AA274 (Winter 2017-18): Problem Set 3

Anqi Fu

February 27, 2018

1 Extended Kalman Filter (EKF)

(i) See submitted code. Let the state variables be $\mathbf{x}(t) = (x(t), y(t), \theta(t))$ and the control variables be $\mathbf{u}(t) = (V(t), \omega(t))$. The differential drive model is

$$\begin{aligned}\dot{x}(t) &= V(t)\cos(\theta(t))\\ \dot{y}(t) &= V(t)\sin(\theta(t))\\ \dot{\theta}(t) &= \omega(t),\end{aligned}$$

which we can discretize by assuming a zero-order hold on the control, i.e. $\mathbf{u}(t)$ is constant over a time interval of length dt. Then we can integrate to get

$$\theta_t - \theta_{t-dt} = \int_{t-dt}^{t} \dot{\theta}(s)ds = \int_{t-dt}^{t} \omega(s)ds = \omega_t \int_{t-dt}^{t} ds = \omega_t dt.$$

If $\omega(t) = 0$ over the interval, $\theta(t)$ remains constant, so our coordinates can be found easily by integrating

$$\begin{aligned}x_t - x_{t-dt} &= \int_{t-dt}^{t} \dot{x}(s)ds = \int_{t-dt}^{t} V(s)\cos(\theta(s))ds = V_t\cos(\theta_{t-dt})dt\\ y_t - y_{t-dt} &= \int_{t-dt}^{t} \dot{y}(s)ds = \int_{t-dt}^{t} V(s)\sin(\theta(s))ds = V_t\sin(\theta_{t-dt})dt.\end{aligned}$$

With a slight abuse of notation, We may thus write

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} x_{t-1} + V_t\cos(\theta_{t-1})dt \\ y_{t-1} + V_t\sin(\theta_{t-1})dt \\ \theta_{t-1} + \omega_t dt \end{pmatrix} = g(\mathbf{x}_{t-1}, \mathbf{u}_t)$$

and the Jacobians of g with respect to $\mathbf{x}$ and $\mathbf{u}$ are, respectively,

$$G_x = \begin{pmatrix} 1 & 0 & -V_t\sin(\theta_{t-1})dt \\ 0 & 1 & V_t\cos(\theta_{t-1})dt \\ 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad G_u = \begin{pmatrix} \cos(\theta_{t-1})dt & 0 \\ \sin(\theta_{t-1})dt & 0 \\ 0 & dt \end{pmatrix}.$$

If $\omega(t) \neq 0$, we exploit the fact that $\dot{\theta}(t) = \omega(t)$ to get

$$\begin{aligned} x_t - x_{t-dt} &= \int_{t-dt}^{t} V(s)\cos(\theta(s))\frac{\dot{\theta}(s)}{\omega(s)}ds = \frac{V_t}{\omega_t}\int_{t-dt}^{t}\cos(\theta(s))\dot{\theta}(s)ds \\ &= \frac{V_t}{\omega_t}(\sin(\theta_t) - \sin(\theta_{t-dt})) = \frac{V_t}{\omega_t}(\sin(\theta_{t-dt} + \omega_t dt) - \sin(\theta_{t-dt})) \\ y_t - y_{t-dt} &= \frac{V_t}{\omega_t}\int_{t-dt}^{t}\sin(\theta(s))\dot{\theta}(s)ds = \frac{V_t}{\omega_t}(-\cos(\theta_{t-dt} + \omega_t dt) + \cos(\theta_{t-dt})) \end{aligned}$$

Our discrete model becomes

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} x_{t-1} + \frac{V_t}{\omega_t}(\sin(\theta_{t-1} + \omega_t dt) - \sin(\theta_{t-1})) \\ y_{t-1} - \frac{V_t}{\omega_t}(\cos(\theta_{t-1} + \omega_t dt) - \cos(\theta_{t-1})) \\ \theta_{t-1} + \omega_t dt \end{pmatrix} = g(\mathbf{x}_{t-1}, \mathbf{u}_t)$$

with Jacobians

$$\begin{aligned} G_x &= \begin{pmatrix} 1 & 0 & \frac{V_t}{\omega_t}(\cos(\theta_{t-1} + \omega_t dt) - \cos(\theta_{t-1})) \\ 0 & 1 & \frac{V_t}{\omega_t}(\sin(\theta_{t-1} + \omega_t dt) - \sin(\theta_{t-1})) \\ 0 & 0 & 1 \end{pmatrix} \\ G_u &= \frac{1}{\omega_t}\begin{pmatrix} \sin(\theta_t) - \sin(\theta_{t-1}) & -\frac{V_t}{\omega_t}(\sin(\theta_t) - \sin(\theta_{t-1})) + V_t\cos(\theta_t)dt \\ -\cos(\theta_t) + \cos(\theta_{t-1}) & \frac{V_t}{\omega_t}(\cos(\theta_t) - \cos(\theta_{t-1})) + V_t\sin(\theta_t)dt \\ 0 & \omega_t dt \end{pmatrix}. \end{aligned}$$

where $\theta_t := \theta_{t-1} + \omega_t dt$.

(ii) See submitted code. Let the belief state at $t-1$ be distributed as $\mathcal{N}(\mathbf{x}_{t-1}, P_{t-1})$. We model the dynamic certainty with white noise $\nu \sim \mathcal{N}(\mathbf{0}, Q)$ applied to the control input. For a time step of dt, the EKF prediction step is

$$\begin{aligned} \bar{\mathbf{x}}_t &= g(\mathbf{x}_{t-1}, \mathbf{u}_t) \\ \bar{P}_t &= G_x P_{t-1} G_x^T + dt \cdot G_u Q G_u^T. \end{aligned}$$

(iii) See submitted code. Let $\mathbf{m} = (\alpha, r)$ be the polar coordinate parameters of a line in the world frame. We are given the robot's state $\mathbf{x}_{rob} = (x_{rob}, y_{rob}, \theta_{rob})$, which is the offset/yaw of the robot's base frame with respect to the world frame, and $\mathbf{x}_{cam} = (x_{cam}, y_{cam}, \theta_{cam})$, the offset/yaw of the camera's frame with respect to the robot's base frame. To transform between frames, we must first translate by the (x_0, y_0) coordinates of the new origin, then rotate by the relative angle θ between the frames.

Define the (counterclockwise) rotation matrix and translation function

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \quad \text{and} \quad T(\mathbf{p}; \mathbf{p}_0) = \begin{pmatrix} x + x_0 \\ y + y_0 \end{pmatrix}.$$

where $\mathbf{p} = (x, y)$ and $\mathbf{p}_0 = (x_0, y_0)$. Given a point in space, the relationship between its (x, y)-coordinates in the old frame and $(\tilde{x}, \tilde{y})$-coordinates in the new frame is

$$\begin{aligned}
\begin{pmatrix} x \\ y \end{pmatrix} &= T(R(\theta)\tilde{\mathbf{p}}; \mathbf{p}_0) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \\
&= \begin{pmatrix} \tilde{x}\cos(\theta) - \tilde{y}\sin(\theta) \\ \tilde{x}\sin(\theta) + \tilde{y}\cos(\theta) \end{pmatrix} + \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} \\
\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} &= R(-\theta)T(\mathbf{p}; -\mathbf{p}_0) = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix} \\
&= \begin{pmatrix} x\cos(\theta) + y\sin(\theta) \\ -x\sin(\theta) + y\cos(\theta) \end{pmatrix} + \begin{pmatrix} x_0\cos(\theta) + y_0\sin(\theta) \\ -x_0\sin(\theta) + y_0\cos(\theta) \end{pmatrix}.
\end{aligned}$$

If our line has polar parameters (α, r) in the old frame, it can be expressed as

$$\begin{aligned}
x\cos(\alpha) + y\sin(\alpha) &= r \\
(\tilde{x}\cos(\theta) - \tilde{y}\sin(\theta) + x_0)\cos(\alpha) + (\tilde{x}\sin(\theta) + \tilde{y}\cos(\theta) + y_0)\sin(\alpha) &= r \\
\tilde{x}(\cos(\theta)\cos(\alpha) + \sin(\theta)\sin(\alpha)) + \tilde{y}(\cos(\theta)\sin(\alpha) - \sin(\theta)\cos(\alpha)) &= r - x_0\cos(\alpha) - y_0\sin(\alpha) \\
\tilde{x}\cos(\alpha - \theta) + \tilde{y}\sin(\alpha - \theta) &= r - x_0\cos(\alpha) - y_0\sin(\alpha) \\
\tilde{x}\cos(\tilde{\alpha}) + \tilde{y}\sin(\tilde{\alpha}) &= \tilde{r}
\end{aligned}$$

where the parameters in the new frame are

$$\tilde{\alpha} = \alpha - \theta, \quad \tilde{r} = r - x_0\cos(\alpha) - y_0\sin(\alpha)$$

Thus, to determine a map entry's coordinates in the camera frame, we first convert from the world frame to the robot's base frame,

$$\begin{aligned}
\alpha_{rob} &= \alpha - \theta_{rob} \\
r_{rob} &= r - x_{rob}\cos(\alpha) - y_{rob}\sin(\alpha),
\end{aligned}$$

then convert from the robot's base frame to the camera frame,

$$\begin{aligned}
\alpha_{cam} &= \alpha_{rob} - \theta_{cam} = \alpha - \theta_{rob} - \theta_{cam} \\
r_{cam} &= r_{rob} - x_{cam}\cos(\alpha_{rob}) - y_{cam}\sin(\alpha_{rob}) \\
&= r - x_{rob}\cos(\alpha) - y_{rob}\sin(\alpha) - x_{cam}\cos(\alpha - \theta_{rob}) - y_{cam}\sin(\alpha - \theta_{rob}).
\end{aligned}$$

The mean camera frame parameters are

$$\mathbf{h}_t = \begin{pmatrix} \alpha - \theta - \theta_{cam} \\ r - x\cos(\alpha) - y\sin(\alpha) - x_{cam}\cos(\alpha - \theta) - y_{cam}\sin(\alpha - \theta) \end{pmatrix}$$

and its Jacobian with respect to the belief state mean $\bar{\mathbf{x}}_t = (x, y, \theta)$ is

$$H_t = \begin{pmatrix} 0 & 0 & -1 \\ -\cos(\alpha) & -\sin(\alpha) & -x_{cam}\sin(\alpha - \theta) + y_{cam}\cos(\alpha - \theta) \end{pmatrix}$$

(iv) See submitted code. Let $\mathbf{h}_t^j$ be a predicted measurement and $\mathbf{z}_t^i$ be an observation. Define the innovation to be $\mathbf{v}_t^{ij} = \mathbf{z}_t^i - \mathbf{h}_t^j$ with covariance $S_t^{ij} = H_t^j \bar{P}_t H_t^{j^T} + R_t^i$, where R_t^i is the covariance of $\mathbf{z}_t^i$. The Mahalanobis distance is

$$d_t^{ij} = \mathbf{v}_t^{ij^T} (S_t^{ij})^{-1} \mathbf{v}_t^{ij}.$$

We associate each observation with the map entry with the least Mahalanobis distance. To avoid corrupted measurements that do not correspond to any map entries, we introduce a validation gate g and consider only associations where $d_t^{ij} < g^2$.

(v) See submitted code. We stack the innovation vectors, covariance matrices, and Jacobians with respect to $\mathbf{x}$ into

$$\mathbf{z}_t = \begin{pmatrix} \mathbf{v}_t^1 \\ \vdots \\ \mathbf{v}_t^K \end{pmatrix}, \quad R_t = \begin{pmatrix} R_t^1 & & 0 \\ & \ddots & \\ 0 & & R_t^K \end{pmatrix}, \quad H_t = \begin{pmatrix} H_t^1 \\ \vdots \\ H_t^K \end{pmatrix}.$$

(vi) See submitted code. The standard EKF measurement correction is

$$\begin{aligned} \Sigma_t &= H_t \bar{P}_t H_t^T + R_t \\ K_t &= \bar{P}_t H_t^T \Sigma_t^{-1} \\ x_t &= \bar{x}_t + K_t \mathbf{z}_t \\ P_t &= \bar{P}_t - K_t \Sigma_t K_t^T \end{aligned}$$

(vii) See submitted code. I inject extra uncertainty into the system by adding Gaussian noise to the scanner measurements. Specifically, the range data is perturbed by a standard normal error term subject to the minimum/maximum bounds, i.e. given a range observation ρ_i, the scanner returns

$$\tilde{\rho}_i = \max(\min(\rho_i + \epsilon_i, U), L) \quad \text{for } \epsilon_i \sim \mathcal{N}(0,1)$$

where $L =$ `range_min` and $U =$ `range_max` of the particular scan update. The variance of the perturbed term is approximately

$$Var(\tilde{\rho}_i) \approx Var(\rho_i) + Var(\epsilon_i) = Var(\rho_i) + 1,$$

where for mathematical simplicity, I have ignored the effect of the bounds. In this simulation, `var_rho` $= 0.05 + 1 = 1.05$.

2 EKF Simultaneous Localization and Mapping (SLAM)

(i) See submitted code. In EKF SLAM, the belief state is augmented with

$$\mathbf{x}(t) = (x(t), y(t), \theta(t), \alpha^1, r^1, \dots, \alpha^J, r^J)$$

where the map features are assumed to be static in the world frame, so that $\dot{\alpha}^j = 0$ and $\dot{r}^j = 0$ for all $j = 1, \ldots, J$. We can readily extend the discrete model from Problem 1. If $\omega(t) = 0$, the state transition function is

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \\ \alpha_t^1 \\ r_t^1 \\ \vdots \\ \alpha_t^J \\ r_t^J \end{pmatrix} = \begin{pmatrix} x_{t-1} + V_t \cos(\theta_{t-1}) dt \\ y_{t-1} + V_t \sin(\theta_{t-1}) dt \\ \theta_{t-1} + \omega_t dt \\ \alpha_{t-1}^1 \\ r_{t-1}^1 \\ \vdots \\ \alpha_{t-1}^J \\ r_{t-1}^J \end{pmatrix} = g(\mathbf{x}_{t-1}, \mathbf{u}_t)$$

with Jacobians $G_x = \frac{\partial g(\mathbf{x}_{t-1}, \mathbf{u}_t)}{\partial \mathbf{x}}$ and $G_u = \frac{\partial g(\mathbf{x}_{t-1}, \mathbf{u}_t)}{\partial \mathbf{u}}$ given by

$$G_x = \begin{pmatrix} 1 & 0 & -V_t \sin(\theta_{t-1}) dt & 0 & 0 & \ldots & 0 & 0 \\ 0 & 1 & V_t \cos(\theta_{t-1}) dt & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \ldots & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 1 \end{pmatrix} \quad \text{and} \quad G_u = \begin{pmatrix} \cos(\theta_{t-1}) dt & 0 \\ \sin(\theta_{t-1}) dt & 0 \\ 0 & dt \\ 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Similarly, when $\omega(t) \neq 0$, we have

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ y_t \\ \theta_t \\ \alpha_t^1 \\ r_t^1 \\ \vdots \\ \alpha_t^J \\ r_t^J \end{pmatrix} = \begin{pmatrix} x_{t-1} + \frac{V_t}{\omega_t}(\sin(\theta_{t-1} + \omega_t dt) - \sin(\theta_{t-1})) \\ y_{t-1} - \frac{V_t}{\omega_t}(\cos(\theta_{t-1} + \omega_t dt) - \cos(\theta_{t-1})) \\ \theta_{t-1} + \omega_t dt \\ \alpha_{t-1}^1 \\ r_{t-1}^1 \\ \vdots \\ \alpha_{t-1}^J \\ r_{t-1}^J \end{pmatrix} = g(\mathbf{x}_{t-1}, \mathbf{u}_t)$$

and Jacobians

$$G_x = \begin{pmatrix} 1 & 0 & \frac{V_t}{\omega_t}(\cos(\theta_{t-1}+\omega_t dt) - \cos(\theta_{t-1})) & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & \frac{V_t}{\omega_t}(\sin(\theta_{t-1}+\omega_t dt) - \sin(\theta_{t-1})) & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 1 \end{pmatrix}$$

$$G_u = \frac{1}{\omega_t} \begin{pmatrix} \sin(\theta_t) - \sin(\theta_{t-1}) & -\frac{V_t}{\omega_t}(\sin(\theta_t) - \sin(\theta_{t-1})) + V_t \cos(\theta_t) dt \\ -\cos(\theta_t) + \cos(\theta_{t-1}) & \frac{V_t}{\omega_t}(\cos(\theta_t) - \cos(\theta_{t-1})) + V_t \sin(\theta_t) dt \\ 0 & \omega_t dt \\ 0 & dt \\ 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

where $\theta_t := \theta_{t-1} + \omega_t dt$.

(ii) See submitted code. The mean camera frame parameters for $\mathbf{m}^j = (\alpha^j, r^j)$ is

$$\mathbf{h}_t^j = \begin{pmatrix} \alpha^j - \theta - \theta_{cam} \\ r^j - x\cos(\alpha^j) - y\sin(\alpha^j) - x_{cam}\cos(\alpha^j - \theta) - y_{cam}\sin(\alpha^j - \theta) \end{pmatrix}$$

and its Jacobian with respect to $\bar{\mathbf{x}}_t = (x, y, \theta, \alpha^1, r^1, \dots, \alpha^J, r^j)$ is

$$H_t^j = \begin{pmatrix} A_t^j & \mathbf{0} & \dots & B_t^j & \dots & \mathbf{0} \end{pmatrix} \in \mathbb{R}^{2\times(2J+3)}$$

where the leftmost block is

$$A_t^j = \begin{pmatrix} 0 & 0 & -1 \\ -\cos(\alpha^j) & -\sin(\alpha^j) & -x_{cam}\sin(\alpha^j - \theta) + y_{cam}\cos(\alpha^j - \theta) \end{pmatrix} \in \mathbb{R}^{2\times 3}$$

and the center block

$$B_t^j = \begin{pmatrix} 1 & 0 \\ x\sin(\alpha^j) - y\cos(\alpha^j) + x_{cam}\sin(\alpha^j - \theta) - y_{cam}\cos(\alpha^j - \theta) & 1 \end{pmatrix} \in \mathbb{R}^{2\times 2}$$

begins at column $4 + 2(j-1) = 2(j+1)$.

(iii) See submitted code.

3 Turtlebot Localization

(i) See submitted code.

(ii) See submitted code.