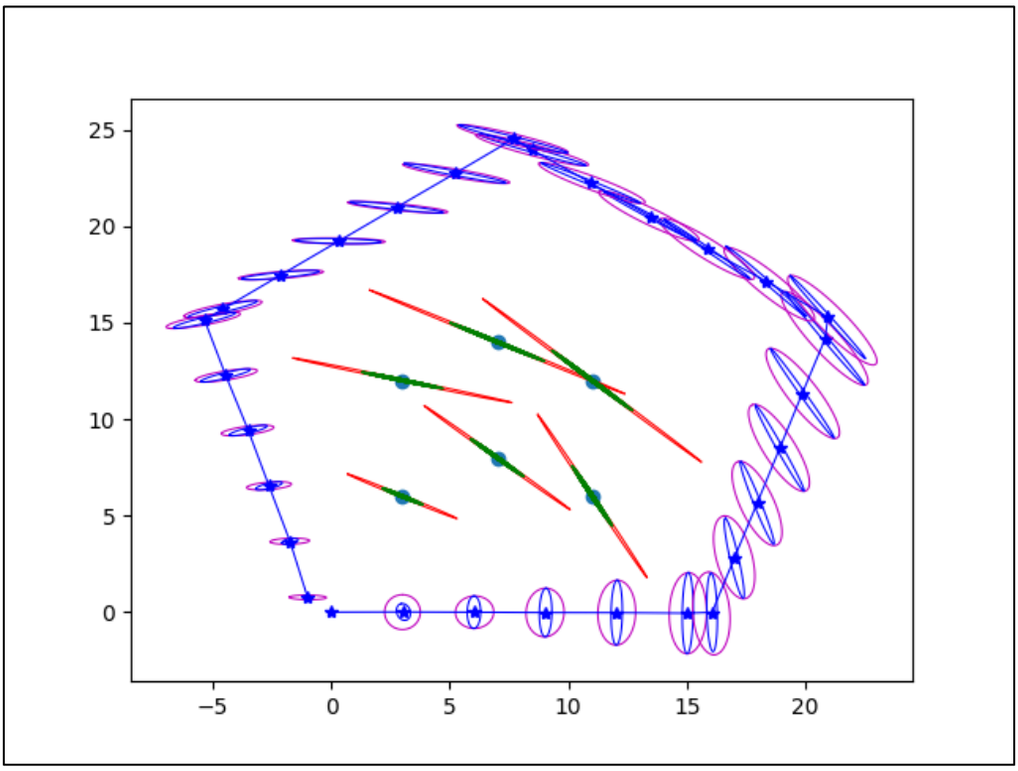


Figure 6- output of sig_y/10

- The uncertainty ellipses are larger than those in the baseline, reflecting increased uncertainty in the y-direction due to higher noise. Increasing sig_y leads to the Euclidean and Mahalanobis distances to slightly increase, indicating a minor increase in positional error due to higher noise. The final landmark and final pose show minor deviations from the baseline.
- Decreasing sig_y results in uncertainty ellipses are smaller, indicating reduced uncertainty in position. The final landmark positions show more significant deviations from the baseline. Whereas, the final robot pose shows slight deviations from the baseline, indicating potential instability due to reduced noise levels

Sig_alpha

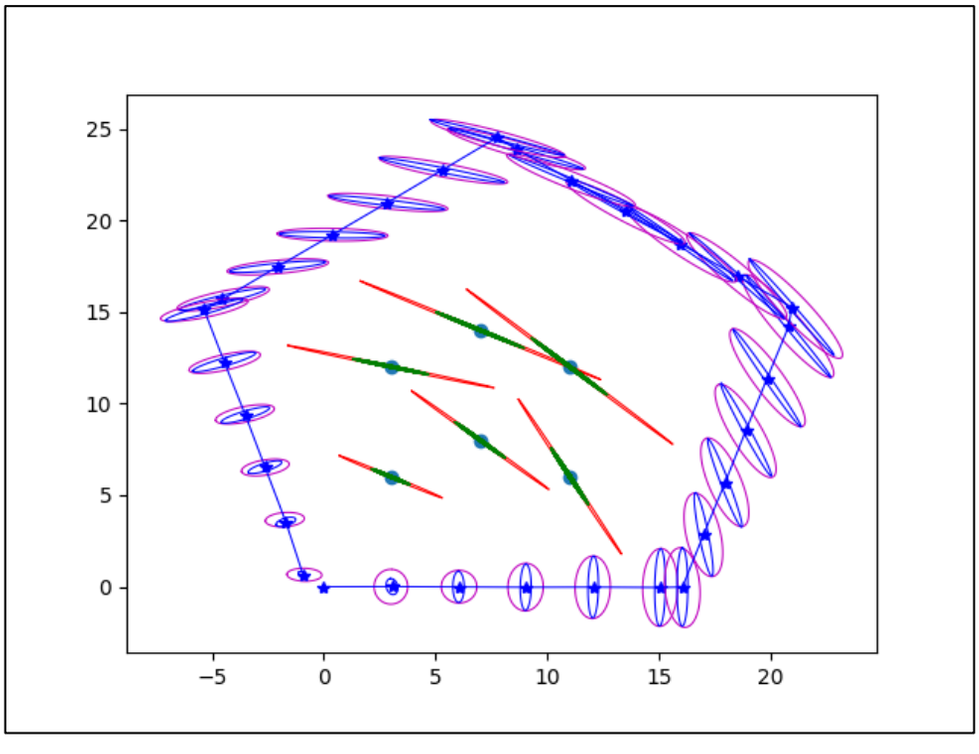


Figure 7- output of sig_alpha/10

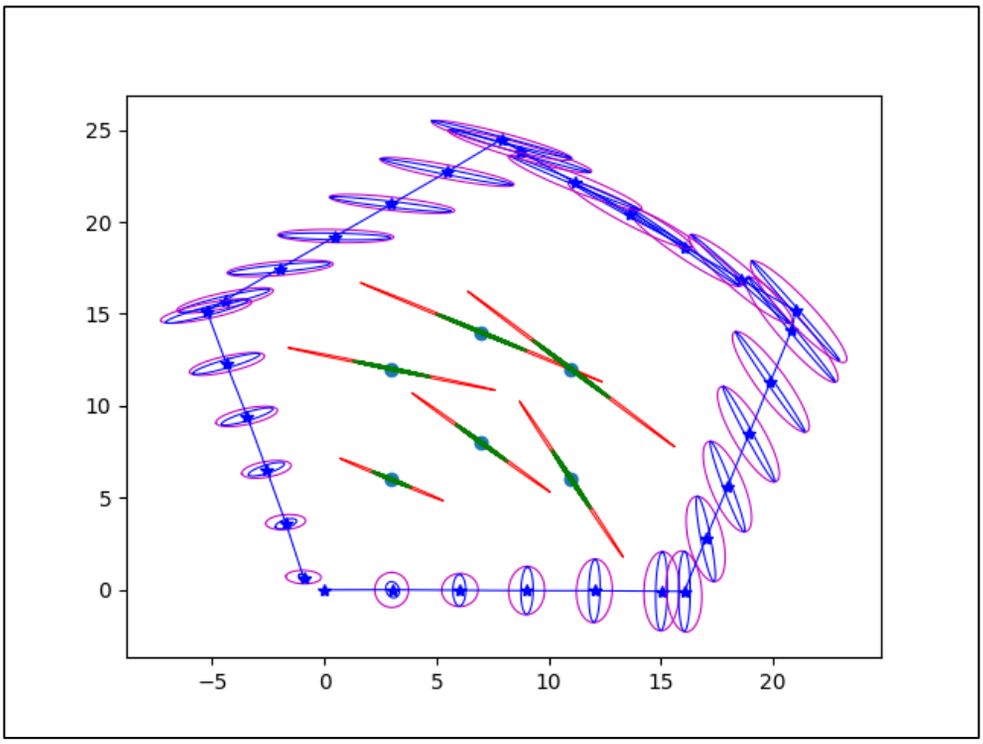


*Figure 8- output of sig_alpha * 10*

- Increasing sig_alpha leads to slightly larger uncertainty ellipses due to higher rotational noise. Both distances are generally lower than the baseline, suggesting that the system can handle increased rotational noise without significant loss of accuracy. The landmark positions and final robot pose are very similar to the baseline with negligible differences. This indicates robustness against increased rotational noise.
- Decreasing sig_alpha results in smaller uncertainty ellipses, indicating reduced uncertainty in orientation. Both distances increase significantly, indicating greater positional error and uncertainty. The final robot pose shows slight deviations from the baseline, suggesting potential instability due to reduced noise levels.

Sig_beta

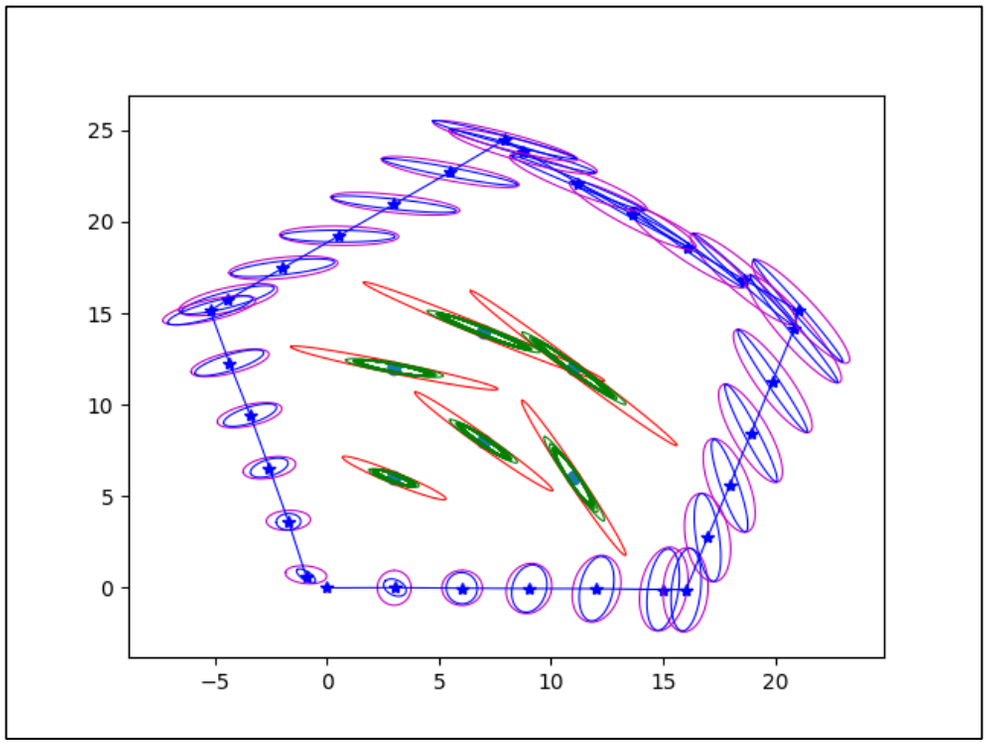


*Figure 9- output of sig_beta *10*

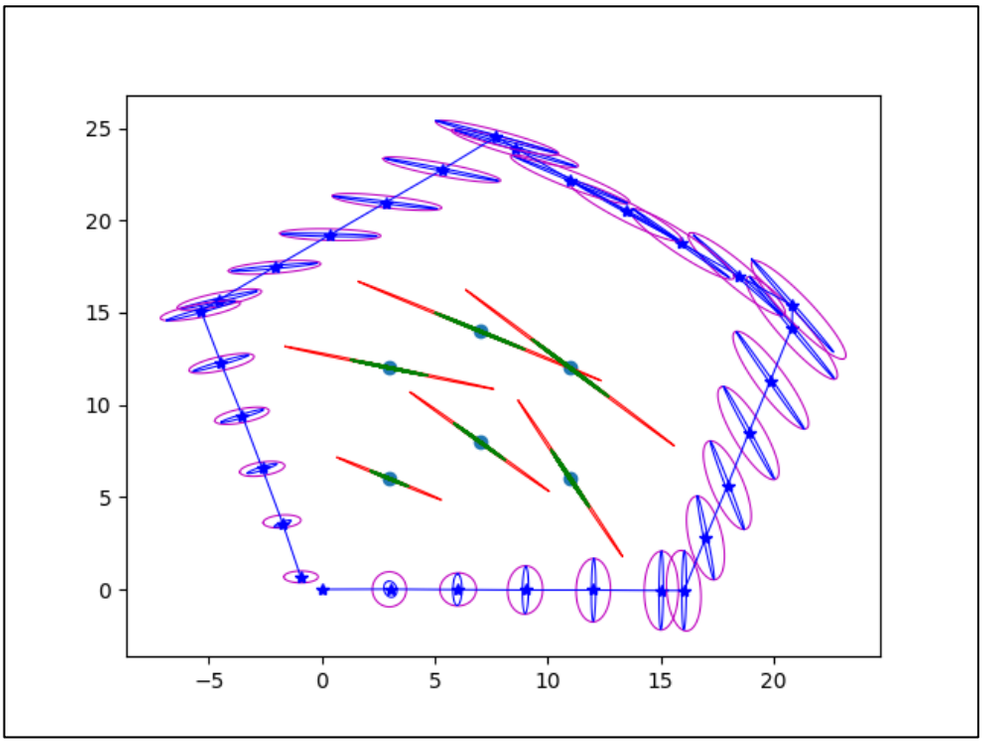


Figure 10- output of sig_beta / 10

- Increasing sig_beta led to the landmark positions are very close to the baseline, with minimal shifts. Euclidean and Mahalanobis distances are generally lower than the baseline, indicating improved accuracy. The final robot pose is very similar to the baseline with negligible differences again indicating robustness against increased bearing noise.

- Decreasing sig_beta led to landmark positions deviating more significantly from the baseline. Both Euclidean and Mahalanobis distances also significantly increased, indicating greater positional error and uncertainty. The final robot pose shows only slight deviations.

Sig_r

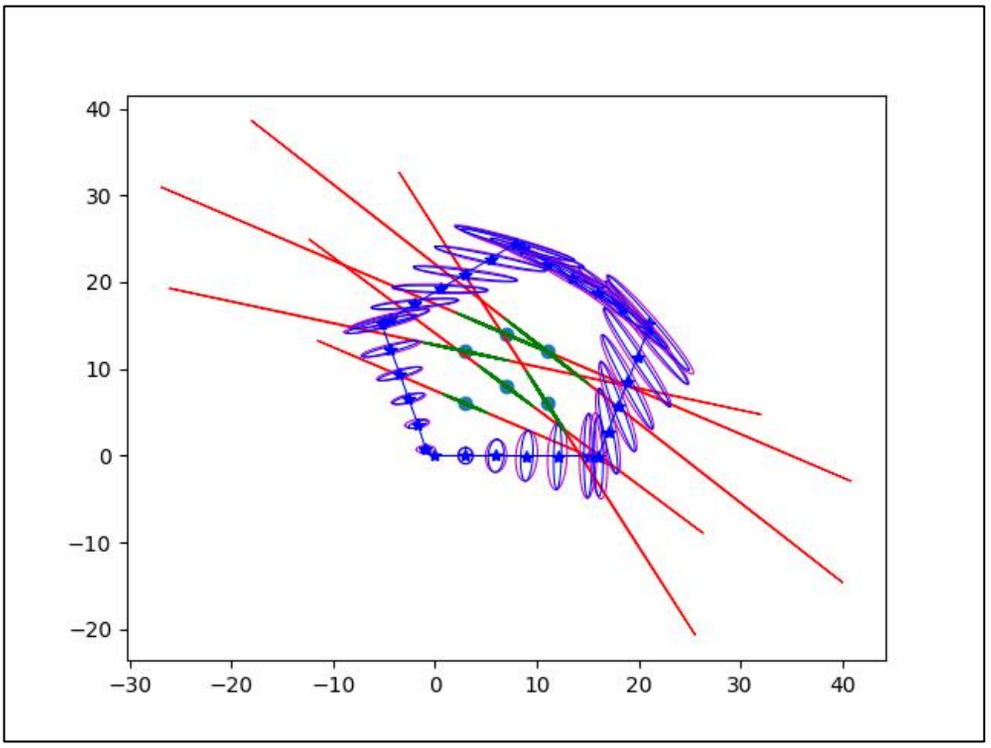


*Figure 11- output of sig_r *10*

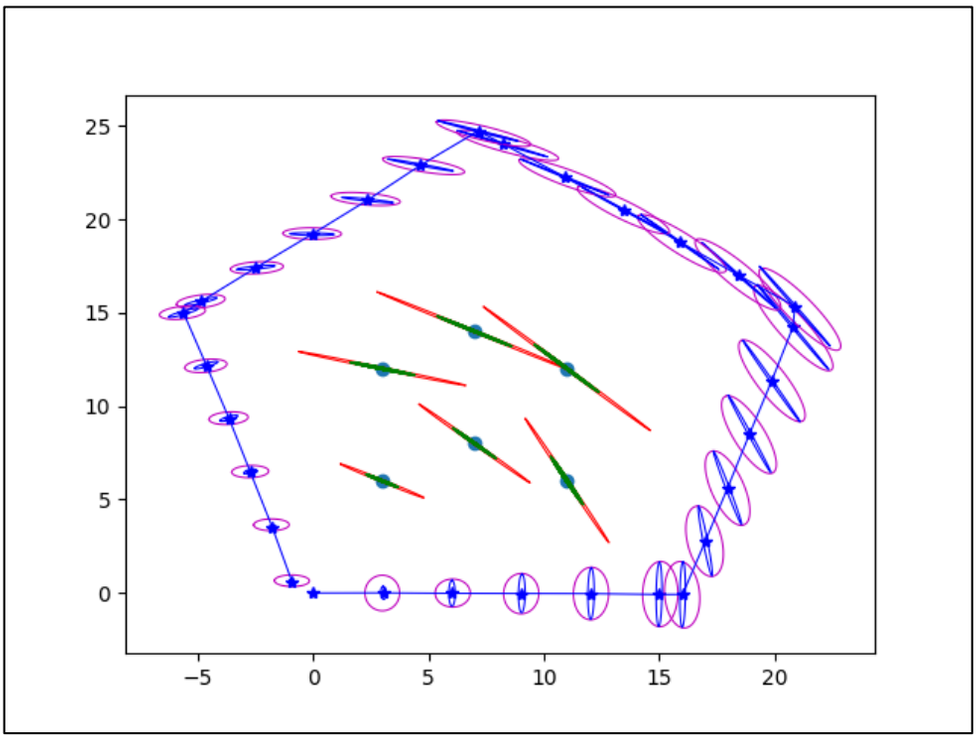


Figure 12- output of sig_r/10

- Increasing sig_r leads to the landmark positions are very close to the baseline, with minimal shifts. Euclidean and Mahalanobis distances are generally lower than the baseline, indicating improved accuracy in some cases, not all. The final robot pose shows slight deviation from the baseline with high uncertainty.
- Decreasing sig_r results in smaller uncertainty ellipses but the landmark positions deviate more significantly from the baseline. Both Euclidean and Mahalanobis distances are significantly increased, indicating greater positional error and uncertainty. The final robot pose shows slight deviations from the baseline due to overconfidence in range measurements.

Q3.3

To avoid the unbounded growth in the number of landmarks a few of the following approaches can be considered:

- Instead of using and updating a full state vector and covariance matrix for all the landmarks, we can consider only the most recent observations and landmarks. This would limit the size of the state vector and covariance matrices. It would also ensure constant computation time.
- Implementing feature selection algorithms (Pearson Correlation, RFE, etc.) to choose the most informative landmarks, discarding less useful ones. This would help maintain a bounded number of landmarks in the state vector
- Submapping
    - Divide the map into smaller, manageable submaps. Each submap contains a limited number of landmarks, and the robot localizes within the current submap.
    - This approach reduces the computational complexity and allows for parallel processing of submaps.
- Graph-based SLAM
        - Representing the SLAM problem as a graph, where nodes represent robot poses and landmark positions, and edges represent constraints.
        - Using various optimization techniques like graph relaxation or bundle adjustment to solve the SLAM problem efficiently.
- FastSLAM
    - Using particle filter-based SLAM methods like FastSLAM, which can handle a large number of landmarks more efficiently than EKF-SLAM
    - FastSLAM uses particle filters to estimate the robot's path, allowing it to maintain multiple hypotheses about the robot's location. For each particle, it uses separate extended Kalman filters to estimate landmark positions.
    - FastSLAM can also better manage data association uncertainties by maintaining multiple hypotheses
- Local Maps
    - This approach involves creating small, manageable maps of the robot's immediate surroundings, each containing a limited number of landmarks.
    - This approach allows for constant-time updates in the local map while still maintaining global consistency as these local maps are merged into a global map periodically.
    - It reduces the computational burden of updating a large global map in real-time, allows for parallel processing of different local maps, and can handle loop closures more effectively.
- Hierarchical SLAM:
    - At the lower level, it creates a set of independent local maps, each covering a small area with limited landmarks. These local maps can be updated in constant time, regardless of the overall environment size.
    - The upper level maintains a graph structure, where nodes represent local maps and edges represent their relative positions.
    - When a loop is detected, only the affected local maps are updated, reducing computational complexity.
- Incremental Smoothing and Mapping (iSAM):
    - It uses a Bayes tree data structure to represent the SLAM problem and performs incremental updates. iSAM maintains a factored representation of the posterior probability, allowing for quick updates without recalculating the entire map.
    - It intelligently decides when to relinearize and reorder variables, balancing computational efficiency with solution accuracy.