Invariant EKF Implementation documentation

Long Hui

May 25, 2022

Part I
Theorem

Go to Resources, and read Mobile Robotics PowerPoints on Matrix Lie Groups and Invariant EKF.

Part II
Our system

1 Overview

We define our robot vehicle equipped with an IMU, odometer encoder (optional), and Global Positioning System (GPS) sensors. The state is modeled using $\mathrm{SE}_2(3)$ such that

$$X_k = \begin{bmatrix} R_k & v_k & p_k \\ 0_{1,3} & 1 & 0 \\ 0_{1,3} & 0 & 1 \end{bmatrix} \in \mathrm{SE}_2(3)$$

we use a Right-Invariant EKF to estimate its pose (R_k, p_k) and velocity v_k in the world frame. v_k and p_k are in $\mathbb{R}^3$. R_k is the rotation matrix which is in the $SO(3)$ group. In addition, we have a

$$\text{bias} = \begin{bmatrix} b_{gyro} \\ b_{acc} \end{bmatrix} \in \mathbb{R}^6$$

as our bias terms, in Euclidean space.

Collectively, our system state is defined as, $\{X_k, \text{bias}, P_r\}$.

- X is a 5x5 matrix,
- bias is a 6x1 vector,
- P_r being the right-error covariance matrix which is a 15x15 matrix.

This is called **augmented Lie Matrix Group** $SE_2(3)$, with 6 dof vector.

Main Algorithm

1. For the prediction step, we use the discretized IMU model and use Right-Invariant Dynamics matrix model to propagate both our state and r-covariance,
2. and we have 3 update methods,
   (a) Odometer and non-holonomic constraint update (RIEKF method),
   (b) GPS measurement update (LIEKF method, we will project our right-covariance to left-covariance, perform update via Left-Invariant observation model, then project it back to right-covaraince),
   (c) Non-holonomic constraint update (partial RIEKF method).

1.1 Propagation/ Prediction

1.1.1 Prediction Overview

We have 2 steps in prediction steps. One is state propagation, and second is covariance propagation. I have designed this system to be primarily using RIGHT-Variant methods. Note that this is a design by choice. Either way (Left or right as dominate method) should be fine.

How-to: Our **state** can be propagated normally similar to Quaternion Error State EKF. Our **covariance** should be propagated in a Right-Invariant error method. See the box below.

We can put our process noise as a matrix. Process noise matrix, is defined:

$$Q_{15,15} = \text{blkdiag}\left(\sigma_{\text{gyro}}, \sigma_{\text{acc}}, 0_{3,3}, \sigma_{\text{biasgyro}}, \sigma_{\text{biasacc}}\right)$$

each σ looks like $\begin{bmatrix} n_x & 0 & 0 \\ 0 & n_y & 0 \\ 0 & 0 & n_z \end{bmatrix}$ for n stands for noise in each sensor.

A response to the Invariant errors/ group affine properties: (you can skip reading this part if you don't care about the theories, and jump to 1.1.2) We can express our deterministic nonlinear dynamics as,

$$f_{u_t}\left(\bar{X}_t\right) = \begin{bmatrix} \bar{R}_t \tilde{w}_t{}^\wedge & \bar{R}_k \tilde{a}_t + g & \bar{v}_t \\ 0_{1,3} & 1 & 0 \\ 0_{1,3} & 0 & 1 \end{bmatrix}$$

we have bias-corrected $\tilde{w}$, $\tilde{a}$ for gyrometer and accelerometer reading. This f_{u_t} is helpful to understand the properties behind the autonomous error dynamics

g refers to the gravational vector, $g = \begin{bmatrix} 0 \\ 0 \\ 9.81 \end{bmatrix}$. Our log-linear Right-Invariant Dyanmics Matrix is

$$A_t^r = \begin{bmatrix} 0_{3,3} & 0_{3,3} & 0_{3,3} & -R_t & 0_{3,3} \\ (g)_\times & 0_{3,3} & 0_{3,3} & -(v_t)_\times R_t & -R_t \\ 0_{3,3} & I & 0_{3,3} & -(p_t)_\times R_t & 0_{3,3} \\ 0_{3,3} & 0_{3,3} & 0_{3,3} & 0_{3,3} & 0_{3,3} \\ 0_{3,3} & 0_{3,3} & 0_{3,3} & 0_{3,3} & 0_{3,3} \end{bmatrix}$$

note, it's not entirely invariant since there's R_t, v_t, and p_t in the bias terms, but it's should be better than Q-ESKF. The r in A_t^r refers to RIGHT-Invariant method.

1.1.2 Our prediction algorithm

Propagation $(\Delta t, p_k, v_k, R_k, b_{gyro,k}, b_{acc,k}, acc_k, gyro_k, P_r, A_k^r, Q)$:

1. $\tilde{\omega} = gyro_k - b_{gyro,k}$
2. $\tilde{a} = acc_k - b_{acc,k}$
3. $R_{k+1} = R_k \exp(\text{skew}(\tilde{\omega} * dt))$
4. $v_{k+1} = v_k + R_k \tilde{a} \cdot \Delta t + g \cdot \Delta t$
5. $p_{k+1} = p_k + v_t \cdot \Delta t + 0.5 \left(R_k \tilde{a} \cdot \Delta t^2 + g \cdot \Delta t^2 \right)$
6. $\Phi = \exp(A_k^r \cdot \Delta t)$,
7. Make augmented adjoint map, $Ad_{\bar{X}} = \begin{bmatrix} \bar{R}_k & 0_{3,3} & 0_{3,3} & 0_{3,3} \\ \text{skew}(\bar{v}_k) * \bar{R}_k & R_k & 0_{3,3} & 0_{3,3} \\ \text{skew}(\bar{p}_k) * \bar{R}_k & 0_{3,3} & 0_{3,3} & 0_{3,3} \\ 0_{3,3} & 0_{3,3} & I_3 & 0_{3,3} \\ 0_{3,3} & 0_{3,3} & 0_{3,3} & I_3 \end{bmatrix}$
8. $Q_r = BQB^T$
9. $P_{k+1} = \Phi P_k \Phi^T + \Phi Q_r \Phi^T \Delta t$

Note that g, A_k^r and Q are provided above

There exists discrete dynamics to do an exact integration of the continuous-time system under the assumptionthat the IMU measurements are constant over Δt which we won't do those here. Because the difference is extremely small, we opt fot the easier method here.

1.2 Update/ Correction

We have **3** update methods: 1) odometer and non-holonomic constaint update; 2) GPS update; 3) Non-holonomic constraint update.

1.2.1 Odometer and non-holonomic constraint update

The velocity of the robot in the kinematics center frame is

$$v_c = \begin{bmatrix} v_{c,\text{forward}} \\ v_{c,\text{lateral}} \\ v_{c,\text{vertical}} \end{bmatrix}$$

hence we can write our measurement and pseudo-measurements as

$$y_k = \begin{bmatrix} v_{c,\text{forward}} \\ v_{c,\text{lateral}} \\ v_{c,\text{vertical}} \end{bmatrix} \approx \begin{bmatrix} v_{\text{odometer}} \\ 0 \\ 0 \end{bmatrix} + v_k \in \mathbb{R}^3$$

Odometer measurement and pseudo measurement model corresponds to the Right-Invariant observation form: $Y_k = \bar{X_k}^{-1} b + V_k$.

$$Y_k = \begin{bmatrix} y_k \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \bar{R}_k & \bar{v}_k & \bar{p}_k \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 0_{3*1} \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} v_k \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^5$$

for y_k and v_k in $\mathbb{R}^3$. Next we can solve our measurement jacobian H such that $H\xi_k^r = -\xi_k^{r\wedge} b$. $H = \begin{bmatrix} 0 & -I & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{3 \cdot 15}$.

Odometer and non-holonomic update(odometer)

1. Create and transform obervation noise matrix, $N = R \cdot \mathrm{diag}\left(v_k\right) \cdot R^T$
2. Calculate innovation covariance, $S = HP_rH^T + N$
3. Calculate optimal Kalman Gain, $K = P_rH^T/S$
4. Separate the gain K into 2 parts, $K = \begin{bmatrix} K_X \\ K_{\mathrm{bias}} \end{bmatrix}$. K_X is the first 9 rows, and K_{bias} is the last 6 rows.
5. Create a reduction matrix, $\Pi = \begin{bmatrix} I_3 & 0_{3,2} \end{bmatrix}$
6. Innovation terms, $\nu = \bar{X}Y - b$.
   (a) Recall that, $Y_k = \begin{bmatrix} v_{\mathrm{odometer}} \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \in \mathbb{R}^5$;
   (b) $b = \begin{bmatrix} 0_{3*1} \\ 1 \\ 0 \end{bmatrix} \in \mathbb{R}^5$
7. $\delta_X = K_X \cdot \Pi \cdot \nu$, and $\delta_{\mathrm{bias}} = K_{\mathrm{bias}} \cdot \Pi \cdot \nu$
8. $\xi_X = \left(\delta_X\right)^{\wedge}$
9. Update state estimate, $X^+ = \exp\left(\xi_X\right)\bar{X}$, and $\mathrm{bias}^+ = \bar{\mathrm{bias}} + \delta_{\mathrm{bias}} \in \mathbb{R}^6$
10. Update estimate covariance, $P_r^+ = \left(I_{15} - KH\right) P_r \left(I_{15} - KH\right)^T + KNK^T$

1.2.2 GPS update

GPS measurement model corresponds to the Left-Invariant observation form: $Y_k = \bar{X}_k b + V_k$.

$$\begin{bmatrix} y_k \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \bar{R}_k & \bar{v}_k & \bar{p}_k \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0_{3*1} \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} v_k \\ 0 \\ 1 \end{bmatrix}$$

for y_k and v_k in $\mathbb{R}^3$. Next, we can solve $H\xi_k^l = \xi_k^{l\wedge} b$. We obtain $H = \begin{bmatrix} 0 & 0 & I & 0 & 0 \end{bmatrix}$.

GPS correction method (gps)

1. Make the Adjoint map, $Ad_{\bar{X}} = \begin{bmatrix} \bar{R}_k & 0_{3,3} & 0_{3,3} & 0_{3,3} \\ \text{skew}\left(\bar{v}_k\right) * \bar{R}_k & R_k & 0_{3,3} & 0_{3,3} \\ \text{skew}\left(\bar{p}_k\right) * \bar{R}_k & 0_{3,3} & 0_{3,3} & 0_{3,3} \\ 0_{3,3} & 0_{3,3} & I_3 & 0_{3,3} \\ 0_{3,3} & 0_{3,3} & 0_{3,3} & I_3 \end{bmatrix}$,
2. Transform our right covariance to the left, $P_l = \mathrm{Ad}_{\bar{X}_k}^{-1} P_r \left(\mathrm{Ad}_{\bar{X}_k}^T\right)^{-1}$
3. $N = R^T \cdot \text{diag}\left(v_k\right) \cdot R$
4. $S = HP_lH^T + N$
5. $K = P_lH^T/S$
6. Separate the gain K into 2 parts, $K = \begin{bmatrix} K_X \\ K_{\text{bias}} \end{bmatrix}$. K_X is the first 9 rows, and K_{bias} is the last 6 rows.
7. $\Pi = \begin{bmatrix} I_3 & 0_{3,2} \end{bmatrix}$
8. $\nu = \bar{X}^{-1}Y - b$.
   (a) Recall that, $Y_k = \begin{bmatrix} y_k \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^5$;
   (b) $b = \begin{bmatrix} 0_{3*1} \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^5$
9. $\delta_X = K_X \cdot \Pi \cdot \nu$, and $\delta_{\text{bias}} = K_{\text{bias}} \cdot \Pi \cdot \nu$
10. $\xi_X = \left(\delta_X\right)^\wedge$
11. $X^+ = \bar{X} \exp\left(\xi_X\right)$, and $\text{bias}^+ = \bar{\text{bias}} + \delta_{\text{bias}} \in \mathbb{R}^3$
12. $P_l^+ = \left(I_{15} - KH\right) P_l \left(I_{15} - KH\right)^T + KNK^T$
13. $P_r^+ = \mathrm{Ad}_{X_k^+} P_l^+ \mathrm{Ad}_{X_k^+}^T$
    (a) Note: we should use $\mathrm{Ad}_{X_k^+}$,
    (b) The differences between $\mathrm{Ad}_{X_k^+}$ and $\mathrm{Ad}_{\bar{X}_k}$ isn't big, it doesn't make a big difference.

1.2.3 Non-holonomic constraint update

For this method, we want to use pseudo measurement $y_k = \begin{bmatrix} v_{c,\text{lateral}} \\ v_{c,\text{vertical}} \end{bmatrix} \approx \begin{bmatrix} 0 \\ 0 \end{bmatrix} + v_k \in \mathbb{R}^2$ to update our state. This **DOES NOT** response to any of the Invariant-Kalman filter observation model. This loses the autonomous errer dynamics property since H is linearized at some $\bar{R}_t^{\,T}$. We use traditional EKF method to drive the errors term, and use the wedge function to project it back onto the group.

Non-holonomic update()

1. We create a reduction matrix, $A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$,
2. $H_{\text{full}} = \begin{bmatrix} 0 & R^T & 0 & 0 & 0 \end{bmatrix}$,
3. $H = A \cdot H_{\mathbf{full}}$
4. $N = I_2 \cdot v_k$
5. $S = HP_rH^T + N$
6. $K = P_rH^T/S$
7. Vehicle velocity at the robotic local frame, $v_c = R^T v = \begin{bmatrix} v_{c,\text{forward}} \\ v_{c,\text{lateral}} \\ v_{c,\text{vertical}} \end{bmatrix}$, and extract the 2nd and 3rd row as $v = \begin{bmatrix} v_{c,\text{lateral}} \\ v_{c,\text{vertical}} \end{bmatrix}$,
8. Calculate linear lateral and vertical error, $e = \begin{bmatrix} 0 \\ 0 \end{bmatrix} - v = -\begin{bmatrix} v_{c,\text{lateral}} \\ v_{c,\text{vertical}} \end{bmatrix}$
9. $\delta = K{\cdot}e$.
10. Separate δ into 2 parts, δ_X and δ_{bias}.
    (a) δ_X is the first 9 rows, and
    (b) δ_{bias} is the last 6 rows.
11. $\xi_X = (\delta_X)^\wedge$
12. $X^+ = \exp(\xi_X)\,\bar{X}$, and $\text{bias}^+ = \bar{\text{bias}} + \delta_{\text{bias}} \in \mathbb{R}^6$
13. $P_r^+ = (I_{15} - KH)\,P_r\,(I_{15} - KH)^T + KNK^T$

Part III
Appendix

- skew(u) is same as the wedge function

$$\text{skew}\left(\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}\right) = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}$$

- $\exp(\cdot)$ can be computed using MATLAB expm() command, or use an approximation. For example for approximation, $exp(A) = I_3 + A$ for any 3*3 matrix A

Part IV
Resources

1. Mobile robotics
   (a) `https://github.com/UMich-CURLY-teaching/UMich-ROB-530-public`
   (b) It provides Lie Matrix Theory in Robotics, IEKF lectures on YouTube, and provide with .pdf
2. RINS-W
   (a) `https://github.com/mbrossar/RINS-W`
   (b) It provides an example on non-holonomic constraint update method,
3. Slip-Robust InEKF
   (a) `https://github.com/XihangYU630/inekf_wheeled`
   (b) It provides example on non-holonomic update, and vehicle encoder method
4. LIEKF: IMU +GPS
   (a) `https://github.com/ghaggin/invariant-ekf`