Nonlinear Filters for Attitude Estimation

Chi Zhang

Abstract

This short note presents various nonlinear filtering algorithms for the problem of attitude estimation on the Lie group $SO(3)$. The purpose is to provide a technical background for the C++ codes on Github. The attitude filters presented in this note includes the multiplicative EKF, the invariant EKF, the unscented Kalman filter, the invariant ensemble Kalman filter, the bootstrap particle filter and the recently proposed feedback particle filter.

I. Problem Statement

The following attitude estimation problem is considered,

$$\mathrm{d}q_t = \frac{1}{2} q_t \otimes \big(\omega_t \,\mathrm{d}t + \sigma_B \,\mathrm{d}B_t\big), \tag{1a}$$

$$\mathrm{d}Z_t = R(q_t)^T r \,\mathrm{d}t + \sigma_W \,\mathrm{d}W_t, \tag{1b}$$

where relevant notations are as below:

- q_t – quaternion attitude at time t,
- Z_t – sensor measurement at time t,
- $\omega_t \in \mathbb{R}^3$ – angular velocity vector,
- $R(q)$ – attitude represented by rotation matrix,
- $r \in \mathbb{R}^3$ – a generic reference vector in the inertial frame,
- B_t and W_t – mutually independent Gaussian white noise,
- σ_B and σ_W – positive scalar noise parameters.

Such a simplified observation model avoids undue notations in the description of the filters. A more realistic model with both gravity and magnetic field observed is used in the actual code that produces the simulation results.

II. Overview of Attitude Filters

Many of the filters presented in this section are discrete-time filters. They require a discrete-time filtering model that is chosen to be consistent with the continuous-time model (1a)-(1b). For the discrete-time filters, the sampled observations, denoted as $\{Y_n\}$, are made at discrete times $\{t_n\}$, whose model is formally expressed as $Y_n := \frac{\Delta Z_{t_n}}{\Delta t} = h(q_{t_n}) + W_n^\Delta$ where $\{W_n^\Delta\}$ are i.i.d. with the distribution $\mathcal{N}(0, \frac{\sigma_W^2}{\Delta t} I)$. Such a model leads to the correct scaling between the continuous and the discrete-time filter implementations.

The multiplicative EKF, unscented quaternion estimator and bootstrap particle filter are originally developed for aerospace applications, and they use one of the several three-dimensional parameters to represent the attitude estimation error. The filter then reduces to a classical filter defined in the parameter space which is a subset of $\mathbb{R}^3$. In our numerical studies, the modified Rodrigues parameter (MRP) is used. Conversion between an error MRP x and an error quaternion δq is given by [8],

$$\delta q(x) = \frac{1}{16+|x|^2}\begin{bmatrix} 16-|x|^2 \\ 8x \end{bmatrix}, \tag{2}$$

$$x(\delta q) = \frac{4}{1+\delta q_0}\begin{bmatrix} \delta q_1 \\ \delta q_2 \\ \delta q_3 \end{bmatrix}, \tag{3}$$

where notationally $\delta q = (\delta q_0, \delta q_1, \delta q_2, \delta q_3)$.

All the filters are implemented using the quaternion coordinates. The posterior filter estimate at time t_n is denoted as $\hat{q}_n$. The filter estimate after the propagation step between t_{n-1} and t_n in discrete-time filters is denoted as $\hat{q}_n'$. For Kalman-type filters, the associated covariance matrices are denoted as Σ_n and Σ_n', respectively. Each filter is described for one iteration that maps q_{n-1} to q_n and Σ_{n-1} to Σ_n.

A. Multiplicative EKF

The MEKF algorithm is described in [8], [9]. The linearized model of the estimation error, represented using the MRP, is given by,

$$\mathrm{d}x_t = -[\,\omega_t\,]_\times x_t \,\mathrm{d}t - \sigma_B \,\mathrm{d}B_t, \tag{4}$$

where $x_t \in \mathbb{R}^3$ denotes the error MRP at time t. Such linearization is reasonable when $|x_t|$ and σ_B are both sufficiently small. The MEKF then follows the classical EKF based on the linearized model (4) defined in $\mathbb{R}^3$. The detailed algorithm is presented below.

Input: Current quaternion estimate $\hat{q}_{n-1}$ and Σ_{n-1}. The estimate of error MRP is $\hat{x}_{n-1} = 0$.

Propagation:

$$\hat{q}_n' = \hat{q}_{n-1} \otimes \exp(\omega_{n-1}\Delta t), \tag{5}$$

$$\Sigma_n' = \Phi \Sigma_{n-1} \Phi^T + Q, \tag{6}$$

where by slight abuse of notation,

$$\exp(\omega) := \begin{bmatrix} \cos\left(|\omega|/2\right) \\ \frac{\omega}{|\omega|}\sin\left(|\omega|/2\right) \end{bmatrix}$$

for $\omega \in \mathbb{R}^3$. The matrix Φ and Q are given by

$$\Phi = I - [\,\omega_{n-1}\,]_\times \Delta t \;, \quad Q = (\sigma_B^2 \Delta t) I.$$

Update: The observation update is first carried out for the error MRP:

$$\hat{x}_n = \mathsf{K}_n \mathrm{I}_n,$$
$$\mathsf{K}_n = \Sigma_n' H_n^T S_n^{-1},$$
$$S_n = H_n \Sigma_n' H_n^T + R,$$
$$\widetilde{\Sigma}_n = (I - \mathsf{K}_n H_n)\Sigma_n',$$

where $I_n = Y_n - R(\hat{q}_n')^T r$, $H_n = [R(\hat{q}_n')^T r]_\times$, and $R = (\sigma_W^2/\Delta t) I$.

Reset: The update of error MRP is then "transmitted" to the quaternion estimate in a multiplicative way:

$$\hat{q}_n = \hat{q}_n' \otimes \delta q(\hat{x}_n), \tag{7}$$
$$\Sigma_n = G_n \widetilde{\Sigma}_n G_n^T, \tag{8}$$

where $G_n = I - \frac{1}{2}[\mathsf{K}_n \mathrm{I}_n]_\times$, and $\delta q(\hat{x}_n)$ is calculated by (2).

The reset step is needed such that $\hat{x}_n = 0$. This is analogous to the "re-linearization" step in a classical EKF [7]. While the reset step is implicit and trivial in a Euclidean EKF, it explicitly modifies the updated covariance (see (8)) in a Lie group setting due to the multiplicative nature of the update formula (7).

Remark 1: The formula (6) for covariance propagation is obtained by discretizing the continuous-time error model (4). $O((\Delta t)^2)$ terms are neglected in both Φ and Q. For higher-order terms in these matrices, c.f., [9]. For the sake of consistency, all the filters in this section maintain the same order of numerical approximation.

B. Invariant EKF

The invariant EKF (IEKF) is a discrete-time filter proposed in [**?**]. The IEKF is originally developed to solve a slightly different attitude estimation problem,

$$\mathrm{d}q_t = \frac{1}{2} q_t \otimes \omega_t \,\mathrm{d}t + \frac{1}{2}(\sigma_B \,\mathrm{d}B_t) \otimes q_t, \tag{9a}$$
$$\mathrm{d}Z_t = R(q_t)^T (r + \sigma_W \,\mathrm{d}W_t), \tag{9b}$$

where both the process and observation noise are defined in the inertial frame instead of the body frame. However, the noise defined in two frames are statistically equivalent if the noise is *isotropic* (see [**?**]). The noise considered in the filtering problem (1a)-(1b) is isotropic and rotation-invariant since the components of the noise are independent and have equal variance.

The detailed algorithm is presented below:

Input: Current quaternion estimate $\hat{q}_{n-1}$ and Σ_{n-1}. The error estimate $\hat{e}_{n-1} = 0$.

Propagation:

$$\hat{q}'_n = \hat{q}_{n-1} \otimes \exp(\omega_{n-1}\Delta t),$$
$$\Sigma'_n = \Sigma_{n-1} + \left(\sigma_B^2 \Delta t\right) I,$$

The propagation step is identical to MEKF.

Update: The observation update is carried out for the Lie-algebraic error as follows,

$$\hat{e}_n = \mathsf{K}_n \mathrm{I}_n,$$
$$\mathsf{K}_n = \Sigma'_n H^T S_n^{-1},$$
$$S_n = H \Sigma'_n H^T + (\sigma_W^2/\Delta t) I,$$

where $H = [r]_\times$, and the innovation $\mathrm{I}_n = R(\hat{q}'_n) Y_n - r$ is modeled in the inertial frame. The update in the error estimate is then "transmitted" to the quaternion estimate in a multiplicative way:

$$\hat{q}_n = \delta q(\hat{e}_n) \otimes \hat{q}'_n, \tag{10}$$
$$\Sigma_n = (I - \mathsf{K}_n H) \Sigma'_n.$$

C. Unscented Quaternion Estimator

The unscented quaternion estimator (USQUE) is described in [5]. The estimation error is parameterized using the MRP, and sigma points are employed to represent the error distribution. A conventional unscented Kalman filter is then implemented in the parameter space. The number of sigma points $L = 2d + 1$ where $d = 3$ is the dimensional of the MRP. The detailed algorithm is presented below.

Input: Current quaternion estimate $\hat{q}_{n-1}$ and Σ_{n-1}. The error estimate in MRP is $\hat{x}_{n-1} = 0$.

Sigma points generation:

i) Generate L sigma points of error MRP $\{x_{n-1}(l)\}_{l=0}^{L-1} \in \mathbb{R}^3$ from Σ_{n-1},

$$x_{n-1}(0) = \hat{x}_{n-1} = 0, \qquad \{x_{n-1}(l)\}_{l=1}^{L-1} = 2d \text{ columns of } \pm\sqrt{(d+\lambda)\Sigma_{n-1}},$$

where λ is a tuning parameter. In practice, the square root of a positive definite matrix is computed using the Cholesky decomposition. It is recommended in [5] that $\lambda = 1$.

ii) Convert error MRP $x_{n-1}(l)$ to error quaternion $\delta q_{n-1}(l)$,

$$\delta q_{n-1}(0) = q_I, \quad \delta q_{n-1}(l) = \delta q(x_{n-1}(l)) \text{ using (2) for } l = 1, \ldots, L-1.$$

iii) Generate quaternion sigma points $\{q_{n-1}(l)\}_{l=0}^{L-1}$,

$$q_{n-1}(0) = \hat{q}_{n-1}, \qquad q_{n-1}(l) = \hat{q}_{n-1} \otimes \delta q_{n-1}(l).$$

Propagation:

i) Propagate quaternion sigma points according to the process model,

$$q'_n(l) = q_{n-1}(l) \otimes \exp(\omega_{n-1}\,\Delta t), \qquad \forall\ l.$$

ii) Calculate propagated error quaternion,

$$\delta q'_n(0) = q_I, \qquad \delta q'_n(l) = \big(q'_n(0)\big)^{-1} \otimes q'_n(l).$$

iii) Calculate propagated error MRP,

$$x'_n(0) = 0, \qquad x'_n(l) = x(\delta q'_n(l)) \quad \text{(see (3))}.$$

iv) Calculate propagated mean $\hat{x}'_n$ and covariance Σ'_n from the sigma points $\{x'_n(l)\}_{l=0}^{L-1}$ using (11) and (12) below. Additionally, process noise matrix $Q = \frac{1}{2}(\sigma_B^2 \Delta t)I$ is added to Σ'_n.

In general, given a set of sigma points $\{x(l)\}_{l=0}^{L-1} \in \mathbb{R}^d$, their mean (weighted average) and covariance are calculated as,

$$\hat{x} = \frac{1}{d+\lambda}\Big(\lambda\, x(0) + \frac{1}{2}\sum_{l=1}^{L-1} x(l)\Big), \tag{11}$$

$$\Sigma = \frac{1}{d+\lambda}\Big(\lambda\,(x(0)-\hat{x})(x(0)-\hat{x})^T + \frac{1}{2}\sum_{l=1}^{L-1}(x(l)-\hat{x})(x(l)-\hat{x})^T\Big). \tag{12}$$

Update:

i) Calculate predicted observation of each sigma point,

$$y_n(l) = R(q'_n(l))^T r, \quad \forall\ l,$$

whose covariance, denoted as $\widetilde{\Sigma}_n^{yy}$, is calculated using (12).

ii) Calculate the cross-correlation matrix Σ_n^{xy} from the two sets of sigma points $\{\hat{x}'_n\}_{l=0}^{L-1}$ and $\{y_n\}_{l=0}^{L-1}$. The formula is analogous to (12).

iii) Calculate the gain matrix $\mathsf{K}_n = \Sigma_n^{xy}\Sigma_n^{yy\,-1}$, where $\Sigma_n^{yy} = \widetilde{\Sigma}_n^{yy} + R$ and $R = (\sigma_W^2/\Delta t)\,I$.

iv) Calculate the innovation error $\mathrm{I}_n = Y_n - y_n(0)$.

v) Update the error MRP estimate,

$$\hat{x}_n = \mathsf{K}_n\,\mathrm{I}_n.$$

vi) Update the covariance matrix,

$$\Sigma_n = \Sigma_{n-1} - \mathsf{K}_n\,\Sigma_n^{yy}\,\mathsf{K}_n^T.$$

vii) Update quaternion estimate,

$$\hat{q}_n = \hat{q}'_n \otimes \delta q(\hat{x}_n).$$

Reset: After the reset step, $\hat{x}_n = 0$. The covariance matrix is unchanged after reset in USQUE.

D. Bootstrap Particle Filter

The bootstrap particle filter (BPF) is based on importance sampling and resampling. The detailed algorithm is presented below:

Input: Current quaternion particles $\{q_{n-1}^i\}_{i=1}^N$

i) Propagate each particle,

$$q_n^{i'} = q_n^i \otimes \exp(\omega_{n-1}\Delta t)$$

ii) Calculate importance weights,

$$w_n^i \propto \exp\Big(-\frac{(Y_n - h(q_n^{i'}))^T(Y_n - h(q_n^{i'}))}{2\sigma_W^2/\Delta t}\Big).$$

iii) Resampling with replacement [].

iv) Diffuse the particles with a Gaussian kernel (white noise).

E. Feedback Particle Filter

The feedback particle filter (FPF) for attitude estimation is recently proposed in []. The FPF consists of a set of interacting particles $\{q_t^I\}_{i=1}^N$, and each particle q_t^I evolves according to,

$$\mathrm{d}q_t^i = \frac{1}{2}\, q_t^i \otimes \mathrm{d}v_t^i, \tag{13}$$

where $v_t^i \in \mathbb{R}^3$ evolves according to,

$$\mathrm{d}v_t^i = \omega_t\,\mathrm{d}t + \mathrm{d}B_t^i + \mathsf{K}(q_t^i,t) \circ \Big(\mathrm{d}Z_t - \frac{h(q_t^i) + \hat{h}}{2}\,\mathrm{d}t\Big), \tag{14}$$

where $\mathsf{K}(q,t) \in \mathbb{R}^{3\times m}$ is called the *gain function*, and for notational ease, $h(q) := R(q)^T r \in \mathbb{R}^3$, and h_j denotes the j-th component of h. K is obtained as follows: For $j = 1,2,3$, the j-th column of K contains the coordinates of the vector-field $\mathrm{grad}(\phi_j)$, where the function $\phi_j \in H^1(SO(3);\pi_t)$ is a solution to the Poisson equation,

$$\begin{aligned} &\pi_t\big(\langle \mathrm{grad}(\phi_j), \mathrm{grad}(\psi)\rangle\big) = \frac{1}{\sigma_W^2}\,\pi_t\big((h_j - \hat{h}_j)\psi\big), \\ &\pi_t(\phi_j) = 0 \qquad \text{(normalization)}, \end{aligned} \tag{15}$$

for all $\psi \in H^1(SO(3);\pi)$.

The Poisson equation needs to be solved at each time t. A *Galerkin* numerical procedure is presented next to obtain approximate solution of the Poisson equation. Since the procedure is repeated for each j and each t, The subscript j and t are omitted.

Galerkin Gain Function Approximation: In a Galerkin approach, the solution ϕ is approximated as,

$$\phi = \sum_{l=1}^{L} \kappa_l\,\psi_l,$$

where $\{\psi_l\}_{l=1}^L \subset H_0^1(SO(3);\pi)$ is a given (assumed) set of *basis functions* on $SO(3)$. Denote $S := \text{span}(\psi_1,...,\psi_L)$. The finite-dimensional approximation of the Poisson equation (**??**) is to choose coefficients $\{\kappa_l\}_{l=1}^L$ such that,

$$\pi\big(\langle \text{grad}(\phi), \text{grad}(\psi)\rangle\big) = \pi\big((h-\hat{h})\psi\big), \tag{16}$$

for all $\psi \in H^1(SO(3);\pi)$. On taking $\psi = \psi_1,...,\psi_L$, (16) is compactly written as a linear matrix equation,

$$A\kappa = b, \tag{17}$$

where $\kappa := (\kappa_1,\ldots,\kappa_L)$, and the entries of the $L\times L$ matrix A and the $L\times 1$ vector b are defined as,

$$\begin{aligned} A_{kl} &= \pi\big(\langle \text{grad}(\psi_l), \text{grad}(\psi_k)\rangle\big), \\ b_k &= \pi\big((h-\hat{h})\psi_k\big), \end{aligned}$$

In numerical implementations with a finite set of particles $\{X^i\}_{i=1}^N$ sampled from the distribution π, the empirical approximation of (17) is denoted as

$$A\kappa = b,$$

where the entries of A and b are given by,

$$\begin{aligned} A_{kl} &= \frac{1}{N}\sum_{i=1}^N \langle \text{grad}(\psi_l)(q^i), \text{grad}(\psi_k)(X^i)\rangle = \frac{1}{N}\sum_{i=1}^N\sum_{n=1}^d (E_n\cdot\psi_l)(q^i)\,(E_n\cdot\psi_k)(q^i), \\ b_k &= \frac{1}{N}\sum_{i=1}^N (h(q^i)-\hat{h}^{(N)})\psi_k(q^i), \end{aligned}$$

where $\hat{h}^{(N)} := \frac{1}{N}\sum_{i=1}^N h(X^i)$. The solution ϕ is then empirically approximated by the function

$$\phi := \sum_{l=1}^L \kappa_l \psi_l,$$

and the gain function K is empirically approximated by the function $\mathsf{K}^{(N)} \in so(3)$ whose coordinates with respect to a basis $\{E_n\}_{n=1}^d$ of $so(3)$ are given by

$$\mathsf{k}_n = \sum_{l=1}^L \kappa_l\, E_n\cdot\psi_l, \quad n = 1,...,d.$$

One choice of the basis functions on $SO(3)$ is given by Table I below. The basis functions and their gradients are expressed using both the rotation matrix and the quaternion.

The numerical algorithm of FPF is summarized below, presented for one iteration from time t to $t+\Delta t$:

Input: Current quaternion particles $\{q_t^i\}_{i=1}^N$

Basis functions on $SO(3)$

	expression in R	expression in q	$E_1\cdot$	$E_2\cdot$	$E_3\cdot$
ψ_1	R_{33}	$2(q_0^2+q_3^2)-1$	$2(-q_0q_1-q_2q_3)$	$2(-q_0q_2+q_1q_3)$	0
ψ_2	R_{13}	$2(q_0q_2+q_1q_3)$	$2(q_0q_3-q_1q_2)$	$2(q_0^2+q_1^2)-1$	0
ψ_3	$-R_{23}$	$2(q_0q_1-q_2q_3)$	$2(q_0^2+q_2^2)-1$	$2(-q_0q_3-q_1q_2)$	0
ψ_4	R_{31}	$2(-q_0q_2+q_1q_3)$	0	$-2(q_0^2+q_3^2)+1$	$2(q_0q_1+q_2q_3)$
ψ_5	R_{32}	$2(q_0q_1+q_2q_3)$	$2(q_0^2+q_3^2)-1$	0	$2(q_0q_2-q_1q_3)$
ψ_6	$(1/2)(R_{21}-R_{12})$	$2q_0q_3$	$-q_0q_2-q_1q_3$	$q_0q_1-q_2q_3$	$q_0^2-q_3^2$
ψ_7	$(1/2)(R_{11}+R_{22})$	$q_0^2-q_3^2$	$-q_0q_1+q_2q_3$	$-q_0q_2-q_1q_3$	$-2q_0q_3$
ψ_8	$(1/2)(R_{21}+R_{12})$	$2q_1q_2$	$q_0q_2+q_1q_3$	$q_0q_1-q_2q_3$	$q_2^2-q_1^2$
ψ_9	$(1/2)(R_{11}-R_{22})$	$q_1^2-q_2^2$	$q_0q_1-q_2q_3$	$-q_0q_2-q_1q_3$	$2q_1q_2$

i) Calculate $\hat{h}^{(N)} = \frac{1}{N}\sum_{i=1}^{N} h(q_t^i)$
ii) Generate random vector $\Delta B_t^i \sim N(0,(\Delta t)I)$
iii) Calculate innovation $\Delta I_t^i = \mathrm{d}Z_t - \frac{1}{2}(h(q_t^i)+\hat{h}^{(N)})\,\mathrm{d}t$
iv) Calculate gain function $\mathsf{K}(q_t^i,t)$ using the Galerkin method
v) Propagate each particle according to (13) and (14),

$$q_{t+\Delta t}^i = q_t^i \otimes \exp(\omega_t\Delta t + \sigma_B \Delta B_t^i + \mathsf{K}(q_t^i,t)\Delta I_t^i).$$

References

[1] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[2] A. Barrau and S. Bonnabel. Intrinsic filtering on Lie groups with applications to attitude estimation. *IEEE Trans. Autom. Control*, 60(2):436–449, 2015.

[3] S. Bonnabel, P. Martin, and E. Salaün. Invariant extended Kalman filter: theory and application to a velocity-aided attitude estimation problem. In *Proc. 48th IEEE Conf. Decision Control held jointly with the 28th Chinese Control Conf. (CDC/CCC)*, pages 1297–1304, 2009.

[4] Y. Cheng and J. L. Crassidis. Particle filtering for attitude estimation using a minimal local-error representation. *J. Guid. Control Dynam.*, 33(4):1305–1310, 2010.

[5] J. L. Crassidis and F. L. Markley. Unscented filtering for spacecraft attitude estimation. *J. Guid. Control Dynam.*, 26(4):536–542, 2003.

[6] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[7] A. H. Jazwinski. *Stochastic processes and filtering theory*. Dover Publications, 1970.

[8] F. L. Markley. Attitude error representations for Kalman filtering. *J. Guid. Control Dynam.*, 26(2):311–317, 2003.

[9] N. Trawny and S. I. Roumeliotis. Indirect Kalman filter for 3D attitude estimation. *University of Minnesota, Dept. of Comp. Sci. and Eng., Tech. Rep.*, 2, 2005.

[10] M. Zamani. *Deterministic attitude and pose filtering, an embedded Lie groups approach*. PhD thesis, Australian National University, 2013.

[11] C. Zhang, A. Taghvaei, and P. G. Mehta. Feedback particle filter on matrix Lie groups. To be submitted to IEEE Trans. Autom. Control.