MTRN4010 Project 2

z5257541 - Marcus Oates

Term 1 2023

Contents

Part A2

Prediction Step

We have the following prediction step,

$$\hat{\mathbf{x}}(k+1 \mid k) = \mathbf{f}(\hat{\mathbf{x}}(k \mid k), \mathbf{u}(k)) + \zeta_u(k),$$
$$\mathbf{P}(k+1 \mid k) = \mathbf{J}(k) \cdot \mathbf{P}(k \mid k) \cdot \mathbf{J}(k)^T + \mathbf{Q_u}(k),$$

where $\zeta_u(k)$ is the White Gaussian Noise of the inputs such that $\zeta_u(k) \sim \mathcal{N}(0, \mathbf{Q_u})$.

For the expected value prediction we have,

$$\hat{\mathbf{x}}(k+1 \mid k) = \begin{bmatrix} x(k) \\ y(k) \\ \phi(k) \end{bmatrix} + \tau \cdot \begin{bmatrix} v(k) \cdot \cos(\phi(k)) \\ v(k) \cdot \sin(\phi(k)) \\ w(k) \end{bmatrix},$$

where τ is the sampling period.

For the covariance matrix prediction we have,

$$\mathbf{J}(k) = \frac{\partial \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k))}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & -\tau \cdot v(k) \cdot \sin(\phi(k)) \\ 0 & 1 & \tau \cdot v(k) \cdot \cos(\phi(k)) \\ 0 & 0 & 1 \end{bmatrix},$$
$$\mathbf{J_u}(k) = \frac{\partial \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k))}{\partial \mathbf{u}} = \tau \cdot \begin{bmatrix} \cos(\phi(k)) & 0 \\ \sin(\phi(k)) & 0 \\ 0 & 1 \end{bmatrix},$$
$$\mathbf{P_u} = \begin{bmatrix} 0.1^2 & 0 \\ 0 & (4 \cdot \frac{\pi}{180})^2 \end{bmatrix},$$
$$\mathbf{Q_u}(k) = \mathbf{J_u}(k) \cdot \mathbf{P_u}(k) \cdot \mathbf{J_u}(k)^T,$$
$$\mathbf{P}(k+1 \mid k) = \mathbf{J}(k) \cdot \mathbf{P}(k) \cdot \mathbf{J}(k)^T + \mathbf{Q_u}(k).$$

where $\mathbf{P}(k)$ is the covariance matrix at discrete time k. Note that since the platforms pose is accurately known we have the following initial conditions,

$$\hat{\mathbf{x}}(0 \mid 0) = \begin{bmatrix} x_0 \\ y_0 \\ \phi_0 \end{bmatrix},$$

$$\mathbf{P}(0 \mid 0) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Observation Function

The output variables for the update step are the detected coordinates of the landmarks $(\mathrm{OOI}_x, \mathrm{OOI}_y)$, if any are detected, in the LiDAR's coordinate frame (LCF).

To get the relationship between the platforms pose and the detected OOIs we must transform the platforms pose from the global coordinate frame (GCF) to LCF, with an intermediate coordinate frame of the car (CCF). So we have,

$$\begin{aligned} p^{GCF} &= R_\alpha p^{CCF} + T_{CCF}, \\ p^{CCF} &= R_0 p^{LCF} + T_{LCF}. \end{aligned}$$

Rearranging gives us,

$$\begin{aligned} p^{CCF} &= R_\alpha^{-1}(p^{GCF} - T_{CCF}), \\ p^{LCF} &= R_0^{-1}(p^{CCF} - T_{LCF}). \end{aligned}$$

Substituting for p^{CCF} renders,

$$p^{LCF} = R_0^{-1}(R_\alpha^{-1}(p^{GCF} - T_{CCF}) - T_{LCF}),$$

where we have the following data,

$$\begin{aligned}
\alpha = \phi - \frac{\pi}{2} = &\text{ rotation of CCF,} \\
p^{GCF} &= (x_l, y_l) \text{ known landmarks in GCF,} \\
T_{CCF} &= (x, y) \text{ cars position in GCF,} \\
T_{LCF} &= (L_x, L_y) \text{ LiDAR's position in CCF,} \\
p^{LCF} &= \begin{bmatrix} \mathrm{OOI}_x \\ \mathrm{OOI}_y \end{bmatrix}.
\end{aligned}$$

As such we have the following observation function,

$$\begin{bmatrix} \mathrm{OOI}_x \\ \mathrm{OOI}_y \end{bmatrix} = \mathbf{h}(\mathbf{x}) = \begin{bmatrix} (x_l - x)\cos(\alpha) + (y_l - y)\sin(\alpha) - L_x \\ (y_l - y)\cos(\alpha) - (x_l - x)\sin(\alpha) - L_y \end{bmatrix}.$$

Proposed H Matrix

The H matrix is the Jacobian matrix of function $\mathbf{h}(\mathbf{x})$, so we have,

$$H = \begin{bmatrix} -\cos(\alpha) & -\sin(\alpha) & -(x_l - x)\sin(\alpha) + (y_l - y)\cos(\alpha) \\ \sin(\alpha) & -\cos(\alpha) & -(y_l - y)\sin(\alpha) - (x_l - x)\cos(\alpha) \end{bmatrix}.$$

Proposed R Matrix

We are given the pollution of the OOI measurements, such that,

$$R = \begin{bmatrix} 0.2^2 & 0 \\ 0 & 0.2^2 \end{bmatrix}.$$

Part A3

For this section of the project we consider the case where both components of the LiDAR's position on the car's platform are unknown, and we extend Part A1.

State Vector

The state vector is given by,

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ \phi \\ L_x \\ L_y \end{bmatrix}.$$

Initialization

Since we perfectly know the platforms pose, but are unsure about the actual position of the LiDAR we set the following initial expected value and covariance matrix,

$$\mathbf{x}(0 \mid 0) = \begin{bmatrix} x_0 \\ y_0 \\ \phi_0 \\ 0.3 \\ 0.1 \end{bmatrix},$$

$$\mathbf{P}(0 \mid 0) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3^2 & 0 \\ 0 & 0 & 0 & 0 & 0.2^2 \end{bmatrix},$$

where (x_0, y_0, ϕ_0) is the initial pose of the platform, which is given.

Prediction Step

The prediction step is given by,

$$\hat{\mathbf{x}}(k+1 \mid k) = \begin{bmatrix} x \\ y \\ \phi \\ L_x \\ L_y \end{bmatrix} + \tau \cdot \begin{bmatrix} v(k) \cdot \cos(\phi(k)) \\ v(k) \cdot \sin(\phi(k)) \\ w(k) \\ 0 \\ 0 \end{bmatrix},$$

$$\mathbf{P}(k+1 \mid k) = \mathbf{J}(k) \cdot \mathbf{P}(k) \cdot \mathbf{J}(k)^T + \mathbf{Q}_{\mathbf{u}}(k),$$

where we have,

$$\mathbf{J}(k) = \frac{\partial \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k))}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & -\tau \cdot v(k) \cdot \sin(\phi(k)) & 0 & 0 \\ 0 & 1 & \tau \cdot v(k) \cdot \cos(\phi(k)) & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$$\mathbf{J}_{\mathbf{u}}(k) = \frac{\partial \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k))}{\partial \mathbf{u}} = \tau \cdot \begin{bmatrix} \cos(\phi(k)) & 0 \\ \sin(\phi(k)) & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$\mathbf{P}_{\mathbf{u}} = \begin{bmatrix} 0.1^2 & 0 \\ 0 & (4 \cdot \frac{\pi}{180})^2 \end{bmatrix},$$

$$\mathbf{Q}_{\mathbf{u}}(k) = \mathbf{J}_{\mathbf{u}}(k) \cdot \mathbf{P}_{\mathbf{u}}(k) \cdot \mathbf{J}_{\mathbf{u}}(k)^T,$$

$$\mathbf{P}(k+1 \mid k) = \mathbf{J}(k) \cdot \mathbf{P}(k) \cdot \mathbf{J}(k)^T + \mathbf{Q}_{\mathbf{u}}(k).$$

Observation Function

We have the following observation function,

$$\begin{bmatrix} r \end{bmatrix} = \mathbf{h}(\mathbf{x}) = \begin{bmatrix} \sqrt{(x_k - x_s)^2 + (y_k - y_s)^2} \end{bmatrix},$$

$$\begin{bmatrix} x_s \\ y_s \\ \phi_s \\ L_{xs} \\ L_{ys} \end{bmatrix} = \mathbf{x}_s = \begin{bmatrix} L_x \cos(\alpha) - L_y \sin(\alpha) + x \\ L_x \sin(\alpha) + L_y \cos(\alpha) + y \\ \phi \\ L_x \\ L_y \end{bmatrix},$$

where (x_k, y_k) are the coordinates of observed landmarks, $\alpha = \phi - \frac{\pi}{2}$ and (x_s, y_s) is the position of the LiDAR in GCF.

Proposed H Matrix

The H matrix can be found by using the derivative chain rule,

$$H = \frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}_s} \frac{\partial \mathbf{x}_s}{\partial \mathbf{x}},$$

where we have,

$$\frac{\partial \mathbf{h}(\mathbf{x})}{\partial \mathbf{x}_s} = \begin{bmatrix} \frac{-(x_k - x_s)}{\sqrt{(x_k - x_s)^2 + (y_k - y_s)^2}} & \frac{-(y_k - y_s)}{\sqrt{(x_k - x_s)^2 + (y_k - y_s)^2}} & 0 & 0 & 0 \end{bmatrix},$$

$$\frac{\partial \mathbf{x}_s}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & -L_x \sin(\alpha) - L_y \cos(\alpha) & \cos(\alpha) & -\sin(\alpha) \\ 0 & 1 & L_x \cos(\alpha) - L_y \sin(\alpha) & \sin(\alpha) & \cos(\alpha) \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Therfore,

$$H = \begin{bmatrix} a & b & ac + bd & a\cos(\alpha) + b\sin(\alpha) & -a\sin(\alpha) + b\cos(\alpha) \end{bmatrix},$$

where,

$$\begin{aligned}
a &= \frac{-(x_k - x_s)}{\sqrt{(x_k - x_s)^2 + (y_k - y_s)^2}}, \\
b &= \frac{-(y_k - y_s)}{\sqrt{(x_k - x_s)^2 + (y_k - y_s)^2}}, \\
c &= -L_x \sin(\alpha) - L_y \cos(\alpha), \\
d &= L_x \cos(\alpha) - L_y \sin(\alpha).
\end{aligned}$$

Proposed R Matrix

We are given the pollution of the LiDAR measurements, such that,

$$R = \begin{bmatrix} 0.25^2 \end{bmatrix}.$$

Part B1

Finding the bias involves solving a cost function with one value, that is, a one dimensional search space, so the implemented approach uses a binary search. The function `optimize`, iteratively updates the bias from an initial guess of zero using a binary search. The function simulates the pose of the platform with the current estimated bias, and at each subsample event, that is, a LiDAR event, it records the ground truth heading and the estimated heading. If the average estimated heading versus the average ground truth heading is less than the threshold, which is set to 0.1 degrees, then the iterations cease as an accurate approximation of the bias has been found. Otherwise we use the cost function to determine the next estimated bias. The cost function is,

$$C = \left| \frac{\sum h_{est}}{N} - \frac{\sum h_{gt}}{N} \right|$$

where we minimise the distance between the estimated heading h_{est} and the ground truth heading h_{gt}. Since the search space is one dimensional and being implemented with a binary search the cost function returns the direction in which the search for the bias should continue so that the cost is minimised, that is, which direction the estimated heading should spin so that the cost is minimised.

In comparison with the manual approach used in Project 1, this is not only far more accurate, although both are within the required threshold, but also finds the correct bias in under a second, as opposed to having to watch the simulation play out multiple times as was done when manually tuning the bias. This is still the case when tested on other datasets with artificial biases. While it is certainly possible to find the correct value for the bias while tuning manually, this may not be that case for two or more variables. This is when the powerful tool of optimisation can become really useful, as it can search greater search spaces in a much faster time, and it is repeatable for different datasets, unlike manual tuning.