Kalman Filtering on Lie Groups

Supervised Learning Project — Soham Sachin Purohit
Guide: Prof. Ravi Banavar

January 2022- April 2022

1 Introduction

Kalman Filtering is a popular algorithm in control theory that uses a series of measurements observed over time, including statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each time frame. The continuous time Kalman Filter algorithm is provided in Figure 1. (Ref: [2]).

Continuous-Time Kalman Filter

System model and measure model

$$\dot{x} = Ax + Bu + Gw \tag{3.20a}$$

$$z = Hx + v \tag{3.20b}$$

$$x(0) \sim (\bar{x}_0, P_0), w \sim (0, Q), v \sim (0, R)$$

Assumptions

$\{w(t)\}$ and $\{v(t)\}$ are white noise processes uncorrelated with $x(0)$ and with each other. $R > 0$.

Initialization

$$P(0) = P_0, \hat{x}(0) = \bar{x}_0$$

Error covariance update

$$\dot{P} = AP + PA^T + GQG^T - PH^T R^{-1} HP \tag{3.21}$$

Kalman gain

$$K = PH^T R^{-1} \tag{3.22}$$

Estimate update

$$\dot{\hat{x}} = A\hat{x} + Bu + K(z - H\hat{x}) \tag{3.23}$$

Figure 1: Continuous Time Kalman Filter Credits: *Optimal and Robust Estimation, Chapter 3*

The above algorithm works on data over $\mathbb{R}^n$, i.e., when your system model and measure model both have all variables which are Euclidean in nature. This project extends this idea and presents an algorithm called the Extended Kalman Filter on Lie Groups for data over a linearized version of Lie Groups. Section 2 contains all the details of this algorithm. Section 3 presents an implementation of a discrete time observer for a simple case. Section 4 presents the results of this implementation, with some inferences.

2 Intrinsic Extended Kalman Filter on Lie Groups

Let G be a Lie Group and $\mathcal{G}$ be the Lie Algebra. Let ϕ : G X M $\rightarrow$ M be a left-invariant linear action on an m-dimensional vector space M. We will look at what this means in the system and measurement equations. The system dynamics and measurement on a Lie Group is now described-

$$\dot{g} = g \cdot (\zeta + \zeta_b + n_\zeta) \quad \cdots\cdots\cdots\cdots [1]$$

$$\dot{\zeta}_b = 0 \quad \cdots\cdots\cdots\cdots [2]$$

$$y = \phi_{g^{-1}}(\gamma) + n \quad \cdots\cdots\cdots\cdots [3]$$

The equation [1] defines the system dynamics that governs the evolution of the state. Equation [2] simply states that the bias term does not change with time. The equation [3] is the measurement equation. To better understand these equations, we consider an example. In this example, G is taken to be SO(3), which is the set of rotation matrices on $\mathbb{R}^3$, and is a Lie Group. We consider the case of a rotating body in $\mathbb{R}^3$, which takes measurements of certain stars through sensors. In this context, the system and the measurement equations are:

$$\dot{R} = R \cdot (\hat{\Omega} + \hat{\Omega}_b + n_\Omega) \quad \cdots\cdots\cdots\cdots [4]$$

$$\dot{\hat{\Omega}}_b = 0 \quad \cdots\cdots\cdots\cdots [5]$$

$$y = R^T \gamma + n \quad \cdots\cdots\cdots\cdots [6]$$

Here, R $\in$ SO(3), $\hat{\Omega}$ and $\hat{\Omega}_b$ are the respective skew symmetric forms of the angular velocity vectors, and n_Ω and n denote the noise. Equation [4] comes from the rotational kinematic equations, wherein pre- multiplication by a rotation matrix rotates a vector by the corresponding angles. The bias and the noise terms are added.

Equation [6] highlights the definition of $\phi_{g^{-1}}(\gamma)$. For a general operation $\phi_g(\zeta)$, the group element g performs a left action on the vector (element of the Lie Algebra) in the bracket ζ. Left action in the context of matrices is simply the matrix multiplication operation. Using the properties of rotation matrices, we know that R^{-1}=R^T. Hence, $\phi_{g^{-1}}(\gamma)$ becomes $R^T\gamma$. This equation converts the direction vectors of the landmarks (stars) whose measurement has been taken from the spatial frame (γ) to the star frame (y).

With the system dynamics and measurement equations in place, we finally define the estimator algorithm. The term "extended Kalman Filter" is used. This just means the usage of the Kalman Filter algorithm on a linearized version of the dynamics at a specified operating point.

We first define the Ad_g and ad_ζ operators. Ad_g is a tangent level operation: G X G $\rightarrow$ G, which first performs a left operation on the tangent space at the identity element of the Lie Group followed by a right operation: v $\rightarrow$ $\mathrm{T}_g\mathrm{R}_g^{-1}(\mathrm{T}_e\mathrm{L}_g\mathrm{v})$. In the context of the above example, Ad_RB= RBR^T.

We know that there is a homeomorphism between skew symmetric matrices and rotational matrices through the exponential operation. ad_ζ is the same operation as Ad_g, except that g is expressed as exp(ζ) in the latter. I.e., v $\rightarrow$ $\mathrm{T}_{exp(\zeta)}\mathrm{R}_{exp(\zeta)^{-1}}(\mathrm{T}_e\mathrm{L}_{exp(\zeta)}\mathrm{v})$. Hence, ad_ζ: $\mathcal{G}$ X G $\rightarrow$ $\mathcal{G}$

Now that we have defined all the operators needed for the Extended Kalman Filter, we now look at the algorithm. On performing linearization at a point (details skipped), we get two linear operators A(t) and H(t), which are used in the algorithm.

$$A(t) = -ad_{\zeta(t)+\zeta_b(t)}$$

$$H(t) = -T_e\phi \circ \phi_{g^{-1}}(\gamma)$$

Finally, the estimator algorithm is governed by the following equations:

$$\dot{\tilde{g}} = \tilde{g} \cdot (\zeta + \zeta_b + K(t)(y - \tilde{y})) \quad \cdots\cdots\cdots\cdots [7]$$

$$\dot{\tilde{\zeta}}_b = 0 \quad \cdots\cdots\cdots\cdots [8]$$

$$\tilde{y} = \phi_{g^{-1}}(\gamma) \quad \cdots\cdots\cdots\cdots [9]$$

Where the Kalman Filter Gain K(t) is given by solving:

$$\dot{P} = A^T P + PA^T - PH^T \Sigma_y^{-1} HP + \Sigma_\zeta \quad \cdots\cdots\cdots\cdots [10]$$

$$K = PH^T \Sigma_y^{-1} \quad \cdots\cdots\cdots\cdots [11]$$

Here, A and H are the operators defined above, P is the process covariance matrix. The Σ are the covariance of the noise corresponding to the system and the measurement based on the subscript. This is the complete continuous-time Kalman Filter algorithm for Lie Groups.

3 2D Robot in a Plane: An Observer on a Lie Group

In this section, we consider the implementation of an observer on smooth manifolds. The observer design is based on the reference [1].

First, we must understand the difference between an observer and a filter: An observer obtains the state of the dynamic system from measurements of outputs, whereas a filter obtains estimates of noise-free system outputs at the present time. In this case, we assume that there is no noise present, and we are simply taking a series of measurements of our system and looking at a convergence of state.

We consider the case of a two dimensional ground robot fixed at a point. Our aim is to estimate the angular position of the robot using the Kalman Filter Algorithm. Our input is angular velocity provided to the robot, and the robot has sensors to detect the relative location of two fixed landmarks.

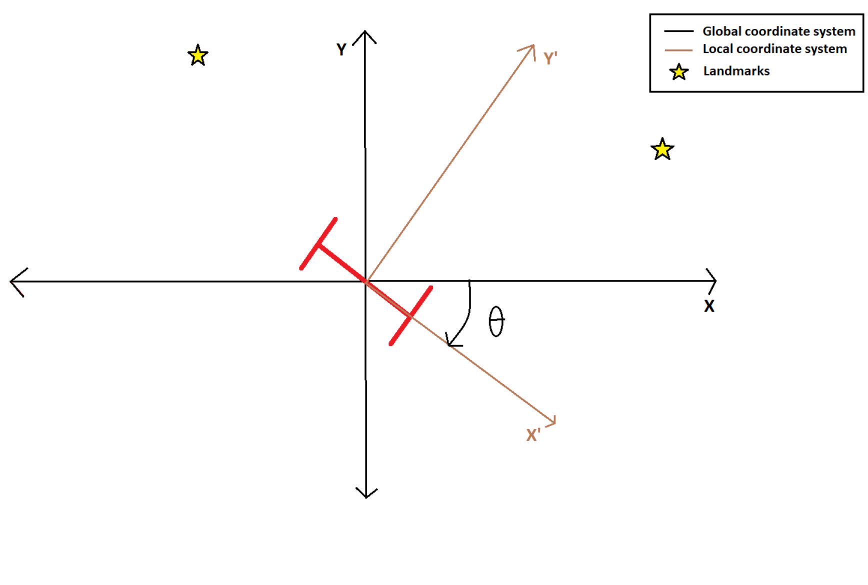


Figure 2: Rotating Robot fixed at a point. The fixed landmarks are indicated by stars

The system can be modeled on the 2-D rotation matrices, i.e., the SO(2) group. For this case, we assume that there is no noise present. The system dynamics are modeled as follows:

$$\dot{R} = R\hat{\Omega} \qquad \cdots\cdots\cdots\cdots [12]$$

Where,

$$R = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$$

$$\hat{\Omega} = \begin{bmatrix} 0 & -\dot{\theta} \\ \dot{\theta} & 0 \end{bmatrix}$$

Here, R is the state of the system and the a priori estimate which we are interested in, and $\hat{\Omega}$ is the angular velocity which we have provided as the input. The measurement equation is as follows:

$$y = R^T\eta \qquad \cdots\cdots\cdots\cdots [13]$$

Here, η is a known, fixed vector that indicates the direction of the landmark in the ground frame.

The estimator equation for this case is as follows:

$$\dot{\hat{R}} = \hat{R}(\hat{\Omega} - \Delta(\hat{R}, y)) \qquad \cdots\cdots\cdots\cdots [14]$$

$$\Delta(\hat{R}, y)) = KAd_{\hat{R}^T}\Omega_{\hat{e}} \qquad \cdots\cdots\cdots\cdots [15]$$

$$\Omega_{\hat{e}} = -\eta y^T\hat{R}^T + \hat{R}y\eta^T \qquad \cdots\cdots\cdots\cdots [16]$$

[14], [15], [16] together form the complete estimator equation. This is the continuous-time form of the system and the measurement equations. In order to simulate this, we need to have the discrete time form of the equations. We consider a case for total time t, with N time steps. For this case, the discrete form of equation [12] is:

$$R_{i+1} = R_i exp(\Omega_{\hat{i}}h) \qquad \cdots\cdots\cdots\cdots [17]$$

Next, we take measurements at each time step:

$$y_i = R_i^T\eta \qquad \cdots\cdots\cdots\cdots [18]$$

Finally, we have the estimator equation:

$$\hat{R}_{i+1} = \hat{R}_i exp((\Omega_{\hat{i}} - \Delta(\hat{R}_i, y_i))h) \qquad \cdots\cdots\cdots\cdots [19]$$

$$\Delta(\hat{R}_i, y_i)) = KAd_{\hat{R}_i^T}\Omega_{\hat{e}_i} \qquad \cdots\cdots\cdots\cdots [20]$$

$$\Omega_{\hat{e}i} = -\eta y_i^T\hat{R}_i^T + \hat{R}_i y_i\eta^T \qquad \cdots\cdots\cdots\cdots [21]$$

We now have the complete set of equations for simulating the example. This was done using Python and the code is available *here*. A code walkthrough is presented on the following page.

3.1 Code Walkthrough

In this section, an explanation of the attached code is provided. The algorithm is as follows-

- User input for the ending time, number of steps, input (Ω_i), directions of landmarks (η_i), R_0 and $\hat{R}_0$ (initial values).
- Calculate all R_i from Equation 17, use the Rodrigues Formula for the matrix exponent of skew symmetric matrices.
- Calculate all the measurements ($(\eta_1)_i$ and $(\eta_2)_i$) from Equation 18.
- Calculate all $\hat{R}_i$ using equations 19, 20, and 21.
- Calculate and plot the trace of $\hat{R}_i^T R_i$, which is a measure of error (should converge to 2).

This algorithm is summarised through the flowchart in Figure 3.

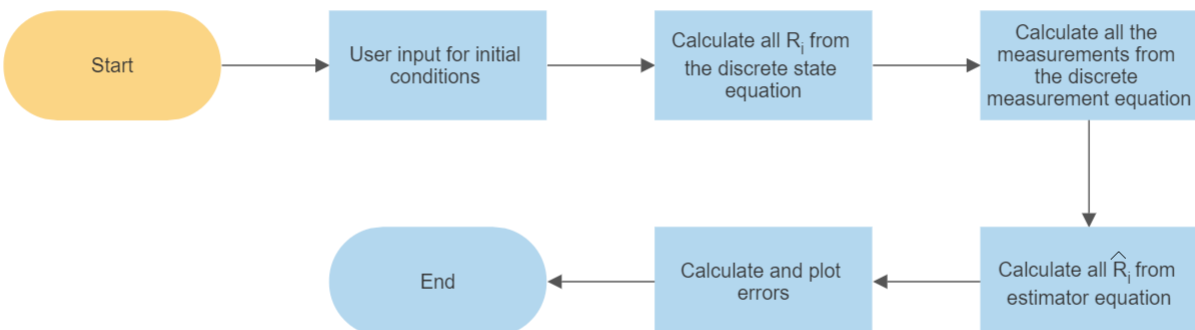
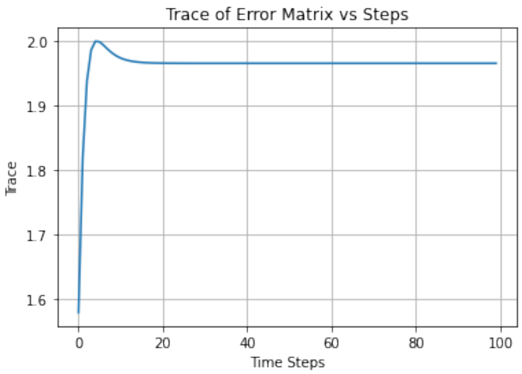


Figure 3: Implemented Observer Algorithm

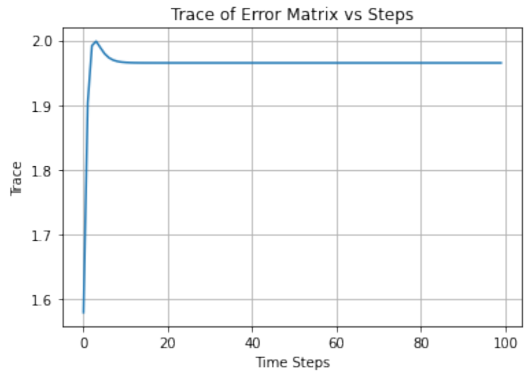
4 Results

We study the effect of change in the Kalman Gain K on convergence. The parameters taken are as follows:

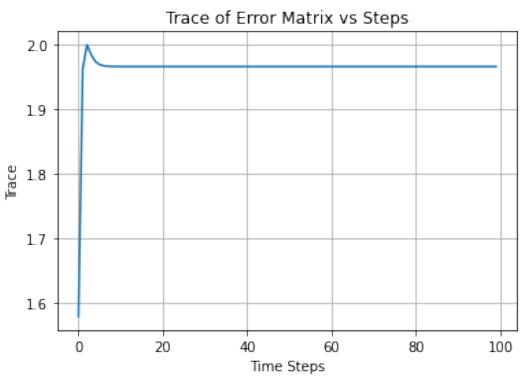
- end time=10.0
- steps=100.0
- eta_1= $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$
- eta_2=$(\frac{-2}{\sqrt{13}}, \frac{3}{\sqrt{13}})$
- R_0 = $\begin{bmatrix} cos(0.86) & -sin(0.86) \\ sin(0.86) & cos(0.86) \end{bmatrix}$
- Rhat_0 = $\begin{bmatrix} cos(0.2) & -sin(0.2) \\ sin(0.2) & cos(0.2) \end{bmatrix}$



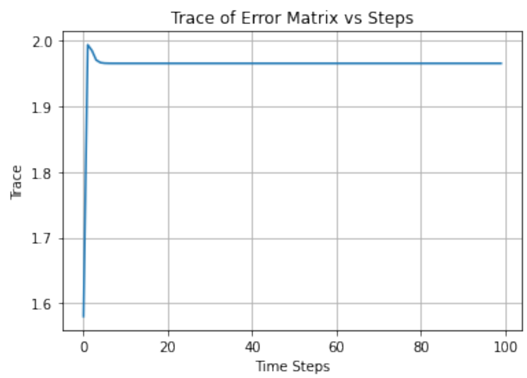
(a) K=2



(b) K=3

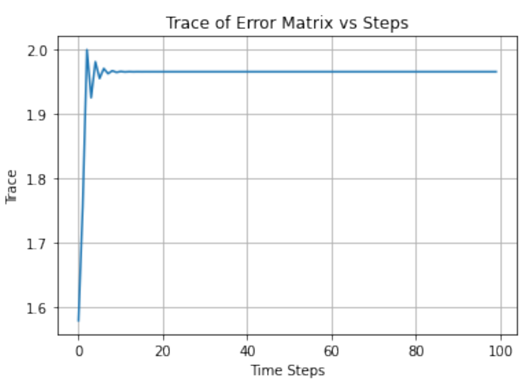


(c) K=4

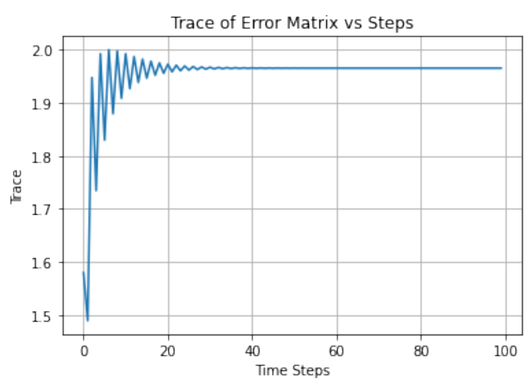


(d) K=5

Figure 4: Trace vs Time plots for various K (end time= 10, steps=100)

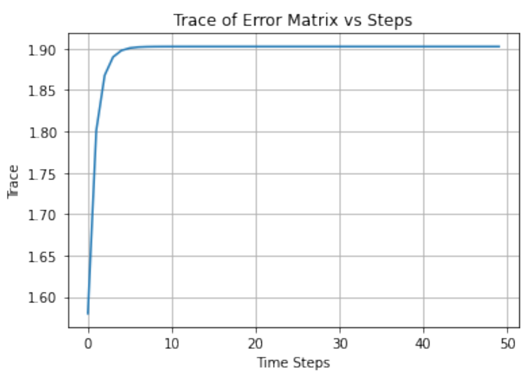


(a) K=7

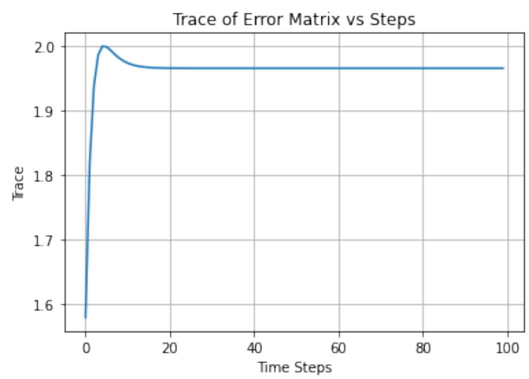


(b) K=10

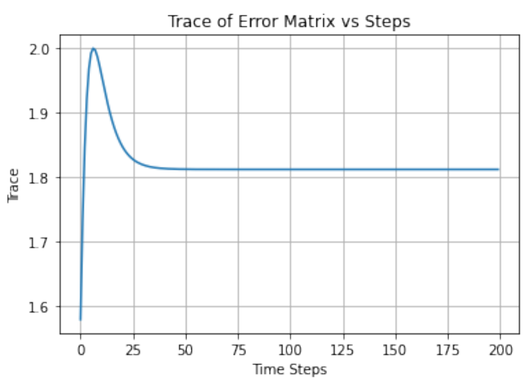
Figure 5: Trace vs Time plots for K beyond working range



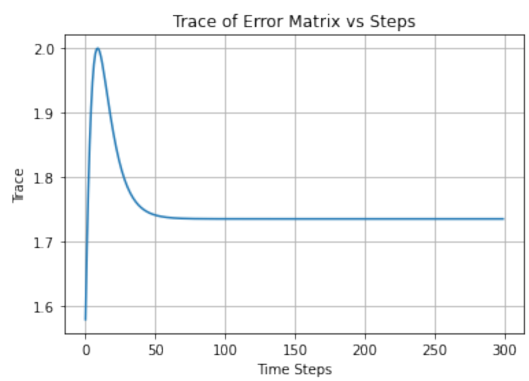
(a) steps=50



(b) steps=100



(c) steps=200



(d) steps=300

Figure 6: Trace vs Time plots for various time steps (end time= 10)

Observations

- Increase in K decreases the rise time (time to first reach 2). The value 2 here is desirable because we want our error matrix to be as close to I as possible, which implies the trace should be close to 2. Figure 3 displays the trace vs time plots for different K and fixed parameters mentioned above.
- For every set of initial conditions, there is only a certain range of values for K for which the traces are seen to smoothly converge to 2. For values beyond this range, the trace is seen to oscillate rapidly before converging. This can be seen in Figure 5.
- Next, the number of time steps were varied keeping the other initial conditions the same. Initially, an improved performance was observed, with the graph converging to values closer to 2. However, beyond a value, this performance began to drop, and the curve began to converge at values further and further away from 2. The results are tabulated in Figure 6.
- Finally, the other initial conditions were varied one at a time. The variation of curves did not show significant trends, and no correlation could be drawn from them. This proved that the most significant effect on the observer algorithm was of the time step values and the filter gain.

References

[1] Anant A. Joshi, D. H. S. Maithripala, and Ravi N. Banavar. A bundle framework for observer design on smooth manifolds with symmetry. *Journal of Geometric Mechanics*, 13(2):247, 2021.

[2] F.L. Lewis, L. Xie, and D. Popa. Optimal and robust estimation: With an introduction to stochastic control theory (2nd ed.), 2008.