Exercise Solutions for Ex 04

Abdulrahman Hameshli

Matrikelnummer: 3768284

Solution

Forward Model First and Second Moments

- **Average Distance:**

$$d = \frac{s_R + s_L}{2}$$

- **Change in Orientation:**

$$\Delta\Theta = \frac{s_R - s_L}{b}$$

- **New Pose:**

$$\begin{aligned}
\Theta_{\text{out}} &= \Theta_0 + \Delta\Theta \\
X_{\text{out}} &= X_0 + \frac{d}{\Delta\Theta}\left(\sin(\Theta_{\text{out}}) - \sin(\Theta_0)\right) \\
Y_{\text{out}} &= Y_0 - \frac{d}{\Delta\Theta}\left(\cos(\Theta_{\text{out}}) - \cos(\Theta_0)\right)
\end{aligned}$$

For $\Delta\Theta \approx 0$:

$$X_{\text{out}} = X_0 + d\cos(\Theta_{\text{out}}), \quad Y_{\text{out}} = Y_0 + d\sin(\Theta_{\text{out}})$$

Covariance Update

$$\mathbf{covOut} = \mathbf{F}_x \mathbf{C}_{\text{start}} \mathbf{F}_x^\top + \mathbf{F}_u \mathbf{C}_u \mathbf{F}_u^\top$$

where:

$$\mathbf{F}_x = \mathbf{I}_3, \quad \mathbf{F}_u = \begin{bmatrix} \cos\left(\Theta_{\text{out}} + \frac{\Delta\Theta}{2}\right) & 0 \\ \sin\left(\Theta_{\text{out}} + \frac{\Delta\Theta}{2}\right) & 0 \\ 0 & 1 \end{bmatrix}$$

Arc Points Calculation

- **Straight Line:**

$$\text{arcPoints}(:, i) = \begin{bmatrix} X_0 + (i-1)\frac{d}{n_{\text{ArcPoints}}-1}\cos(\Theta_0) \\ Y_0 + (i-1)\frac{d}{n_{\text{ArcPoints}}-1}\sin(\Theta_0) \end{bmatrix}$$

- **Arc Case:**

$$\text{arcPoints}(:, i) = \begin{bmatrix} X_0 + \frac{d}{\Delta\Theta}\left(\sin(\theta_{\text{Arc}}) - \sin(\Theta_0)\right) \\ Y_0 - \frac{d}{\Delta\Theta}\left(\cos(\theta_{\text{Arc}}) - \cos(\Theta_0)\right) \end{bmatrix}$$

  where:

$$\theta_{\text{Arc}} = \Theta_0 + (i-1)\frac{\Delta\Theta}{n_{\text{ArcPoints}} - 1}$$

Monte Carlo Sampling for Pose Propagation

The function uses Monte Carlo sampling to estimate the propagated pose and its covariance. Given n_{samples}, initial pose $\mathbf{x}$, covariance $\mathbf{C}$, and base width b, it proceeds as follows:

Noisy Samples Generation

- **Noisy State:**

$$\mathbf{x}_{\text{noisy}} = \mathbf{x} + \mathbf{L} \cdot \mathbf{w}$$

  where $\mathbf{L} = \text{chol}(\mathbf{C})$ and $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

Pose Propagation for Each Sample

For each sample, the following calculations are performed:

- **Change in Orientation:**

$$\Delta\Theta = \frac{s_R - s_L}{b}$$

  where s_L and s_R are noisy wheel displacements.

- **Average Distance Traveled:**

$$d = \frac{s_L + s_R}{2}$$

- **New Pose:**

$$\theta_{\text{out}} = \theta_0 + \Delta\Theta$$

$$x_{\text{out}} = \begin{cases} x_0 + d\cos(\theta_0), & \text{if } |\Delta\Theta| < 10^{-6} \\ x_0 + \frac{d}{\Delta\Theta}\left(\sin(\theta_{\text{out}}) - \sin(\theta_0)\right), & \text{otherwise} \end{cases}$$

$$y_{\text{out}} = \begin{cases} y_0 + d\sin(\theta_0), & \text{if } |\Delta\Theta| < 10^{-6} \\ y_0 - \frac{d}{\Delta\Theta}\left(\cos(\theta_{\text{out}}) - \cos(\theta_0)\right), & \text{otherwise} \end{cases}$$

Mean Pose and Covariance Calculation

- **Mean Pose:**

$$\textbf{poseOut} = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} \mathbf{x}_{\text{out}}^{(i)}$$

- **Covariance of Pose Samples:**

$$\textbf{covOut} = \text{cov}(\textbf{samples}^\top)$$

Unscented Transform for Covariance Propagation

This function utilizes the **Unscented Transform (UT)** to propagate a robot's pose $\mathbf{x}$ and its covariance matrix $\mathbf{C}$, given base width b.

Sigma Points Calculation

- **State Dimension:** $n = \text{length}(\mathbf{x})$
- **Weights:**

$$w_0 = \frac{1}{3}, \quad w_i = \frac{1 - w_0}{2n}, \quad \gamma = \sqrt{\frac{n}{1 - w_0}}$$

- **Sigma Points:**

$$\text{cholC} = \text{chol}(\mathbf{C}) \quad \text{(Cholesky decomposition)}$$

$$\text{Sigma Points: } \mathbf{X} = [\mathbf{x}, \mathbf{x} + \gamma\mathbf{L}, \mathbf{x} - \gamma\mathbf{L}]$$

where $\mathbf{L}$ is the lower triangular matrix from $\text{chol}(\mathbf{C})$.

Transforming Sigma Points

For each sigma point, the function propagates the pose as follows:

- **Average Distance:**

$$d = \frac{s_R + s_L}{2}$$

- **Change in Orientation:**

$$\Delta\Theta = \frac{s_R - s_L}{b}$$

- **New Pose:**

$$\Theta_{\text{out}} = \Theta_0 + \Delta\Theta$$

$$X_{\text{out}} = \begin{cases} X_0 + d\cos(\Theta_0), & \text{if } |\Delta\Theta| < 10^{-6} \\ X_0 + \frac{d}{\Delta\Theta}\left(\sin(\Theta_{\text{out}}) - \sin(\Theta_0)\right), & \text{otherwise} \end{cases}$$

$$Y_{\text{out}} = \begin{cases} Y_0 + d\sin(\Theta_0), & \text{if } |\Delta\Theta| < 10^{-6} \\ Y_0 - \frac{d}{\Delta\Theta}\left(\cos(\Theta_{\text{out}}) - \cos(\Theta_0)\right), & \text{otherwise} \end{cases}$$

Mean and Covariance Calculation

- **Mean Pose:**

$$\textbf{poseOut} = w_0 \mathbf{Y}_1 + \sum_{i=2}^{2n+1} w_i \mathbf{Y}_i$$

  where $\mathbf{Y}_i$ are the transformed sigma points.

- **Covariance of Pose:**

$$\textbf{covOut} = w_0(\mathbf{Y}_1 - \textbf{poseOut})(\mathbf{Y}_1 - \textbf{poseOut})^\top + \sum_{i=2}^{2n+1} w_i(\mathbf{Y}_i - \textbf{poseOut})(\mathbf{Y}_i - \textbf{poseOut})^\top$$

Calibration of K

The ‘calibK‘ function is used to **calibrate** the growth coefficient K for odometry error propagation. The calibration is done through a series of simulation runs to ensure the coefficient accurately models the growth of uncertainty in the odometry-based pose estimates.

Purpose

The growth coefficient K determines how uncertainty in odometry accumulates over time. Proper calibration is necessary to ensure that the covariance of the pose estimate appropriately represents the actual error, allowing for robust state estimation. This calibration ensures that the odometry model is neither underestimating nor overestimating the uncertainty.

Input Parameters

- K: Initial, possibly under-approximated, growth coefficient.
- $nRuns$: Number of iterations to run the simulation.
- `visualize`: Boolean flag indicating whether to visualize the calibration process.

Calibration Process

1. **Threshold Definition**:
   - A **chi-square threshold** is calculated based on a 95% confidence level to determine if the covariance estimate is consistent with the actual error:

$$\text{chi2Threshold} = \chi^2_{\text{inv}}(0.95, 2)$$

2. **Simulation Loop** (Repeated for $nRuns$):

(a) Run an **odometry simulation** (‘OdometrySim‘) to generate pose estimates and ground truth values.

(b) Extract the following:

- Estimated pose (p_N) from odometry.
- Estimated covariance (Σ_k).
- Ground truth pose (p_{GT}).

(c) **Calculate Error**:

- Calculate the **position error** ($\delta p = p_{GT} - p_N$).
- Compute the **Mahalanobis distance** squared:

$$D^2 = \delta p^T \Sigma_k^{-1} \delta p$$

This distance measures how well the predicted uncertainty matches the observed deviation.

(d) **Compare to Threshold**:

- If $D^2 <$ chi2Threshold, it means the predicted uncertainty (using the current value of K) is consistent with the ground truth.
- If $D^2 \geq$ chi2Threshold, it indicates that the predicted uncertainty is too small, suggesting that K is underestimating the error growth. In this case, K is increased by a factor of 2.5 to compensate.

$$K = K \times 2.5$$

3. **Result**:

- After all simulation runs, the **calibrated value of K** is printed. This value should provide a more realistic representation of the growth in uncertainty during odometry.

Why This Works

The calibration procedure is designed to adjust the growth coefficient K until the **Mahalanobis distance** (which measures the error relative to the predicted uncertainty) falls within an acceptable statistical range (based on the chi-square test). The chi-square threshold represents a 95% confidence region for the true state, meaning that the predicted uncertainty should cover the actual error 95% of the time.

By comparing the Mahalanobis distance to this threshold over multiple runs:

- If the Mahalanobis distance is consistently too high, it implies that the model is **underestimating uncertainty**. Thus, K is increased to better capture the actual error growth.
- If the distance is below the threshold, it indicates that the current value of K is appropriate.

This iterative approach ensures that K is adjusted to accurately reflect the **real-world growth of uncertainty**, improving the reliability of subsequent state estimation processes.

Yes, systematic errors play a role by introducing consistent biases in the estimation process:

- **Biased Pose Estimates**: Systematic errors, such as sensor biases or calibration issues, lead to consistent drift in the estimated pose $(X_{\text{out}}, Y_{\text{out}}, \Theta_{\text{out}})$.
- **Incorrect Covariance Representation**: The uncertainty (**covOut**) may be underestimated since systematic errors are not random, giving a false sense of precision.
- **Error Accumulation**: Systematic biases accumulate over time, resulting in increasingly inaccurate pose estimates.

Calibration and bias correction are essential to mitigate these effects and improve the accuracy of the estimation.