16-833: Robot Localization and Mapping
Homework 2: Written Report

Shahram Najam Syed
Andrew ID: snsyed

16th October, 2024

Question 1.1: Predicting the Next Pose of the Robot

[For detailed working of the mathematical derivation, please refer to the scanned work annexed with this answer] To predict the next pose of the robot p_{t+1} based on the current pose p_t and control inputs d_t (forward movement) and α_t (rotation), we can derive the motion model as follows:

Given

Current pose:

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

where x_t and y_t are the coordinates of the robot's position, and θ_t is its orientation.

Control inputs:

- Forward movement along the robot's current heading: d_t
- Rotation about the robot's center: α_t

Assumption:

- There is no noise or error in the control system.

Derivation

Calculate the change in position due to forward movement:

The robot moves a distance d_t in the direction of its current orientation θ_t.

Change in x-coordinate:

$$\Delta x_t = d_t \cos\theta_t$$

Change in y-coordinate:

$$\Delta y_t = d_t \sin\theta_t$$

Calculate the change in orientation due to rotation:

Change in orientation:

$$\Delta\theta_t = \alpha_t$$

Update the pose components:

New x-coordinate:

$$x_{t+1} = x_t + \Delta x_t = x_t + d_t \cos\theta_t$$

New y-coordinate:

$$y_{t+1} = y_t + \Delta y_t = y_t + d_t \sin\theta_t$$

New orientation:

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

Combining the updated components, we obtain the next pose:

$$p_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + d_t \cos\theta_t \\ y_t + d_t \sin\theta_t \\ \theta_t + \alpha_t \end{bmatrix}$$

Final Expression

The next pose of the robot p_{t+1} is predicted as a nonlinear function of the current pose p_t and the control inputs d_t and α_t:

$$p_{t+1} = \begin{bmatrix} x_t + d_t \cos\theta_t \\ y_t + d_t \sin\theta_t \\ \theta_t + \alpha_t \end{bmatrix}$$

Question 1.2: Predicted Uncertainty of the Robot's Pose

[For detailed working of the mathematical derivation, please refer to the scanned work annexed with this answer] To determine the predicted uncertainty of the robot's pose at time $t+1$, we need to account for the uncertainties introduced by both the robot's previous state and the control inputs.

1. Define the Robot's State and Control Inputs

Robot's Pose at Time t:

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

- x_t: Position along the global x-axis.
- y_t: Position along the global y-axis.
- θ_t: Orientation angle with respect to the global frame.

Control Inputs at Time t:

$$u_t = \begin{bmatrix} d_t \\ \alpha_t \end{bmatrix}$$

- d_t: Forward movement distance along the robot's current heading.
- α_t: Rotation angle (change in orientation).

2. Define the Process Noise (Movement Errors)

The robot's movement introduces errors due to imperfections. These errors are modeled as zero-mean Gaussian random variables in the robot's coordinate frame:

$$w_t = \begin{bmatrix} e_x \\ e_y \\ e_\alpha \end{bmatrix}$$

where:

$$e_x \sim \mathcal{N}(0, \sigma_x^2), \quad e_y \sim \mathcal{N}(0, \sigma_y^2), \quad e_\alpha \sim \mathcal{N}(0, \sigma_\alpha^2)$$

3. Write the Motion Model Including Errors

The robot's motion can be modeled as:

$$p_{t+1} = f(p_t, u_t, w_t)$$

where f maps the current state, control inputs, and process noise to the next state.

Position Updates:

$$\Delta x_{\text{global}} = \cos\theta_t(d_t + e_x) - \sin\theta_t e_y$$

$$\Delta y_{\text{global}} = \sin\theta_t(d_t + e_x) + \cos\theta_t e_y$$

Orientation Update:

$$\Delta\theta = \alpha_t + e_\alpha$$

4. Linearize the Motion Model

To propagate the uncertainty, we linearize the motion model around the mean values.

Compute the Jacobian with Respect to the State F_t

The Jacobian F_t is:

$$F_t = \frac{\partial p_{t+1}}{\partial p_t} = \begin{bmatrix} 1 & 0 & -d_t \sin\theta_t \\ 0 & 1 & d_t \cos\theta_t \\ 0 & 0 & 1 \end{bmatrix}$$

Compute the Jacobian with Respect to the Process Noise G_t

The Jacobian G_t is:

$$G_t = \frac{\partial p_{t+1}}{\partial w_t} = \begin{bmatrix} \cos\theta_t & -\sin\theta_t & 0 \\ \sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5. Define the Process Noise Covariance Matrix

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

6. Propagate the Covariance

The predicted covariance Σ_{t+1} is:

$$\Sigma_{t+1} = F_t \Sigma_t F_t^T + G_t Q_t G_t^T$$

Question 1.3: Estimated Position of the Landmark in Global Coordinates

[For detailed working of the mathematical derivation, please refer to the scanned work annexed with this answer] To determine the estimated position (l_x, l_y) of the landmark l in global coordinates, we need to consider the robot's pose, the sensor measurements, and the associated noise terms.

1. Define the Robot's Pose and Sensor Measurements

Robot's Pose at Time t:

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

- x_t: Robot's position along the global x-axis.
- y_t: Robot's position along the global y-axis.
- θ_t: Robot's orientation angle with respect to the global frame.

Sensor Measurements:

- **Bearing Angle** β: The angle between the robot's heading and the direction to the landmark, with measurement noise $n_\beta \sim \mathcal{N}(0, \sigma_\beta^2)$.
- **Range** r: The distance from the robot to the landmark, with measurement noise $n_r \sim \mathcal{N}(0, \sigma_r^2)$.

2. Express the Noisy Measurements

$$\beta_{\text{measured}} = \beta + n_\beta$$

$$r_{\text{measured}} = r + n_r$$

3. Calculate the Landmark Position in the Robot's Frame

$$l_x^{\text{robot}} = r_{\text{measured}} \cos \beta_{\text{measured}}$$

$$l_y^{\text{robot}} = r_{\text{measured}} \sin \beta_{\text{measured}}$$

Substitute r_{measured} and β_{measured}:

$$l_x^{\text{robot}} = (r + n_r) \cos(\beta + n_\beta)$$

$$l_y^{\text{robot}} = (r + n_r) \sin(\beta + n_\beta)$$

4. Transform the Landmark Position to the Global Frame

$$\begin{bmatrix} l_x \\ l_y \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \end{bmatrix} + R_t \begin{bmatrix} l_x^{\text{robot}} \\ l_y^{\text{robot}} \end{bmatrix}$$

where the rotation matrix R_t is based on the robot's orientation θ_t:

$$R_t = \begin{bmatrix} \cos \theta_t & -\sin \theta_t \\ \sin \theta_t & \cos \theta_t \end{bmatrix}$$

5. Compute the Global Coordinates of the Landmark

$$l_x = x_t + \cos \theta_t \cdot l_x^{\text{robot}} - \sin \theta_t \cdot l_y^{\text{robot}}$$

$$l_y = y_t + \sin \theta_t \cdot l_x^{\text{robot}} + \cos \theta_t \cdot l_y^{\text{robot}}$$

Substitute l_x^{robot} and l_y^{robot}:

$$l_x = x_t + \cos \theta_t[(r + n_r) \cos(\beta + n_\beta)] - \sin \theta_t[(r + n_r) \sin(\beta + n_\beta)]$$

$$l_y = y_t + \sin \theta_t[(r + n_r) \cos(\beta + n_\beta)] + \cos \theta_t[(r + n_r) \sin(\beta + n_\beta)]$$

6. Simplify the Expressions Using Trigonometric Identities Using the identity $\cos A \cos B - \sin A \sin B = \cos(A + B)$:

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

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

7. Final Expression for the Landmark Position Thus, the estimated position (l_x, l_y) of the landmark l in global coordinates is:

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

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

Question 1.4: Predicted Measurements for Range and Bearing to a Landmark

[For detailed working of the mathematical derivation, please refer to the scanned work annexed with this answer] To predict the measurements of bearing and range to a landmark l located at global coordinates (l_x, l_y), given the robot's pose $p_t = \begin{bmatrix} x_t & y_t & \theta_t \end{bmatrix}^T$ and the noise terms $n_r \sim \mathcal{N}(0, \sigma_r^2)$ and $n_\beta \sim \mathcal{N}(0, \sigma_\beta^2)$.

1. Compute the Relative Position of the Landmark

$$\Delta x = l_x - x_t, \quad \Delta y = l_y - y_t$$

2. Calculate the True Range to the Landmark The true (noise-free) range r_{true} is:

$$r_{\text{true}} = \sqrt{(\Delta x)^2 + (\Delta y)^2}$$

3. Compute the True Bearing to the Landmark The true (noise-free) bearing β_{true} is:

$$\beta_{\text{true}} = \text{arctan2}(\Delta y, \Delta x) - \theta_t$$

4. Adjust the Bearing Angle to the Range $(-\pi, \pi]$ To ensure β_{true} is within $(-\pi, \pi]$, we apply the warp2pi$(\cdot)$ function:

$$\beta_{\text{true}} = \text{warp2pi}\left(\text{arctan2}(\Delta y, \Delta x) - \theta_t\right)$$

5. Incorporate the Measurement Noise The actual sensor measurements, which include noise, are modeled as:

- **Range Measurement:**

$$r = r_{\text{true}} + n_r = \sqrt{(\Delta x)^2 + (\Delta y)^2} + n_r$$

- **Bearing Measurement:**

$$\beta = \beta_{\text{true}} + n_\beta = \text{warp2pi}\left(\text{arctan2}(\Delta y, \Delta x) - \theta_t\right) + n_\beta$$

Final Expressions

Combining the above steps, we obtain the predicted measurements:

Predicted Range Measurement:

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

Predicted Bearing Measurement:

$$\beta = \text{warp2pi}\left(\text{arctan2}(l_y - y_t, l_x - x_t) - \theta_t\right) + n_\beta$$

Question 1.5: Derivation of the Measurement Jacobian H_p

[For detailed working of the mathematical derivation, please refer to the scanned work annexed with this answer] To derive the measurement Jacobian H_p with respect to the robot pose p_t in EKF-SLAM, we need to compute the partial derivatives of the measurement function $h(p_t, l)$ with respect to the components of p_t.

1. Define Variables

Robot Pose at Time t:

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

Landmark Position in Global Coordinates:

$$l = \begin{bmatrix} l_x \\ l_y \end{bmatrix}$$

Measurement Function $h(p_t, l)$: The measurement function maps the robot pose and landmark position to the predicted measurements (bearing β and range r) as follows:

$$h(p_t, l) = \begin{bmatrix} \beta \\ r \end{bmatrix} = \begin{bmatrix} \text{warp2pi}\left(\arctan2(\Delta y, \Delta x) - \theta_t\right) \\ \sqrt{\Delta x^2 + \Delta y^2} \end{bmatrix}$$

where:

$$\Delta x = l_x - x_t, \quad \Delta y = l_y - y_t$$

2. Compute Partial Derivatives for β

We express β directly as:

$$\beta = \text{warp2pi}\left(\arctan2(\Delta y, \Delta x) - \theta_t\right)$$

Partial Derivatives of β with Respect to x_t, y_t, and θ_t:

The partial derivatives are calculated as follows:

$$\frac{\partial \beta}{\partial x_t} = \frac{-\partial \beta}{\partial \Delta x} = \frac{\Delta y}{\Delta x^2 + \Delta y^2}$$

$$\frac{\partial \beta}{\partial y_t} = \frac{-\partial \beta}{\partial \Delta y} = -\frac{\Delta x}{\Delta x^2 + \Delta y^2}$$

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

3. Compute Partial Derivatives for r

We express r as:

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

The partial derivatives of r are:

$$\frac{\partial r}{\partial x_t} = -\frac{\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

$$\frac{\partial r}{\partial y_t} = -\frac{\Delta y}{\sqrt{\Delta x^2 + \Delta y^2}}$$

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

4. Assemble the Measurement Jacobian H_p

The measurement Jacobian H_p is a 2×3 matrix:

$$H_p = \begin{bmatrix} \frac{\partial \beta}{\partial x_t} & \frac{\partial \beta}{\partial y_t} & \frac{\partial \beta}{\partial \theta_t} \\ \frac{\partial r}{\partial x_t} & \frac{\partial r}{\partial y_t} & \frac{\partial r}{\partial \theta_t} \end{bmatrix} = \begin{bmatrix} \frac{\Delta y}{\Delta x^2 + \Delta y^2} & -\frac{\Delta x}{\Delta x^2 + \Delta y^2} & -1 \\ -\frac{\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}} & -\frac{\Delta y}{\sqrt{\Delta x^2 + \Delta y^2}} & 0 \end{bmatrix}$$

5. Express H_p in Terms of p_t and l

Using the definitions:

$$\Delta x = l_x - x_t, \quad \Delta y = l_y - y_t$$

The Jacobian H_p can be written explicitly as:

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

Question 1.6: Derivation of the Measurement Jacobian H_l

[For detailed working of the mathematical derivation, please refer to the scanned work annexed with this answer] To derive the measurement Jacobian H_l with respect to the landmark l in EKF-SLAM, we compute the partial derivatives of the measurement function $h(p_t, l)$ with respect to the components of l.

1. Define Variables

Robot Pose at Time t:

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

Landmark Position in Global Coordinates:

$$l = \begin{bmatrix} l_x \\ l_y \end{bmatrix}$$

Measurement Function $h(p_t, l)$: The measurement function maps the robot pose and landmark position to the predicted measurements (bearing β and range r):

$$h(p_t, l) = \begin{bmatrix} \beta \\ r \end{bmatrix} = \begin{bmatrix} \text{warp2pi}\left(\arctan2(\Delta y, \Delta x) - \theta_t\right) \\ \sqrt{\Delta x^2 + \Delta y^2} \end{bmatrix}$$

where:

$$\Delta x = l_x - x_t, \quad \Delta y = l_y - y_t$$

2. Compute Partial Derivatives of β with Respect to l_x and l_y

We directly express β in terms of Δx and Δy:

$$\beta = \text{warp2pi}\left(\arctan2(\Delta y, \Delta x) - \theta_t\right)$$

Partial Derivatives of β:

$$\frac{\partial \beta}{\partial l_x} = \frac{\partial \beta}{\partial \Delta x} \cdot \frac{\partial \Delta x}{\partial l_x} = \frac{-\Delta y}{\Delta x^2 + \Delta y^2}$$
$$\frac{\partial \beta}{\partial l_y} = \frac{\partial \beta}{\partial \Delta y} \cdot \frac{\partial \Delta y}{\partial l_y} = \frac{\Delta x}{\Delta x^2 + \Delta y^2}$$

3. Compute Partial Derivatives of r with Respect to l_x and l_y

Since r is defined as:

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

Partial Derivatives of r:

$$\frac{\partial r}{\partial l_x} = \frac{\partial r}{\partial \Delta x} \cdot \frac{\partial \Delta x}{\partial l_x} = \frac{\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$
$$\frac{\partial r}{\partial l_y} = \frac{\partial r}{\partial \Delta y} \cdot \frac{\partial \Delta y}{\partial l_y} = \frac{\Delta y}{\sqrt{\Delta x^2 + \Delta y^2}}$$

4. Assemble the Measurement Jacobian H_l

The measurement Jacobian H_l is a 2×2 matrix:

$$H_l = \begin{bmatrix} \frac{\partial \beta}{\partial l_x} & \frac{\partial \beta}{\partial l_y} \\ \frac{\partial r}{\partial l_x} & \frac{\partial r}{\partial l_y} \end{bmatrix} = \begin{bmatrix} \frac{-\Delta y}{\Delta x^2+\Delta y^2} & \frac{\Delta x}{\Delta x^2+\Delta y^2} \\ \frac{\Delta x}{\sqrt{\Delta x^2+\Delta y^2}} & \frac{\Delta y}{\sqrt{\Delta x^2+\Delta y^2}} \end{bmatrix}$$

5. Express H_l in Terms of p_t and l

Using $\Delta x = l_x - x_t$, $\Delta y = l_y - y_t$, and $r = \sqrt{\Delta x^2 + \Delta y^2}$:

$$H_l = \begin{bmatrix} \frac{-(l_y - y_t)}{(l_x - x_t)^2 + (l_y - y_t)^2} & \frac{l_x - x_t}{(l_x - x_t)^2 + (l_y - y_t)^2} \\ \frac{l_x - x_t}{\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}} & \frac{l_y - y_t}{\sqrt{(l_x - x_t)^2 + (l_y - y_t)^2}} \end{bmatrix}$$

Reason We Do Not Need to Calculate the Measurement Jacobian with Respect to Other Landmarks

We do not need to calculate the measurement Jacobian with respect to other landmarks because of the assumption of landmark independence. In EKF-SLAM, each landmark is assumed to be observed independently, and the measurement of one landmark does not depend on the positions of other landmarks. Therefore, the partial derivatives of the measurement function with respect to other landmarks are zero, and their Jacobians do not contribute to the update.

Question 2.1: Number of fixed landmarks

To determine the fixed number of landmarks being observed over the entire sequence, I analyzed the provided data file data/data.txt, and since the size of the measurement vector is 12 i.e. a pair of bearing and range measurements to each landmarks, hence the fixed number of landmarks being observed over the entire sequence is 6.

Question 2.2: EKF SLAM Visualization

To enable the system for EKF SLAM, I implemented the functions `warp2pi`, `init_landmarks`, `predict`, and `update` as outlined in the `ekf_slam.py` file. The visualization after the implementation can be seen below:

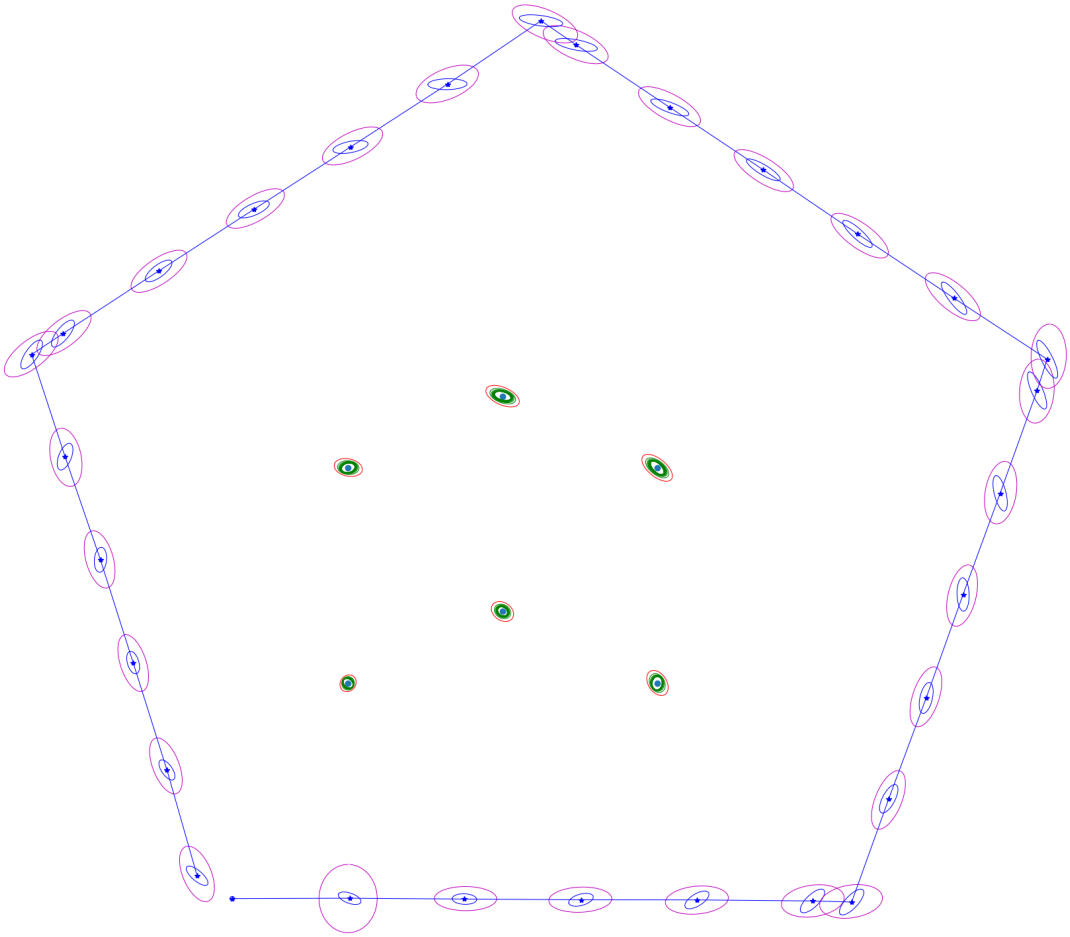


Figure 1: EKF SLAM trajectory and landmarks with covariance ellipses.

Question 2.3: EKF SLAM Improvement Analysis

In the Extended Kalman Filter (EKF) SLAM process, the interplay between the prediction and update steps critically improves both the robot's trajectory and the environmental map estimation. The magenta and blue ellipses in the visualization represent the predicted and updated uncertainties of the robot's position at each time step, respectively, while the red and green ellipses denote the initial and updated uncertainties of the landmarks.

During the prediction step, uncertainties increase due to the inherent inaccuracy in the robot's motion model and sensors. This is reflected by the larger size of the magenta ellipses. When the update step is executed, new measurements are incorporated, allowing the system to refine its estimates based on actual observations. This fusion of prediction and new sensor data results in smaller blue ellipses, indicating reduced uncertainty in the robot's position.

Similarly, the landmarks start with higher uncertainties (red ellipses) which progressively shrink into smaller green ellipses as the robot revisits these landmarks and updates their positions with new measurements. This repeated updating process accumulates evidence and progressively refines the landmark positions, significantly reducing their positional uncertainties over time.

The effective use of the Kalman gain in EKF SLAM dynamically balances the weight given to new measurements versus the predictions. This balance is crucial for adapting the filter's performance to the varying levels of uncertainty in sensor data and movement predictions. As such, through the continuous update cycle, EKF SLAM not only refines the estimates of the robot's trajectory but also improves the accuracy and reliability of the map it constructs.

Question 2.4: Evaluation of EKF-SLAM Implementation

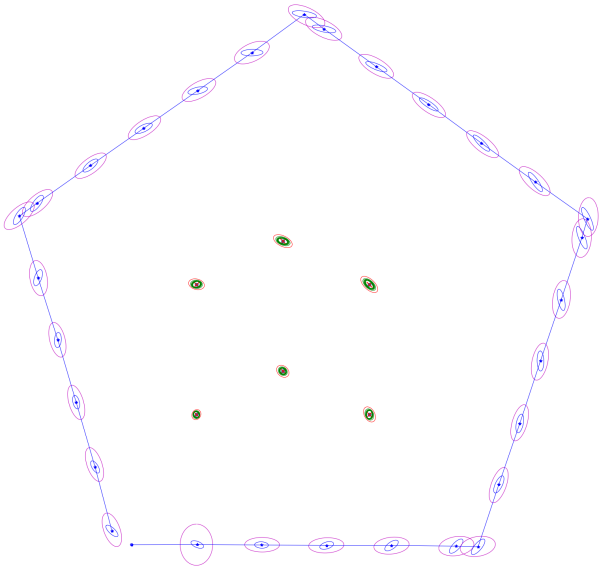
In this evaluation, we compared the estimated positions of the landmarks obtained from our EKF-SLAM implementation to the known ground truth positions. The results are summarized in the table below, showing both the Euclidean and Mahalanobis distances for each landmark.

Landmark	**Euclidean Distance**	**Mahalanobis Distance**
1	0.0021917	8.5549e-05
2	0.0041727	2.3433e-04
3	0.0025231	1.1250e-04
4	0.0027936	1.1257e-04
5	0.0019271	1.1666e-04
6	0.0039974	1.5250e-04

Table 1: Comparison of Euclidean and Mahalanobis distances for each landmark.

The Euclidean distances are very small, indicating that our estimated landmark positions are extremely close to the true positions. This demonstrates the high accuracy of our EKF-SLAM algorithm in estimating the landmarks' locations. The Mahalanobis distances take into account the uncertainty in our estimates. The low values suggest that the discrepancies between the estimated and true positions are within the expected uncertainty bounds. This indicates that our covariance estimates are reliable and that the EKF is correctly quantifying the uncertainty associated with each landmark.

From the visualization below, the estimated landmarks (red crosses) are nearly overlapping with the ground truth landmarks (blue circles). The uncertainty ellipses (green) encompass the true landmark positions, confirming that our uncertainty estimates are consistent with the actual errors. Importantly, each ground truth landmark position is inside the smallest corresponding uncertainty ellipse (green ellipses). This means that our estimated uncertainties (covariance ellipses) correctly encompass the true positions of the landmarks, confirming that our covariance estimates are reliable.



Question 3.1: Covariance Matrix Update in EKF-SLAM

The Extended Kalman Filter (EKF) SLAM algorithm dynamically updates the covariance matrix $\mathbf{P}$, introducing non-zero cross-covariances between the robot's pose and the landmarks. Initially, these cross-covariances are set to zero, implying independence among the pose and landmarks. However, as the robot moves and makes observations, this assumption no longer holds.

The update step uses the Kalman gain $\mathbf{K}$ and the measurement Jacobian $\mathbf{H}$ to update the covariance matrix:

$$\mathbf{P}_{\text{updated}} = (\mathbf{I} - \mathbf{KH})\mathbf{P}_{\text{predicted}}$$

Since $\mathbf{K}$ and $\mathbf{H}$ are generally dense matrices, their multiplication with $\mathbf{P}_{\text{predicted}}$ introduces non-zero elements in positions that were initially zero. This reflects the dependencies introduced by observations, linking the robot's pose with the landmarks and correlating different landmarks.

Initial Assumptions and Corrections

Now coming to the assumptions made on line 201, initially, we assume zero cross-covariances between the robot's pose and the landmarks, as well as among different landmarks. This simplification overlooks inherent dependencies due to:

- **Robot Pose and Landmark Dependence:** Landmarks are observed relative to the robot's pose; thus, uncertainties in the pose affect the estimated landmark positions, introducing cross-covariances.
- **Landmark Interdependence:** Measurements of different landmarks are taken from the same robot pose using the same sensors, leading to correlated measurement noise and shared uncertainties.

These initial assumptions may underestimate the actual uncertainties and dependencies in the system. By correctly initializing $\mathbf{P}$ with appropriate cross-covariances, we can more accurately represent the system's uncertainty, improving the EKF-SLAM's effectiveness in mapping and localization tasks.

Question 3.2: Abalation study of changing parameters

Following are the results and observations I made for the resulting plots by changing parameters in the main function (line 163-167):

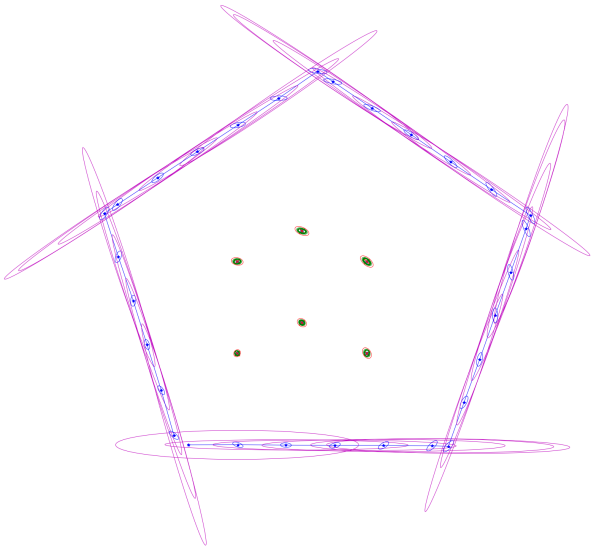


Figure 2: Increased σ_x by factor of 10

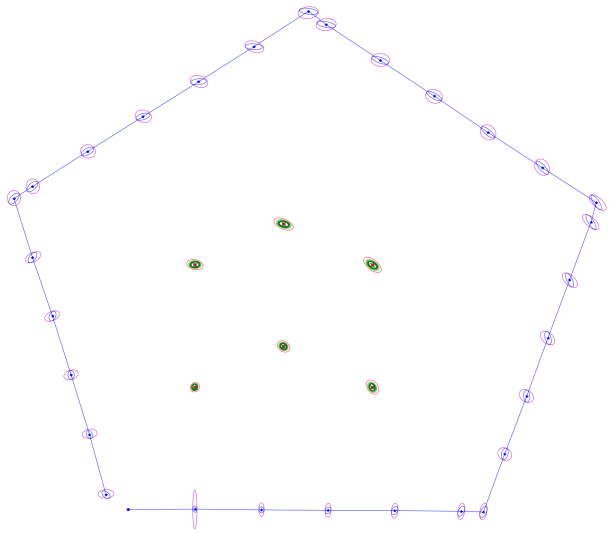


Figure 3: Decreased σ_x by factor of 10

As seen in figure 2, the trajectory's covariance ellipses are significantly elongated, indicating a large increase in uncertainty in the robot's x-coordinate. This elongation reflects lower confidence/belief in the robot's positional accuracy due to increased control noise, leading to a broader spread in position estimates.

Similarly, in figure 3, reduction in noise leads to much tighter covariance ellipses around the robot's trajectory, suggesting improved positional confidence and localization accuracy.

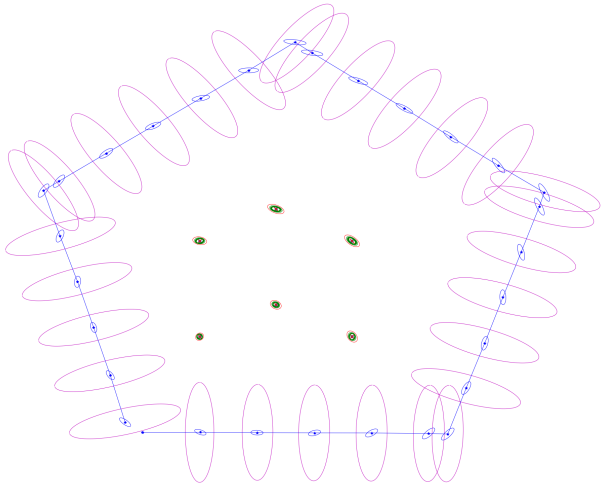


Figure 4: Increased σ_y by factor of 10

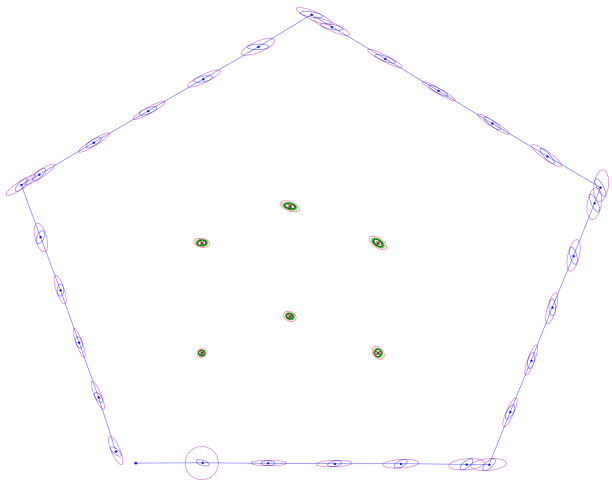


Figure 5: Decreased σ_y by factor of 10

As observed in Figure 4, increases in σ_y result in significantly elongated ellipses along the y-axis, reflecting a heightened uncertainty in the robot's position in this direction. This effect mirrors the increased uncertainty seen with σ_x, leading to a broader spatial spread.

In contrast, Figure 5 shows that a reduction in σ_y leads to much tighter ellipses, indicating enhanced positional certainty and better localization accuracy due to decreased noise.

The influence of σ_α on the visualization is subtle as seen in figure 6 and 7; changes in robot orientation are less noticeable in the 2D trajectory plots, as orientation is not explicitly visualized. Nonetheless, the expected consistency in visualization supports the assumption of minimal impact on observed positional uncertainty.

If the orientation was explicitly visualized, we would observe uncertaininity at sharp turns of the pentagon if the variance is increased, and we would observe the uncertainity go down when we scale it down.

As depicted in Figure 8, increasing σ_β results in more elongated green ellipses around the landmarks, indicating a rise in uncertainty with respect to the bearing measurements. This increase magnifies the estimation errors, leading to a broader dispersion in landmark positioning.

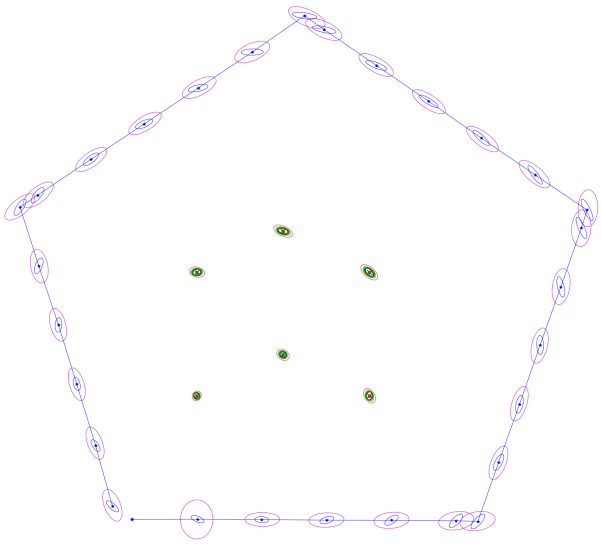


Figure 6: Increased σ_α by factor of 10

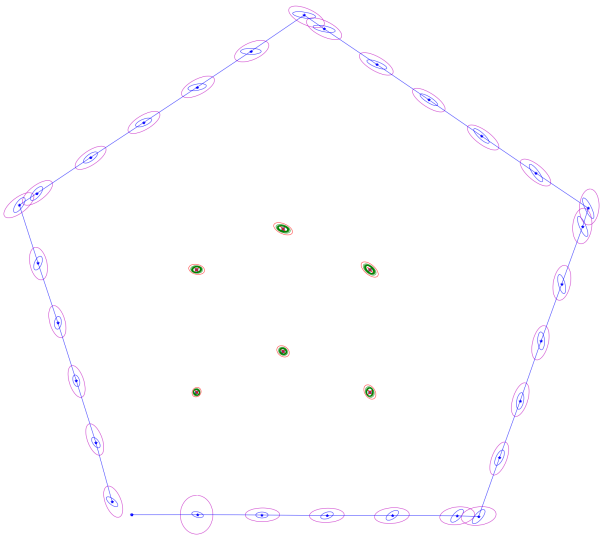


Figure 7: Decreased σ_α by factor of 10

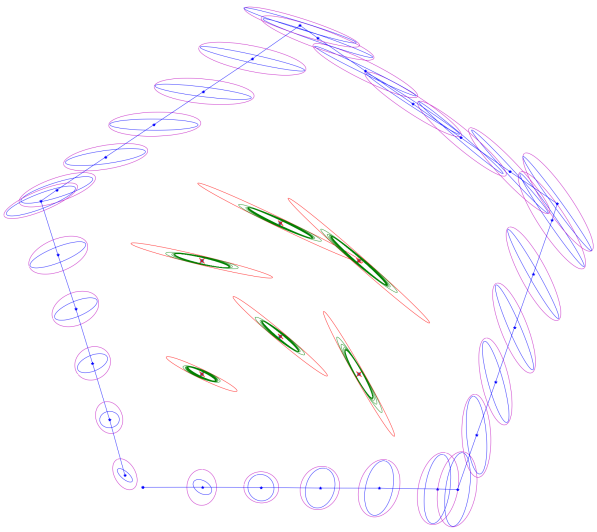


Figure 8: Increased σ_β by factor of 10

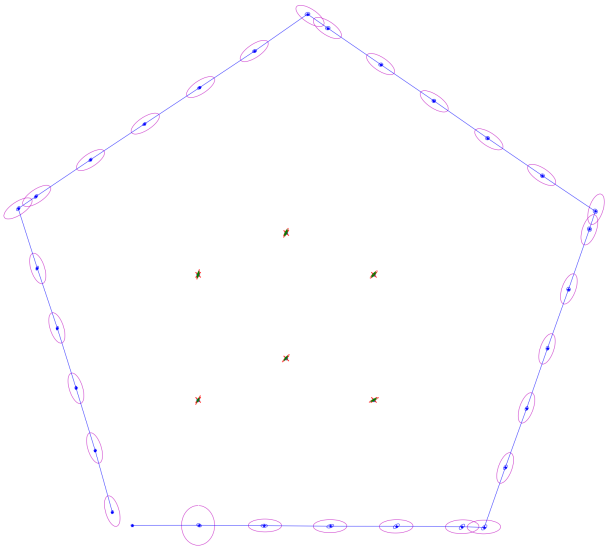


Figure 9: Decreased σ_β by factor of 10

Conversely, Figure 9 illustrates that decreasing σ_β significantly tightens these ellipses, suggesting a higher precision in landmark localization. This decrease implies an enhanced reliability of bearing measurements, resulting in more accurate landmark mapping.

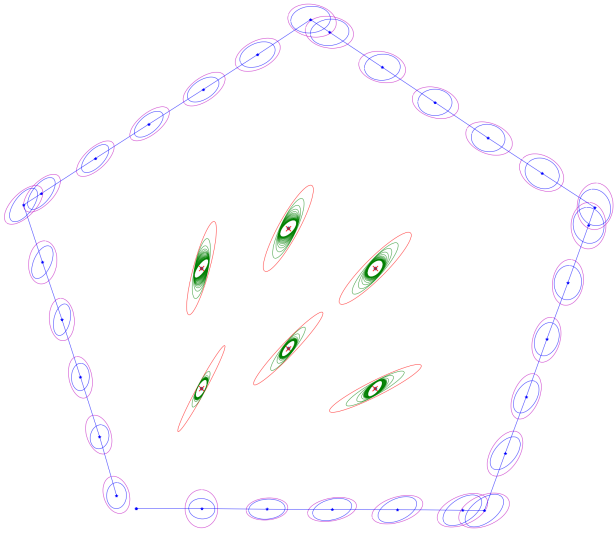


Figure 10: Increased σ_r by factor of 10

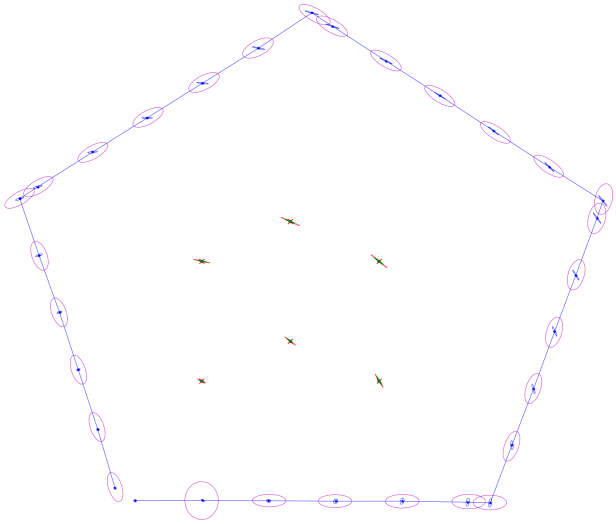


Figure 11: Decreased σ_r by factor of 10

Figure 10 shows that an increase in σ_r leads to larger red ellipses associated with the landmarks, reflecting a greater uncertainty in range measurements. This adjustment exacerbates the errors associated with the distance estimations from the robot to the landmarks.

In Figure 11, a reduction in σ_r causes these ellipses to become notably smaller, indicating a reduction in measurement noise and a corresponding increase in the accuracy of range estimations. The tightened ellipses depict an improvement in the precision of how landmarks are perceived spatially relative to the robot.

Question 3.3: Optimizing EKF-SLAM for Constant Computation Time

1. We can Restrict updates to landmarks within the robot's sensor range, minimizing the size of the state vector and covariance matrix. This selective updating ensures that computation time remains constant by focusing only on currently relevant landmarks.

2. By implementing strategies to prune or forget landmarks that contribute minimal information, such as those with high uncertainty or those not observed for extended periods. This approach helps maintain a manageable number of landmarks, preventing the state vector from becoming unwieldy.

3. We can utilize a multi-level mapping strategy, maintaining detailed local maps for navigation and a coarser global map for overall positioning. This hierarchical approach allows for switching between maps based on the robot's needs, balancing detail and computational efficiency.

4. Optimizing the data association process by employing advanced algorithms and structures like KD-trees for quick landmark matching. Predicting landmark positions based on motion models can also reduce the computational burden during data association.

Question 4: Code submission

Completed and finalized code file (ekf_slam.py) submitted on gradescope.

References

1. Thrun, S., Burgard, W., Fox, D. (2005). *Probabilistic Robotics*. MIT Press. (Chapter 7 on EKF SLAM with bearing and range measurements.)

2. Bailey, T., Durrant-Whyte, H. (2006). Simultaneous Localization and Mapping (SLAM): Part II State of the Art. *IEEE Robotics & Automation Magazine*, 13(3), 108–117.

3. Durrant-Whyte, H., Bailey, T. (2006). Simultaneous Localization and Mapping (SLAM): Part I The Essential Algorithms. *IEEE Robotics & Automation Magazine*, 13(2), 99–110.

4. Castellanos, J. A., Tardós, J. D. (1999). *Mobile Robot Localization and Map Building: A Multisensor Fusion Approach*. Kluwer Academic Publishers.

5. Huang, G., Mourikis, A. I., Roumeliotis, S. I. (2007). A First-Estimates Jacobian EKF for Improving SLAM Consistency. In *Proceedings of the IEEE International Conference on Robotics and Automation* (pp. 1901–1907). IEEE.

6. Smith, R. C., Cheeseman, P. (1986). On the Representation and Estimation of Spatial Uncertainty. *International Journal of Robotics Research*, 5(4), 56–68.

7. Leonard, J. J., Durrant-Whyte, H. F. (1992). *Directed Sonar Sensing for Mobile Robot Navigation*. Kluwer Academic Publishers.

8. Bar-Shalom, Y., Li, X. R., Kirubarajan, T. (2001). *Estimation with Applications to Tracking and Navigation*. Wiley-Interscience.

9. Julier, S. J., Uhlmann, J. K. (2001). A Counter Example to the Theory of Simultaneous Localization and Map Building. In *Proceedings of the IEEE International Conference on Robotics and Automation* (pp. 4238–4243). IEEE.

10. Newman, P., Leonard, J., Rikoski, R., Bosse, M. (2003). Towards Consistent Mapping and Localization Using Feature-Based Maps. *The International Journal of Robotics Research*, 23(1), 1–20.

Q 1.1:-

Since,

$$P_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \& \quad u_t = \begin{bmatrix} d_t \\ \alpha_t \end{bmatrix}$$

Since, there is no noise so,

$$\Delta x_t = d_t \cos\theta_t \quad \text{—①}$$

$$\Delta y_t = d_t \sin\theta_t \quad \text{—②}$$

$$\Delta\theta_t = \alpha_t \quad \text{—③}$$

So,

$$x_{t+1} = x_t + \Delta x_t$$

$$\boxed{x_{t+1} = x_t + d_t \cos\theta_t} \quad \text{(from eqn ①)}$$

$$y_{t+1} = y_t + \Delta y_t$$

$$\boxed{y_{t+1} = y_t + d_t \sin\theta_t} \quad \text{(from eqn ②)}$$

$$\boxed{\theta_{t+1} = \theta_t + \Delta\theta_t = \theta_t + \alpha_t} \quad \text{(from eqn ③)}$$

So,

$$P_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + d_t \cos\theta_t \\ y_t + d_t \sin\theta_t \\ \theta_t + \alpha_t \end{bmatrix}$$

Q1.2

Ans:- At time "t"

$$P_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \quad \& \quad u_t = \begin{bmatrix} d_t \\ \alpha_t \end{bmatrix}$$

Since, the errors are modeled as Zero mean Gaussian

$$W_t = \begin{bmatrix} e_x \\ e_y \\ e_\alpha \end{bmatrix}$$

where, $e_x \sim N(0, \sigma_x^2)$
$e_y \sim N(0, \sigma_y^2)$
$e_\alpha \sim N(0, \sigma_\alpha^2)$

so, motion model with errors become:-

$$P_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + \Delta x_{global} \\ y_t + \Delta y_{global} \\ \theta_t + \Delta \theta_{global} \end{bmatrix} \quad \text{---(1)}$$

Since,

$$\begin{bmatrix} \Delta x_{global} \\ \Delta y_{global} \end{bmatrix} = \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix} \begin{bmatrix} \Delta x_{robot} \\ \Delta y_{robot} \end{bmatrix}$$

Since

$\Delta x_{robot} = d_t + e_x$

$\Delta y_{robot} = e_y$

so,

$$\begin{bmatrix} \Delta x_{global} \\ \Delta y_{global} \end{bmatrix} = \begin{bmatrix} (d_t + e_x)\cos\theta_t - e_y \sin\theta_t \\ (d_t + e_x)\sin\theta_t + e_y \cos\theta_t \end{bmatrix}$$

$\Delta\theta_{global} = \alpha_t + e_\alpha$

so,

$$\textcircled{1} \Rightarrow P_{t+1} = \begin{bmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + (d_t + e_x)\cos\theta_t - e_y \sin\theta_t \\ y_t + (d_t + e_x)\sin\theta_t + e_y \cos\theta_t \\ \theta_t + \alpha_t + e_\alpha \end{bmatrix}$$

now, to propagate the uncertainity :-

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

$$\frac{\partial x_{t+1}}{\partial x_t} = 1$$

$$\frac{\partial x_{t+1}}{\partial y_t} = 0$$

$$\frac{\partial x_{t+1}}{\partial \theta_t} = -(d_t + e_x)\sin\theta_t - e_y \cos\theta_t = -d_t \sin\theta_t \quad (\text{for } e_x = 0, e_y = 0)$$

$$\frac{\partial y_{t+1}}{\partial x_t} = 0$$

$$\frac{\partial y_{t+1}}{\partial y_t} = 1$$

$$\frac{\partial y_{t+1}}{\partial \theta_t} = (d_t + e_x)\cos\theta_t - e_y \sin\theta_t = d_t \cos\theta_t \quad (\text{for } e_x = 0, e_y = 1$$

$$\frac{\partial \theta_{t+1}}{\partial x_t} = 0$$

$$\frac{\partial \theta_{t+1}}{\partial y_t} = 0$$

$$\frac{\partial \theta_{t+1}}{\theta_t} = 1$$

So,

$$F_t = \begin{bmatrix} 1 & 0 & -d_t \sin\theta_t \\ 0 & 1 & d_t \cos\theta_t \\ 0 & 0 & 1 \end{bmatrix}$$

now, Jacobian w.r.t process noise becomes

$$G_t = \begin{bmatrix} \frac{\partial x_{t+1}}{\partial e_x} & \frac{\partial x_{t+1}}{\partial e_y} & \frac{\partial x_{t+1}}{\partial e_\alpha} \\ \frac{\partial y_{t+1}}{\partial e_x} & \frac{\partial y_{t+1}}{\partial e_y} & \frac{\partial y_{t+1}}{\partial e_\alpha} \\ \frac{\partial \theta_{t+1}}{\partial e_x} & \frac{\partial \theta_{t+1}}{\partial e_y} & \frac{\partial \theta_{t+1}}{\partial e_\alpha} \end{bmatrix}$$

$$\frac{\partial x_{t+1}}{\partial e_x} = \cos\theta_t \qquad \frac{\partial y_{t+1}}{\partial e_x} = \sin\theta_t \qquad \frac{\partial \theta_{t+1}}{\partial e_x} = 0$$

$$\frac{\partial x_{t+1}}{\partial e_y} = -\sin\theta_t \qquad \frac{\partial y_{t+1}}{\partial e_y} = \cos\theta_t \qquad \frac{\partial \theta_{t+1}}{\partial e_y} = 0$$

$$\frac{\partial x_{t+1}}{\partial e_\alpha} = 0 \qquad \frac{\partial y_{t+1}}{\partial e_\alpha} = 0 \qquad \frac{\partial \theta_{t+1}}{\partial e_\alpha} = 1$$

$$G_t = \begin{bmatrix} \cos\theta_t & -\sin\theta_t & 0 \\ \sin\theta_t & \cos\theta_t & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

now, process noise covariance becomes

$$Q_t = \begin{bmatrix} \text{var}(e_x) & \text{cov}(e_x, e_y) & \text{cov}(e_x, e_\alpha) \\ \text{cov}(e_y, e_x) & \text{var}(e_y) & \text{cov}(e_y, e_\alpha) \\ \text{cov}(e_\alpha, e_x) & \text{cov}(e_\alpha, e_y) & \text{var}(e_\alpha) \end{bmatrix}$$

Since, the errors are independent:-

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

so, predicted covariance Σ_{t+1} becomes:-

$$\Sigma_{t+1} = \underbrace{F_t \, \Sigma_t \, F_t^T}_{\text{previous uncertainty}} + \underbrace{G_t \, Q_t \, G_t^T}_{\text{added process noise uncertainty}}$$

Q 1·3 :-

Ans :- Since,

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

noisy bearing angle :-

$$\beta_m = \beta + \eta_\beta$$

noisy range measurement :-

$$r_m = r + \eta_r$$

So,

$$L_x^{robot} = r_m \cos\beta_m = (r + \eta_r)\cos(\beta + \eta_\beta) \quad \text{—— (1)}$$

$$L_y^{robot} = r_m \sin\beta_m = (r + \eta_r)\sin(\beta + \eta_\beta) \quad \text{—— (2)}$$

now to convert position from robot's frame to global frame :-

$$\begin{bmatrix} L_x \\ L_y \end{bmatrix} = \begin{bmatrix} x_t \\ y_t \end{bmatrix} + R_t \begin{bmatrix} L_x^{robot} \\ L_y^{robot} \end{bmatrix}$$

where,

$$R_t = \begin{bmatrix} \cos\theta_t & -\sin\theta_t \\ \sin\theta_t & \cos\theta_t \end{bmatrix}$$

So,

$$\begin{bmatrix} L_x \\ L_y \end{bmatrix} = \begin{bmatrix} x_t + \cos\theta_t \; L_x^{robot} - \sin\theta_t \; L_y^{robot} \\ y_t + \sin\theta_t \; L_y^{robot} + \cos\theta_t \; L_y^{robot} \end{bmatrix}$$

plugging in ① and ②

$$L_x = x_t + \cos\theta_t\left[(r+n_r)\cos(\beta+n_\beta)\right] - \sin\theta_t\left[(r+n_r)\sin(\beta+n_\beta)\right]$$

$$L_x = x_t + (r+n_r)\left[\cos\theta_t\cos(\beta+n_\beta) - \sin\theta_t\sin(\beta+n_\beta)\right]$$

$$\boxed{L_x = x_t + (r+n_r)\cos(\theta_t+\beta+n_\beta)}$$

$$L_y = y_t + \sin\theta_t\left[(r+n_r)\cos(\beta+n_\beta)\right] + \cos\theta_t\left[(r+n_r)\sin(\beta+n_\beta)\right]$$

$$L_y = y_t + (r+n_r)\left[\sin\theta_t\cos(\beta+n_\beta) + \cos\theta_t\sin(\beta+n_\beta)\right]$$

$$\boxed{L_y = y_t + (r+n_r)\sin(\theta_t+\beta+n_\beta)}$$

using trignometric identity :-

$$\cos\alpha\cos\beta - \sin\alpha\sin\beta = \cos(\alpha+\beta)$$

$$\sin\alpha\cos\beta + \cos\alpha\sin\beta = \sin(\alpha+\beta)$$

Q1.4

Ans:- Since,

$$\Delta x = l_x - x_t \quad \text{—(1)}$$
$$\Delta y = l_y - y_t \quad \text{—(2)}$$

Also,

$$r_{true} = \sqrt{\Delta x^2 + \Delta y^2}$$

$$\beta_{true} = \arctan2(\Delta y, \Delta x) - \theta_t$$

$$\beta_{true} = \text{wrap2Pi}(\arctan2(\Delta y, \Delta x) - \theta_t)$$

now adding noise:

$$r = r_{true} + n_r = \sqrt{\Delta x^2 + \Delta y^2} + n_r$$
$$\beta = \text{wrap2Pi}(\arctan2(\Delta y, \Delta x) - \theta_t) + n_\beta$$

Plugging in (1) and (2)

$$r = \sqrt{(l_x - x_t)^2 + (l_y - y_t)^2} + n_r$$
$$\beta = \text{wrap2Pi}(\arctan2(l_y - y_t, l_x - x_t) - \theta_t) + n_\beta$$

Q1.5

Ans:- Since,

$$P_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix}, \quad l = \begin{bmatrix} l_x \\ l_y \end{bmatrix}$$

measurement function $h(P_t, l)$:-

$$h(P_t, l) = \begin{bmatrix} \beta \\ r \end{bmatrix}$$

from 1.4:-

β = ~~then~~ wrap2Pi $(\arctan 2(\Delta y, \Delta x) - \theta_t) + \eta_\beta = \tan^{-1}\left(\frac{\Delta y}{\Delta x}\right) - \theta_t + \eta_\beta$ —①

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

Since,

$\Delta x = l_x - x_t$ —③

$\Delta y = l_y - y_t$ —④

plugging ③ & ④ to ① & ②

$$\beta = \tan^{-1}\left(\frac{l_y - y_t}{l_x - x_t}\right) - \theta_t + \eta_\beta \quad —⑤$$

$$r = \sqrt{(l_x - x_t)^2 + (l_y - y_t)^2} + \eta_r \quad —⑥$$

$$H_p(P_t, l) = \begin{bmatrix} \frac{\partial \beta}{\partial x_t} & \frac{\partial \beta}{\partial y_t} & \frac{\partial \beta}{\partial \theta_t} \\ \frac{\partial r}{\partial x_t} & \frac{\partial r}{\partial y_t} & \frac{\partial r}{\partial \theta_t} \end{bmatrix} \quad —⑦$$

deriving ⑤ and ⑥ w.r.t. variables x_t, y_t, θ_t

$$\frac{\partial \beta}{\partial x_t} = \frac{1}{1+\frac{\Delta y^2}{\Delta x^2}} \left(\frac{-\Delta y}{\Delta x^2}\right)(-1) = \frac{\Delta y}{\Delta x^2+\Delta y^2}$$

$$\frac{\partial \beta}{\partial y_t} = \frac{1}{1+\frac{\Delta y^2}{\Delta x^2}} \left(\frac{1}{\Delta x}\right)(-1) = \frac{-\Delta x}{\Delta x^2+\Delta y^2}$$

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

$$\frac{\partial r}{\partial x_t} = \frac{1}{2\sqrt{\Delta x^2+\Delta y^2}} (2\Delta x)(-1) = \frac{-\Delta x}{\sqrt{\Delta x^2+\Delta y^2}}$$

$$\frac{\partial r}{\partial y_t} = \frac{1}{2\sqrt{\Delta x^2+\Delta y^2}} (2\Delta y^2)(-1) = \frac{-\Delta y}{\sqrt{\Delta x^2+\Delta y^2}}$$

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

⑦ $\Rightarrow H_p(P_t, l) = \begin{bmatrix} \frac{\Delta y}{\Delta x^2+\Delta y^2} & \frac{-\Delta x}{\Delta x^2+\Delta y^2} & -1 \\ \frac{-\Delta x}{\sqrt{\Delta x^2+\Delta y^2}} & \frac{-\Delta y}{\sqrt{\Delta x^2+\Delta y^2}} & 0 \end{bmatrix}$

Q1.6

Ans:- $$H_L = \begin{bmatrix} \frac{\partial \beta}{\partial l_x} & \frac{\partial \beta}{\partial l_y} \\ \frac{\partial r}{\partial l_x} & \frac{\partial r}{\partial l_y} \end{bmatrix}$$

$$\frac{\partial \beta}{\partial l_x} = \frac{1}{1+\frac{\Delta y^2}{\Delta x^2}} \left(\frac{-\Delta y}{\Delta x^2}\right)(1) = \frac{-\Delta y}{\Delta x^2+\Delta y^2}$$

$$\frac{\partial \beta}{\partial l_y} = \frac{1}{1+\frac{\Delta y^2}{\Delta x^2}} \left(\frac{1}{\Delta x}\right)(1) = \frac{\Delta x}{\Delta x^2+\Delta y^2}$$

$$\frac{\partial r}{\partial l_x} = \frac{1}{2\sqrt{\Delta x^2+\Delta y^2}} (2\Delta x)(1) = \frac{\Delta x}{\sqrt{\Delta x^2+\Delta y^2}}$$

$$\frac{\partial r}{\partial l_y} = \frac{1}{2\sqrt{\Delta x^2+\Delta y^2}} (2\Delta y)(1) = \frac{\Delta y}{\sqrt{\Delta x^2+\Delta y^2}}$$

$$H_L = \begin{bmatrix} \frac{-\Delta y}{\Delta x^2+\Delta y^2} & \frac{\Delta x}{\Delta x^2+\Delta y^2} \\ \frac{\Delta x}{\sqrt{\Delta x^2+\Delta y^2}} & \frac{\Delta y}{\sqrt{\Delta x^2+\Delta y^2}} \end{bmatrix}$$