Sequential Markov Chain Monte Carlo methods on Matrix Lie Groups

Enzo LOPEZ*†, Karim DAHIA*, Nicolas MERLINGE*,
Benedicte WINTER-BONNET†, Alain MASCHIELLA† and Christian MUSSO*
*DTIS, ONERA, Université Paris-Saclay, 91120, Palaiseau, France
(enzo.lopez / karim.dahia / nicolas.merlinge / christian.musso)@onera.fr
†MBDA, Plessis-Robinson, France
(enzo.lopez / benedicte.winter-bonnet / alain.maschiella)@mbda-systems.com

***Abstract*—Particle filters on Lie Groups represent a cutting-edge approach in nonlinear filtering and control. By generating randomly distributed particles from proposed densities, while accurately preserving rotation matrices, they offer a promising solution to nonlinear systems. However, particle filters grapple with numerous challenges such as high computational costs due to the large number of particles needed, vulnerability to particle degeneracy as well as the curse of dimensionality. To address these issues, Sequential Markov Chain Monte Carlo (SMCMC) methods aim to mitigate sensitivity to high-dimensional systems by iteratively sampling from the posterior density of the system state, gradually refining the estimation. In this paper we extend SMCMC techniques to matrix Lie groups, resulting in the Lie Groups Sequential Markov Chain Monte Carlo (LG-SMCMC) filter that circumvent the major drawbacks of particle filters. The proposed approach incorporates enhancements based on the Metropolis-Hastings algorithm, further improving the algorithms efficiency and robustness. To validate the effectiveness of these methods, the algorithms are tested on an Unmanned Aerial Vehicle (UAV) navigation scenario with challenging discrepancies in the noise tuning.**

I. Introduction

Nonlinear state estimation is a fundamental aspect of robotics, involving the fusion of data from various sensors to track variables of interest while mitigating sensor inaccuracies. The Particle Filter (PF) has traditionally been favored for solving nonlinear problems with non Gaussian noise due to its practical applicability. However, its performance, in particular for angular variables, is heavily influenced by particle impoverishment and sample size dependency, that may yield inconsistent estimates.

Recent studies have highlighted the superiority of particle filters operating on manifolds [1], [2] over their Euclidean counterparts, offering enhanced robustness, accuracy, and computational efficiency. In particular, Lie groups present a structured algebraic framework well-suited for addressing scenarios involving rotational motions. Various extensions of the Particle Filter tailored to Lie groups, such as Kernel Lie Group Particle Filter (KLGPF) [3] and Lie Group Laplacian filter (LGLP) [4] have emerged, setting new benchmarks in nonlinear filtering.

In the context of filtering problems, Sequential Monte Carlo Markov Chain (SMCMC) methods [5] were developed to address the major challenges of the Particle Filter. By exploring the state space more efficiently through Markov chain transitions and by sampling particles directly from the target density, these methods offer a powerful and flexible approach to filtering problems, overcoming challenges associated with high-dimensional state spaces and weight degeneracy.

This paper proposes a novel framework for deriving SMCMC [6] on unimodular matrix Lie groups, offering a broader scope for estimation problems. It introduces a novel approach encompassing techniques such as Metropolis-Hastings sampling [7], therefore providing a framework for generating samples from a target density by constructing a Markov chain within the Lie groups formalism.

The proposed approach is evaluated through simulations of a nonlinear UAV flight estimation scenario, comparing SMCMC and particle filters on Lie groups against their Euclidean counterparts. Results demonstrate improved accuracy and robustness with low computational complexity, making the method promising for real-world applications.

In what follows, Section II states the estimation problem and details the main concepts of matrix Lie groups and SMCMC. Then, Section III details the framework leading to SMCMC on Lie groups, which is the main contribution of this paper. Section IV describes a simulated navigation problem for an UAV and tests the method introduced in this paper. Eventually, Section V concludes the paper.

II. Problem Statement

A. Introduction to Lie groups

1) General definitions: A Lie group $(\mathcal{G}, \cdot)$ refers to a manifold with a group structure [8]. It is characterized by a tangent space at its identity point, known as the *Lie algebra*, represented as $\mathfrak{g}$. A local bijection can be defined between the Lie group and the Lie algebra at the vicinity of the group identity I, the *group exponential* $\exp_{\mathcal{G}} : \mathfrak{g} \to \mathcal{G}$ and the *group logarithm* $\log_{\mathcal{G}} : \mathcal{G} \to \mathfrak{g}$.
For matrix Lie groups, these mappings can be formulated using matrix series [8]:

$$\exp_{\mathcal{G}}(X) = \sum_{k=0}^{\infty} \frac{X^k}{k!} \; ; \; \log_{\mathcal{G}}(X) = \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} (X - I)^k. \tag{1}$$

Given that $\mathfrak{g}$ has a dimension d, we can introduce two isomorphisms with the Euclidean space:

$$[\cdot]^\wedge : \mathbb{R}^d \to \mathfrak{g}, \text{ and } [\cdot]^\vee : \mathfrak{g} \to \mathbb{R}^d, \tag{2}$$

with their compositions with $\exp_\mathcal{G}$ and $\log_\mathcal{G}$ denoted as:

$$\exp_\mathcal{G}([\cdot]^\wedge) = \exp^\wedge_\mathcal{G}(\cdot) \ ; \ \log_\mathcal{G}([\cdot]^\vee) = \log^\vee_\mathcal{G}(\cdot). \tag{3}$$

A summarized depiction of these mappings is provided in Figure 1. Note that the bijection between $\mathcal{G}$ and $\mathbb{R}^d$ theoretically applies at the vicinity of I_d.

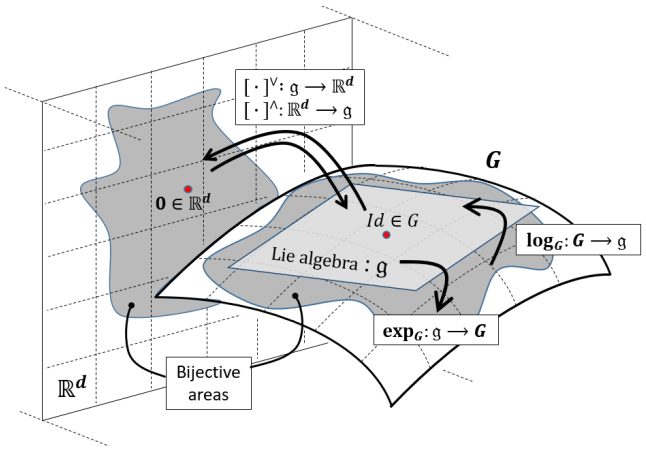


Fig. 1: Visual representation of the Lie group structure. The group exponential $\exp_\mathcal{G}$ and logarithm $\log_\mathcal{G}$ establish a local bijection between $\mathcal{G}$ and $\mathbb{R}^d$, with the algebra $\mathfrak{g}$ representing the tangent space at I [9].

2) Group errors: Consider two state matrices, denoted as $(X, \widehat{X})$, belonging to the group $(\mathcal{G}, \cdot)$. Since the group law is not necessarily commutative, as exemplified by the rotation matrices group $S0(3)$, the group error between X and $\widehat{X}$ can be expressed in two ways: on the *right* as $\eta^R = \widehat{X}X^{-1}$, and on the *left* as $\eta^L = X^{-1}\widehat{X}$.
Furthermore, the representation of the (whether left or right) group error η in Euclidean space, denoted as ϵ, can be described as follows:

$$\epsilon = \log^\vee_\mathcal{G}(\eta). \tag{4}$$

This formulation serves as a crucial aspect of Lie group filters performance [10] as it translates a matrix error into a vector error, which maintains accuracy over a broad domain, even for angular variables.

3) Probabilities on Lie groups: Let $X \sim \mathcal{N}_\mathcal{G}(\mu, P)$ represent a random matrix following a concentrated Gaussian density on $\mathcal{G}$ with mean μ and covariance matrix P. Its probability density function can be approximated by [11] [12]:

$$p(X) \approx \frac{1}{\sqrt{(2\pi)^d \det[P]}} e^{-\frac{1}{2}||\epsilon||^2_P}, \tag{5}$$

where $\epsilon = \log^\vee_\mathcal{G}(\mu^{-1}X)$ in the left case, and $\epsilon = \log^\vee_\mathcal{G}(X\mu^{-1})$ in the right case, and $||\cdot||_P$ denotes the Mahalanobis norm with respect to P. This definition holds when the density is *concentrated* around its mean, indicating that all eigenvalues of P are sufficiently small to remain within the bijective area of the group exponential [12].

B. Particle filtering on Lie groups

Let $\{X_k\}_{k\in\mathbb{N}^*} \in \mathcal{G}$ denote a discrete-time hidden state matrix X, and $\{Y_k\}_{k\in\mathbb{N}^*} \in \mathcal{H}$ represent a set of observations. A generic discrete-time nonlinear system is formulated as follows:

$$\begin{cases} X_k = f(X_{k-1}, n^q_{k-1}), \\ Y_k = h(X_k, n^r_k), \end{cases} \tag{6}$$

where (n^q_{k-1}, n^r_k) are noise vectors, $\mathcal{H}$ is the Lie group of the measurements, and (f, h) denotes a pair of nonlinear functions, the dynamics model and the observation model respectively. A filter aims to estimate the posterior density $p(X_k|Y_{1:k})$, where $p(X_0)$ is known, and $Y_{1:k} = \{Y_1 \ , \ ... \ , Y_k\}$. To address this estimation problem, the Lie Group Particle Filter (LG-PF) generates a set of samples that approximate the filtering density under the following assumptions [13]:

- $\{X_k\}_{k\in\mathbb{N}^*}$ has a Markovian evolution;
- $\{Y_k\}_{k\in\mathbb{N}^*}$ are mutually independent and identically distributed given the state;

In that case, the state follows a density on $\mathcal{G}$. The LG-PF algorithm operates in a manner akin to the Particle Filter (PF) but requires that each particle belongs to the manifold $\mathcal{G}$. Thus, the sampling of the particles has to account for the group geometry and its curved shape:

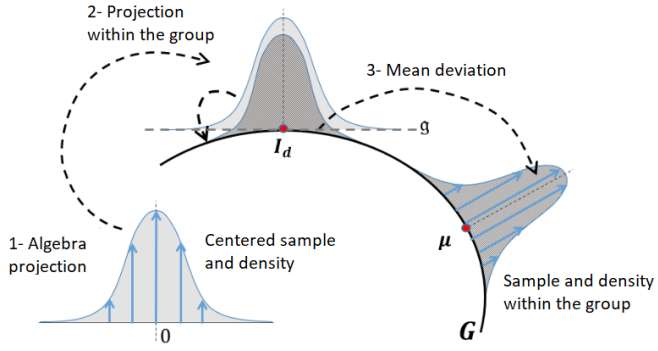


Fig. 2: Illustration of the Sampling on Lie groups [9].

Therefore, the filter initially generates a set of particles $\epsilon^{1:N_p}$ following a centered Gaussian density on the Euclidean space. This ensures that the particles are computed close to the identity:

$$\epsilon^{1:N_p} \sim \mathcal{N}_\mathcal{G}(0, P_0), \tag{7}$$

where N_p is the total number of particles. Then, each particle is projected into the group with an exponential and translated by a left or right mean μ to create a random sample that follows the Lie group geometry, as shown in Figure 2:

$$\begin{aligned} Left : X^i_0 &= \mu \exp^\wedge_\mathcal{G}(\epsilon^i), \\ Right : X^i_0 &= \exp^\wedge_\mathcal{G}(\epsilon^i)\mu. \end{aligned} \tag{8}$$

Then, it performs a propagation step, where each particle follows the dynamics of the system:

$$X_k^i = f(X_{k-1}^i, n_{k-1}^{q^i}). \tag{9}$$

Subsequently the filter performs a weighting step by measuring the likelihood associated to each particle with respect to the observations and the measurement noise density, thus linking a weight w^i to each particle:

$$w_k^i \propto w_{k-1}^i \; p(Y_k|X_k^i). \tag{10}$$

An estimation at step k is then given as a mean of all the particles and their updated weights. As shown in [14], the left case mean of the sample endowed on Lie groups becomes:

$$\mu_k \text{ such that } \sum_{i=1}^{N_p} w_k^i \log_{\mathcal{G}}^{\vee}(\mu_k^{-1} X_k^i) = 0. \tag{11}$$

There are many ways to find an approximate of the mean [15]. In this paper, we will use the fixed point iteration to compute the mean in the left case [14].

The diversity of particles plays an essential role as they need to entirely cover the posterior density function to give a general depiction of the goal density. A resampling step is therefore needed, using a resampling method such as multinomial, stratified or systematic re-sampling to avoid weight degeneracy [16]. The Particle Filter (PF) described in this paper utilizes the Sequential Importance Resampling (SIR) method and thus monitors the particle weights using the following criterion:

$$N_{\mathit{eff}} = \frac{1}{\sum_{i=1}^{N_p} \left(w_k^i\right)^2} < N_{\mathit{th}}, \tag{12}$$

where N_{th} is a parameter chosen equal to $0.75N_p$. In contrast, the Lie Group Particle Filter (LG-PF) presented here implements systematic re-sampling, wherein the resampling step occurs at each iteration.

Weight degeneracy and high dimensional states can negatively affect the filtering process by exponentially increasing the number of particles and thus the cost of computations. This is known as the curse of dimensionality and is one of the biggest issues that particle filters regularly faces. Indeed, not increasing the number of particles in high dimensional areas can degrade the estimation quality and make the filter either over-conservative or divergent.

C. Sequential Monte Carlo Markov Chain methods

SMCMC methods are based on Monte Carlo simulations and rely on the Bayesian theory to estimate the posterior density of the state X_k given the measurements $Y_{1:k}$. The Bayes rule states that:

$$p(X_k|Y_{1:k}) = \frac{p(Y_k|X_k)p(X_k|Y_{1:k-1})}{p(Y_k|Y_{1:k-1})}. \tag{13}$$

This equation updates our beliefs about the state based on new measurements. It is shown that the posterior density can be written as [17]:

$$p(X_k|Y_{1:k}) \approx \sum_{i=1}^{N_p} w_k^i \delta_{X_k^i}(X_k). \tag{14}$$

SMCMC methods then generate samples from the posterior density. One common technique is the Metropolis-Hastings algorithm which involves proposing a new particle $\tilde{X}_k^j$ at step k, where $j \in [1, N_b + N_p]$, and accepting or rejecting it based on a defined acceptance probability:

$$A(\tilde{X}_k^j, X_k^{j-1}) = \min\left(1, \frac{p(\tilde{X}_k^j|Y_{1:k})}{p(X_k^{j-1}|Y_{1:k})} \frac{q(X_k^{j-1}|\tilde{X}_k^j)}{q(\tilde{X}_k^j|X_k^{j-1})}\right), \tag{15}$$

where $\tilde{X}_k^j$ is the proposed particle, X_k^{j-1} is the current accepted particle and $q(\tilde{X}_k^j|X_k^{j-1})$ is the proposal density. The Markov Chain method is visually explained in Figure 3, providing an intuitive showcase of its principles:

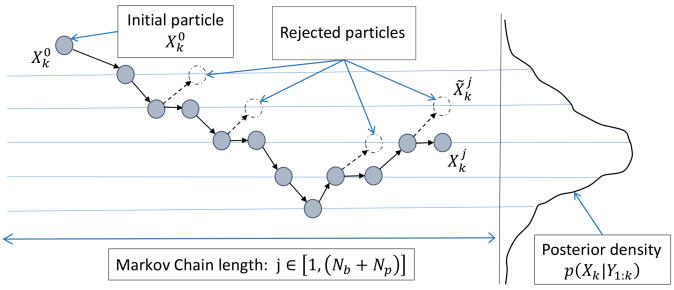


Fig. 3: Visual representation of the Metropolis-Hastings Markov Chain approach.

Therefore, this method computes the acceptance probability for the proposed particle based on the ratio of the posterior and the proposal densities. The sampling process at step k works as follows:

Algorithm 1: Metropolis-Hastings algorithm

1) **Sample:**
   - Initialize the chain with arbitrary value X^0
2) **Iterate:**
   - Propose a new particle $\tilde{X}^j \sim q(\tilde{X}|X^{j-1})$
   - Compute the acceptance ratio $A(\tilde{X}^j, X^{j-1})$
   - Generate a random variable $u \sim \mathcal{U}(0,1)$
   - If $u < A(\tilde{X}^j, X^{j-1})$, accept the proposed particle, $X^j = \tilde{X}^j$; otherwise, reject it, $X^j = X^{j-1}$

III. Sequential Markov Chain Monte Carlo Filter on Lie Groups

The Sequential Markov Chain Monte Carlo (SMCMC) Metropolis-Hastings method on Lie groups, as outlined in this section, operates by iteratively updating the states of a system

based on a sequence of samples. The primary contribution of this paper lies in establishing analytical formulas for this method specifically tailored to Lie groups.

Algorithm 2 involves two main steps: the iterative particle propagation step and the Markov chain generation step. These steps are influenced by the burn-in parameter denoted as N_b, which plays a crucial role in determining the algorithm's behavior. It characterizes the number of iterations required for the filter to converge. The higher the value of N_b, the closer the estimated density will be to the true density.

The initialization step involves sampling $\epsilon^{1:N_p}$ particles, which are then reshaped into the Lie groups framework using a left initial mean, denoted as μ_0. This mean is computed based on the initial true state μ_0^{true}, such that $\mu_0 = \mu_0^{\text{true}} + n_{\text{initial}}$. Initial dispersion values are shown in Table II.

In the propagation step, the algorithm propagates the particles by using a model that describes the dynamics of the system. This step generates $(N_b + N_p)$ proposals for the next state, leveraging both the current state and the dynamics model, with some process noise n_{k-1}^q as outlined in Table III.

Subsequently, in the generation step, the proposed particles are evaluated and either accepted or rejected based on a criterion that balances exploration and exploitation. The Metropolis-Hastings algorithm achieves this by comparing the likelihood of the proposed particle to that of the current state, along with incorporating a random factor to explore the state space effectively.

Finally, the mean and covariance matrix on Lie groups are computed. Here, $(\mu_{k+1})^{-1}$ denotes the $SE_2(3)$ inverse of the mean μ_{k+1} at step $k+1$, which is eventually used for the convergence criterion.

Algorithm 2: (Left) Sequential Monte Carlo Markov Chain on Lie Groups

Initialization

- Initial sample: $\epsilon^{1:N_p} \sim \mathcal{N}_{\mathbb{R}^d}(0, Id)$
- Left mean displacement: $X_0^{1:N_p} = \mu_0 \exp_G^{\wedge}(\epsilon^{1:N_p})$
- Initial Weights: $w_0^{1:N_p} = 1/N_p$

Time loop: for $k = 1 : T$

1) **Propagation step:**
   Loop: for $j = 1 : (N_b + N_p)$
   - Generate a random variable: $I \sim \mathcal{U}(1, N_p)$
   - Spread particles: $\tilde{X}_k^j = f(X_{k-1}^I, n_{k-1}^{q^j})$
   - Compute weight with likelihood: $\tilde{w}_k^j = p(Y_k|\tilde{X}_k^j)$

   End loop
2) **Generation step:**
   Initialize Markov chain: $\overline{X}_k^1 = \tilde{X}_k^1$ and $\overline{w}_k^1 = \tilde{w}_k^1$
   Loop: for $j = 2 : (N_b + N_p)$
   - Generate a random variable: $u \sim \mathcal{U}(0, 1)$
   - Compute the acceptance probability $A(\tilde{X}_k^j, \overline{X}_k^{j-1})$
   - If $u < A(\tilde{X}_k^j, \overline{X}_k^{j-1})$
     - Accept the proposal: $\overline{X}_k^j = \tilde{X}_k^j$
     - Otherwise, reject it: $\overline{X}_k^j = \overline{X}_k^{j-1}$

   End loop
3) **State estimation**
   - Retain last particles: $X_k^{1:Np} = \overline{X}_k^{N_b+1:N_b+N_p}$
   - Retain last weights: $\underline{w_k}^{1:Np} = \overline{w}_k^{N_b+1:N_b+N_p}$
   - Normalize weights: $w_k^{1:Np} = \underline{w_k}^{1:Np} / \sum_{i=1}^{N_p} \underline{w_k}^i$
   - The mean is given by μ_k such that: $\sum_{i=1}^{N_p} w_k^i \log_{\mathcal{G}}^{\vee}(\mu_k^{-1} X_k^i) = 0$
   - The matrix covariance P_k is given by: $\sum_{i=1}^{N_p} w_k^i \log_{\mathcal{G}}^{\vee}\left(\mu_k^{-1} X_k^i\right) \log_{\mathcal{G}}^{\vee}\left(\mu_k^{-1} X_k^i\right)^T$

End loop

IV. Application to Navigation

A. Context and Modelling

In this section, we apply the Monte Carlo method introduced in Section III to the navigation of a UAV, with a simulated trajectory depicted in Figure 4. This trajectory is designed to emulate a short flight of duration $75.1s$, where the total time step $T = 751$ and the integration time step $dt = 0.1$. The UAV has a maximum speed of 160 $m.s^{-1}$.

In our model, we establish the Earth frame $[e]$ as fixed relative to the ground. Concurrently, the body frame $[b]$ is affixed to the vehicle, with its axes oriented forward, rightward, and downward. The objective of this scenario is to determine the UAV's position x^e, velocity v^e, and attitude represented by the rotation matrix C_b^e. Utilizing discretized kinematic equations of second order, we have:

$$\begin{cases} C_{b,k+1}^e = C_{b,k}^e \exp_{\mathrm{SO(3)}}^{\wedge}\left(\mathrm{dt}\ \omega_{eb}^b + n_{k+1}^{rot}\right), \\ v_{k+1}^e = v_k^e + \mathrm{dt}\left(C_{b,k}^e a_{eb}^b + g^e\right) + n_{k+1}^{vel}, \\ x_{k+1}^e = x_k^e + \mathrm{dt}\ v_k^e + 0.5\ \mathrm{dt}^2\left(C_{b,k}^e a_{eb}^b + g^e\right) + n_{k+1}^{pos}, \end{cases} \tag{16}$$

where $(a_{eb}^b, \omega_{eb}^b)$ represent the inertial acceleration and rotation rates, respectively, as measured by an Inertial Measurement Unit (IMU). The parameter g^e is the magnitude of the acceleration due to gravity and the process noise terms $n_k^q = [n_k^{pos}, n_k^{vit}, n_k^{rot}]^T$ denote centered Gaussian noises. The IMU's measurement frequency and noise characteristics are detailed in Table I.

Throughout the flight, a set of nine beacons is continually observed, each with positions marked as $\{p_1^e, ..., p_9^e\}$. A sensor onboard the UAV captures the angles of arrival (AOA) relative to each beacon, as illustrated in Figure 4. The equations

governing angular measurements are:

$$y = \begin{bmatrix} \arctan 2\left(\Delta^b_{n,y}, \Delta^b_{n,x}\right) \\ \arctan 2\left(\Delta^b_{n,z}, \sqrt{(\Delta^b_{n,x})^2 + (\Delta^b_{n,y})^2}\right) \end{bmatrix} + n^r, \quad (17)$$

where $\Delta^b_n = [\Delta^b_{n,x}, \Delta^b_{n,y}, \Delta^b_{n,z}]^T = C^b_e(p^e_n - x^e)$ is the relative distance between a landmark and the UAV resolved in the body frame [b], n^r is a white centered Gaussian noise which density is given in Table I, and $\arctan 2(y, x)$ is such that $\forall (x, y) \neq (0, 0)$:

$$\arctan 2(y, x) = \begin{cases} \text{sign}(y) \arctan \left|\frac{y}{x}\right| & x \leq 0, \\ \text{sign}(y) \frac{\pi}{2} & x = 0, \\ \text{sign}(y)(\pi - \arctan \left|\frac{y}{x}\right|) & x \leq 0. \end{cases} \quad (18)$$

Sensor (IMU)	**Noise (1σ)**	**Rate**
Accelerometer	10^{-4} m.s^{-2}	10 Hz
Gyrometer	10^{-5} rad.s^{-1}	10 Hz
Sensor (AOA)	**Noise (1σ)**	**Rate**
AOA	1.15 deg	10 Hz

TABLE I: Parameters and rate of the simulated sensors.

B. Implementing and comparing the filters

This paper applies the Lie groups framework to two filters, the (Left) Lie Group Particle filter (L)LG-PF and the (Left) Lie Group Sequential Markov Chain Monte Carlo filter (L)LG-SMCMC. To compare the filters, we will use the Root Mean Square Error (RMSE) which is a measure of the average deviation between the estimated values produced by a filter and the true values of the system states. In the case of state estimation, if $\mathbf{e}^m_k$ represents the error vector of convergent Monte Carlo run m at time k, and N_{conv} represents the number of convergent runs, then RMSE at time step k can be expressed as:

$$\text{RMSE}_k = \sqrt{\frac{1}{N_{conv}} \sum_{m=1}^{N_{conv}} (e^m_k)^2}, \quad (19)$$

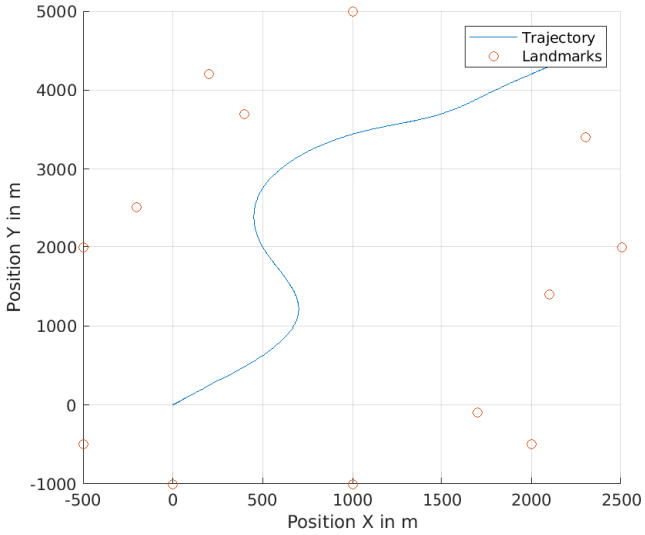
where $\mathbf{e}^m_k$ is defined as:

$$e^m_k = \begin{cases} \log^{\vee}_{\text{SO(3)}}\left((C^e_{b,k})^{-1} \hat{C}^{e,m}_{b,k}\right), \\ v^e_k - \hat{v}^{e,m}_k, \\ x^e_k - \hat{x}^{e,m}_k. \end{cases} \quad (20)$$

This equation provides a quantitative measure of how well the filter's estimated values match the true values of the system states over time. Lower values of RMSE shows better performance, as they indicate smaller deviations between the estimated and true values.

The estimated state matrix for Left and Right LG-SMCMC belongs to the group $SE_2(3)$ [10]:

$$X_k = \begin{pmatrix} C^e_{b,k} & v^e_k & x^e_k \\ 0_{1\times 3} & 1 & 0 \\ 0_{1\times 3} & 0 & 1 \end{pmatrix}. \quad (21)$$



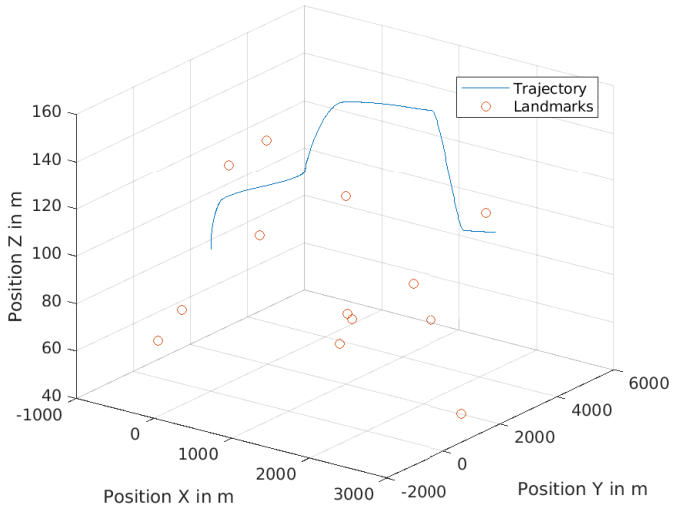


Fig. 4: Top down and 3D simulated trajectory. The landmarks are placed randomly around the trajectory and simulate some characteristic features in the UAV environment.

On the other hand, Euclidean state is a vector from $\mathbb{R}^9$ where the attitude is represented by Euler angles:

$$x_k = \begin{pmatrix} x^e_k \\ v^e_k \\ \theta^e_{b,k} \end{pmatrix}. \quad (22)$$

The implementation of these filters and the computation of the Jacobian matrices is detailed in [9].

C. Simulations

The measurements and process noise tuning of the filters are outlined in Table III, the trajectory is shown in Figure 4 and the initial dispersion of the particles is defined in Table II.

Initial uncertainties	Standard deviation (1σ)
Position	300 m
Velocity	5 m.s^{-1}
Attitude angles	5.73 deg

TABLE II: Initial particle dispersion of the filters, denoted as n_{initial} and following a Gaussian density with zero mean and a standard deviation specified in this table.

Process noise	Standard deviation (1σ)
Position	10 m
Velocity	0.1 m.s^{-1}
Attitude angles	0.573 deg
Measurement noise	**Standard deviation (1σ)**
AOA	1.15 deg

TABLE III: Standard deviations of the filters, consistent across all filter types.

The simulations were executed across 100 Monte Carlo iterations for a total of $N_p = N_b = 500$ particles, comparing left (L) non-Euclidean methods against their Euclidean counterparts. Only results from convergent simulations are analyzed to eliminate statistical outliers. From now on, the filter fails to converge if during the final five consecutive measurement time-steps, the position state estimate $\hat{x}_k^e$ exits the confidence ellipsoid Γ_k, defined by the position covariance matrix P_k^e and calculated as per the Mahalanobis distance convergence criterion:

$$\Gamma_k = \{x_k^e | (x_k^e - \hat{x}_k^e)^T (P_k^e)^{-1} (x_k^e - \hat{x}_k^e) \leq \alpha_{th}^2\}, \qquad (23)$$

where the threshold α_{th} is such that $\mathbb{P}(\chi^2(3) \leq \alpha_{th}^2) = 0.99$ with χ^2 the Chi-squared distribution. Thus, convergence rates in Table IV denote the proportion of convergent runs compared to the total number of Monte Carlo simulations conducted.

Filters	Convergence rates
PF	51%
SMCMC	79%
(L)LG-PF	81%
(L)LG-SMCMC	90%

TABLE IV: Convergence rates of tested filters for 100 MC.

The RMSE values for the simulations regarding position, velocity, and attitude angles are detailed in Table V. To enhance readability, the table provides the norm for position and velocity instead of their individual components.

Filters	PF	SMCMC	(L)LG-PF	(L)LG-SMCMC
Position [m]	20.03	19.84	11.56	8.86
Velocity [m/s]	11.26	9.83	11.17	7.62
Roll [deg]	0.404	0.734	0.361	0.162
Pitch [deg]	0.574	0.958	0.366	0.159
Yaw [deg]	0.492	0.592	0.346	0.163

TABLE V: Average final root mean square error norm of the position and velocity, as well as attitude angles, for convergent runs.

As shown in Table V, SMCMC variants of the filters consistently yield significantly lower average errors in

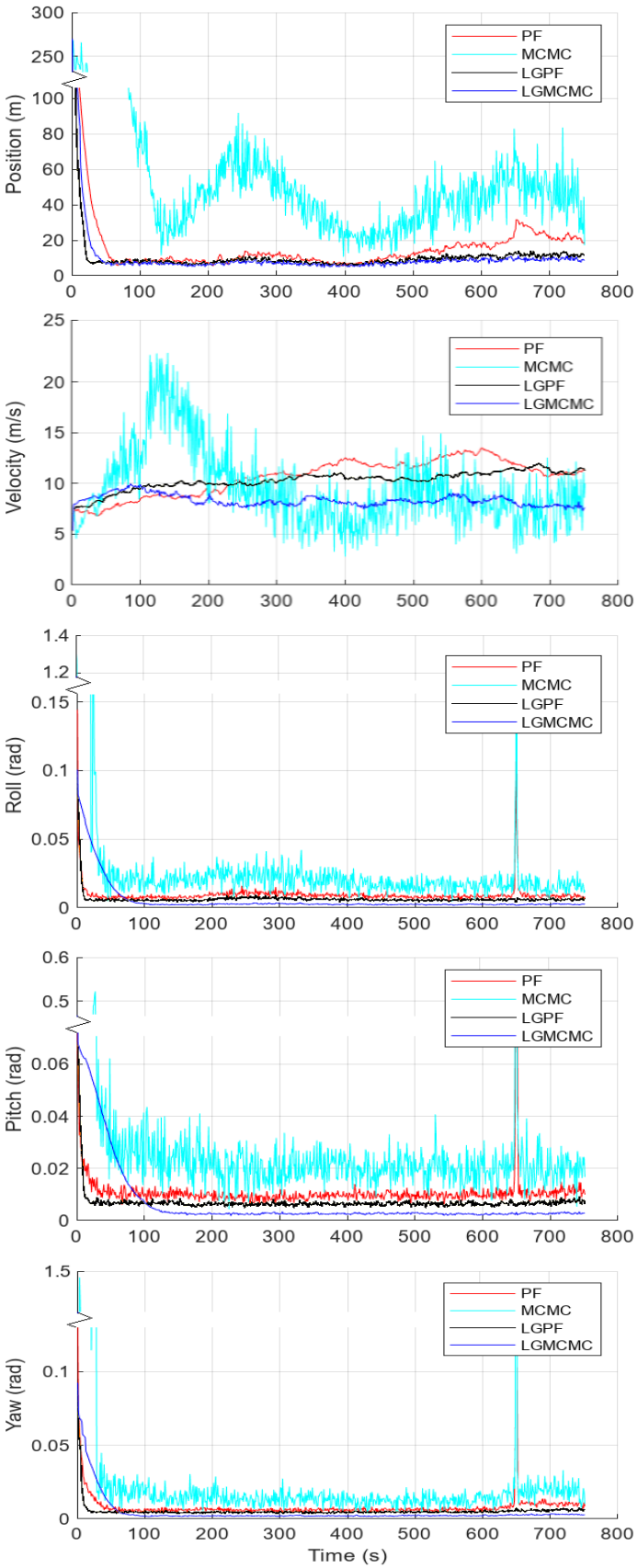


Fig. 5: RMSE of several filters for $N_p = 500$ particles. The results are displayed for 100 Monte Carlo runs.

position, velocity and attitude angles. These enhancement are attributed to the burn-in parameter, a crucial step that generates state proposals by maximizing likelihood.

Furthermore, SMCMC filters exhibit increased convergence rates on a small number of particles, showing once more the effectiveness of these methods in the realm of filtering algorithms.

In the comparative analysis conducted in Figure 5, Lie group filters are shown to be very precise and robust, outperforming their noisy Euclidean counterparts that struggle with angle estimation. This is notably evident at $k = 650$ seconds, where a peak error arises due to the rapid decline of the trajectory. Lie groups particle filter performance is further enhanced when coupled with SMCMC techniques, owing to their ability to accurately represent those rotations.

Therefore, (L)LG-SMCMC addresses the main drawbacks of particles filters (e.g. high number of particles needed, weight degeneracy) while preserving the advantages of filters on Lie Groups (e.g. angles error management, computational cost) with promising results.

V. Conclusion

This paper introduces a new formulation for the Sequential Monte Carlo Markov Chain filter on Lie groups. This new framework is a generalization of the SMCMC, as it also applies to filters designed on the Euclidean space. Besides, it demonstrated improved accuracy and robustness compared to Euclidean methods on a simulated scenario with challenging discrepancies in the noise tuning. Future work will focus on applying this framework to complex scenarios with higher dimensional states to further exploit the advantages of enhanced SMCMC filters on Lie groups.

Acknowledgement

The authors would like to express their gratefulness to the Innovation and Defense Agency (Agence Innovation Defense) of the French Ministry of Defense for their support in this research endeavor.

References

[1] H. Snoussi and A. Mohammad-Djafari, "Particle filtering on riemannian manifolds," *AIP Conference Proceedings*, vol. Vol. 872, pp. 219–226, 2006.

[2] C. Zhang, A. Taghvaei, and P. G. Mehta, "Feedback particle filter on riemannian manifolds and matrix lie groups," *IEEE Transactions on Automatic Control*, 2017.

[3] C. Chahbazian, N. Merlinge, K. Dahia, B. Winter-Bonnet, K. Honore, and C. Musso, "Improved kalman-particle kernel filter on lie groups applied to angles-only navigation," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1689–1694, 2022.

[4] C. Chahbazian, N. Merlinge, K. Dahia, B. Winter-Bonnet, J. Marini, and C. Musso, "Laplace particle filter on lie groups applied to angles-only navigation," *IEEE 24th International Conference on Information Fusion*, pp. 1–8, 2021.

[5] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte-carlo sampling methods for bayesian filtering," *Stat. and Comput*, vol. 10, p. 197–208, 2000.

[6] F. Septier and G. W. Peters, "Langevin and hamiltonian based sequential mcmc for efficient bayesian filtering in high-dimensional spaces," *IEEE Journal of Selected Topics in Signal Processing*, 2016.

[7] H. Tjelmeland, "Using all metropolis–hastings proposals to estimate mean values," 2004. [Online]. Available: http://www.math.ntnu.no/preprint/statistics/2004/S4-2004.ps

[8] J. Hilgert and K.-H. Neeb, *Structure and Geometry of Lie Groups*, Springer, Ed. Springer, 2011.

[9] C. Chahbazian, "Particle Filtering on Lie Groups : Application to Navigation," Theses, Université Paris-Saclay, May 2023. [Online]. Available: https://theses.hal.science/tel-04154275

[10] A. Barrau, "Non-linear State Error Based Extended Kalman Filters with Applications to Navigation," Ph.D. dissertation, Ecole Nationale Supérieure des Mines de Paris, 2015.

[11] G. Bourmaud, "Estimation de pramètres évoluant sur des groupes de Lie : application à la cartographie et à la loalisation d'une caméra monoculaire," Ph.D. dissertation, Université de Bordeaux, 2015.

[12] G. Chirikjian and M. Kobilarov, "Gaussian approximation of non-linear measurement models on Lie groups," *53rd IEEE Conference on Decision and Control*, pp. 6401–6406, 2014.

[13] C. Chahbazian, "Lie Groups Extended Kalman Filter," *IEEE Trans. Signal Process.*, vol. 46, pp. 1386–1396, 2023.

[14] N. Miolane, "Defining a mean on lie group," *Medical Imaging - https://inria.hal.science/hal-00938320*, 2013.

[15] E. Hairer, "Introduction à l'analyse numérique," 2001.

[16] C. Kuptametee and N. Aunsri, "A review of resampling techniques in particle filtering framework," *ISSN 0263-2241*, vol. 193, 2022.

[17] C. Andrieu, A. Doucet, and R. Holenstein, "Particle markov chain monte carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, p. 269–342, 2010.