Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7092526, 6 pages
https://doi.org/10.1155/2020/7092526

Research Article

Pipeline Gaussian Particle Filter and Hardware Design for Attitude Estimation with UAV

Yali Xue, Hu Chen, Jie Chen, and Jiahui Wang

College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China

Correspondence should be addressed to Yali Xue; xueyali@nuaa.edu.cn

Received 3 October 2019; Revised 21 April 2020; Accepted 22 April 2020; Published 11 May 2020

Academic Editor: Xiangyu Meng

This paper based on the Gaussian particle filter (GPF) deals with the attitude estimation of UAV. GPF algorithm has better estimation accuracy than the general nonlinear non-Gaussian state estimation and is usually used to improve the system's real-time performance whose noise is specific such as Gaussian noise during the mini UAV positioning and navigation. The attitude estimation algorithm is implemented on FPGA to verify the effectiveness of the Gaussian particle filter. Simulation results have illustrated that the GPF algorithm is effective and has better real-time performance than that of the particle filter.

1. Introduction

Mini UAV (unmanned aerial vehicle) has got a rapid development in the military and civilian fields for their low cost, rescue, and geographic surveying [1]. UAV has a high requirement on accurate and real-time positioning and navigation because of flight safety and reliability. Fortunately, GPF algorithm has better estimation accuracy than the general nonlinear non-Gaussian state estimation and is usually used to improve the system's real-time performance whose noise is specific such as Gaussian noise during the mini UAV positioning and navigation. The attitude estimation of UAV is usually estimated by nonlinear filtering that many scholars are committed to study such as FLANN filter [2], Volterra filter [3], neural network filter [4], and Affine projection algorithm [5]. GPF is a particle filter which is based on a Gaussian filter framework and is suitable for parallel implementation since it does not require resampling. [6]. It assumes the probability density function of the estimated state quantity as the multidimensional Gaussian distribution. Furthermore, the Gaussian particle filter is used to solve Gaussian distribution-related parameters.

In the face of UAV attitude estimation problem, we can assume that the process noise and the observation noise are both Gaussian noise and can be processed by GPF. This paper applies GPF to the attitude estimation of UAV, which has the following advantages: First, GPF needs to sample particles from the Gaussian distribution so that the particles are diverse. Second, the approximate expected probability density of the GPF algorithm is Gaussian, so there is not the problem of particle depletion and degradation in the standard particle filter. Here, the resampling step is unnecessary. Third, GPF has not only the asymptotic optimality but also a simple logic structure which is easy to be implemented in hardware. All these advantages have brought more attention to GPF.

2. Model Equations

This section will introduce the attitude and velocity solution described by the quaternion method first. The geodetic coordinate system (east-north-sky) is used as the navigation frame, and the aircraft is taken as the research object. The measured data of the inertial measurement unit (IMU) and the initial attitude of the aircraft will be given. Real-time attitude and velocity estimation will be run on every time period. The quaternion description $\boldsymbol{\omega}_{gf}^{f}$ is the angular velocity of the geodetic frame to the body-fixed frame under the body-fixed frame. Together with the quaternion $\boldsymbol{\Lambda}$ corresponding to the rotation matrix $\mathbf{C}_{g}^{f}$, they can be expressed by the kinematics differential equation as

$$\dot{\boldsymbol{\Lambda}} = \frac{1}{2}\boldsymbol{\Lambda} \circ \boldsymbol{\omega}_{gf}^{f}. \tag{1}$$

Based on quaternion multiplication, equation (1) can be written as

$$\mathbf{Q}(\dot{\Lambda}) = \frac{1}{2}\mathbf{M}(\Lambda)\mathbf{Q}\left(\boldsymbol{\omega}_{gf}^{f}\right) = \frac{1}{2}\mathbf{M}^{*}\left(\boldsymbol{\omega}_{gf}^{f}\right)\mathbf{Q}(\Lambda), \tag{2}$$

where

$$\begin{bmatrix}\dot{\lambda}_0\\ \dot{\lambda}_1\\ \dot{\lambda}_2\\ \dot{\lambda}_3\end{bmatrix} = \frac{1}{2}\times\begin{bmatrix}0 & -\boldsymbol{\omega}_{gfx}^{f} & -\boldsymbol{\omega}_{gfy}^{f} & -\boldsymbol{\omega}_{gfz}^{f}\\ \boldsymbol{\omega}_{gfx}^{f} & 0 & \boldsymbol{\omega}_{gfz}^{f} & -\boldsymbol{\omega}_{gfy}^{f}\\ \boldsymbol{\omega}_{gfy}^{f} & -\boldsymbol{\omega}_{gfz}^{f} & 0 & \boldsymbol{\omega}_{gfx}^{f}\\ \boldsymbol{\omega}_{gfz}^{f} & \boldsymbol{\omega}_{gfy}^{f} & -\boldsymbol{\omega}_{gfx}^{f} & 0\end{bmatrix}\begin{bmatrix}\lambda_0\\ \lambda_1\\ \lambda_2\\ \lambda_3\end{bmatrix}. \tag{3}$$

$\lambda_0 \sim \lambda_3$ are the elements of Λ. For equation (3), it can be discretized as

$$\mathbf{Q}(\Lambda_k) - \mathbf{Q}(\Lambda_{k-1}) = \frac{\Delta t}{2}\mathbf{M}^{*}\left({}_{k-1}\boldsymbol{\omega}_{gf}^{f}\right)\mathbf{Q}(\Lambda_{k-1}). \tag{4}$$

Therefore, the updating equation of discrete state quaternion can be expressed as

$$\mathbf{Q}(\Lambda_k) = \left(\mathbf{I}_{4\times4} + \frac{\Delta t}{2}\mathbf{M}^{*}\left({}_{k-1}\boldsymbol{\omega}_{gf}^{g}\right)\right)\mathbf{Q}(\Lambda_{k-1}). \tag{5}$$

In strapdown inertial navigation system (SINS), the specific force equation can be calculated from the following equation:

$$\dot{\mathbf{v}}^{c} = \mathbf{f}^{c} + \mathbf{g}^{c}, \tag{6}$$

where $\mathbf{v}^{c}$ is the velocity vector based on the navigation frame $\mathbf{f}^{c}$is the accelerometer output which is transformed by $\mathbf{C}_{f}^{g}$, and $\mathbf{g}^{c}$ is the local gravitational acceleration. Similarly, the force equation can be rewritten as the following discrete expression:

$$\mathbf{v}_k^{c} - \mathbf{v}_{k-1}^{c} = \left(\mathbf{C}(\Lambda_{k-1})_{f}^{g}\left({}_{f}\mathbf{a}_{k-1}\right) + \mathbf{g}^{c}\right)\Delta t. \tag{7}$$

3. Gaussian Particle Filter

The basic theory of GPF is not given in this section and has been discussed in [7]. The algorithm pseudocode of the generic Gaussian particle filter is shown in Pseudocode 1.

For the particle filter-based attitude estimation algorithm proposed in this section, the state of the ith particle is selected as

$$\mathbf{x}_k^{(i)} = \left(\mathbf{q}_k^{(i)}, {}_{g}\mathbf{v}_k^{(i)}\right), \tag{8}$$

where $\mathbf{q}_k^{(i)}$ is the attitude described by the quaternion and ${}_{g}\mathbf{v}_k^{(i)}$ is the velocity vector under geodetic frame. In order to obtain a suitable approximation of the target distribution, the following conditions are necessary:

(i) The probability distribution of the initial state of the system $p(\mathbf{x}_0)$

(ii) A proposal distribution $q(\mathbf{x}_k \mid \mathbf{x}_{k-1}, \mathbf{y}_k, \mathbf{u}_k)$ for sampling particles

(iii) For a given state $\mathbf{x}_k$, a method for calculating observations $\mathbf{y}_k$

For the designed particle filter, it is mainly divided into two steps: prediction and update.

3.1. Prediction Step of Particle Filter. Suppose $p(\mathbf{x}_0)$ is known, then an approximate distribution of the target can be obtained by sampling the particles as follows:

$$\mathbf{x}_k^{(i)} \sim q\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k, \mathbf{u}_k\right), \quad i = 1, \ldots, N, \tag{9}$$

where the proposal distribution is

$$q\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}, \mathbf{y}_k, \mathbf{u}_k\right) = p\left(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}\right). \tag{10}$$

Equations (6) and (8) already give out the quaternion-based attitude and velocity solution for discrete equations, use it as sampling strategy, and are rewritten for easy understanding:

$$\begin{aligned}\mathbf{q}_k^{(i)} &= \left(\mathbf{I}_{4\times4} + \frac{\Delta t}{2}\boldsymbol{\Omega}\left({}_{f}\boldsymbol{\omega}_{k-1}^{(i)}\right)\right)\mathbf{q}_{k-1}^{(i)} + \mathbf{w}_q^{(i)},\\ {}_{g}\mathbf{v}_k^{(i)} &= {}_{g}\mathbf{v}_{k-1}^{(i)} + \left(\mathbf{C}\left(\mathbf{q}_{k-1}^{(i)}\right)_{f}^{g}\left({}_{f}\mathbf{a}_{k-1} + \mathbf{w}_a^{(i)}\right) + \mathbf{g}\right)\Delta t + \mathbf{w}_{\mathbf{v}}^{(i)},\end{aligned} \tag{11}$$

where $\mathbf{I}_{4\times4}$ is the unit matrix and $\mathbf{C}(\mathbf{q}_{k-1}^{(i)})_{f}^{g}$, $\boldsymbol{\Omega}({}_{f}\boldsymbol{\omega}_{k-1}^{(i)})$ is the matrix as

$$\boldsymbol{\Omega}\left({}_{f}\boldsymbol{\omega}_{k-1}^{(i)}\right) = \begin{bmatrix}0 & -{}_{f}\boldsymbol{\omega}_{xk-1}^{(i)} & -{}_{f}\boldsymbol{\omega}_{yk-1}^{(i)} & -{}_{f}\boldsymbol{\omega}_{zk-1}^{(i)}\\ {}_{f}\boldsymbol{\omega}_{xk-1}^{(i)} & 0 & {}_{f}\boldsymbol{\omega}_{zk-1}^{(i)} & -{}_{f}\boldsymbol{\omega}_{yk-1}^{(i)}\\ {}_{f}\boldsymbol{\omega}_{yk-1}^{(i)} & -\boldsymbol{\omega}_{zk-1}^{(i)} & 0 & {}_{f}\boldsymbol{\omega}_{xk-1}^{(i)}\\ {}_{f}\boldsymbol{\omega}_{zk-1}^{(i)} & {}_{f}\boldsymbol{\omega}_{yk-1}^{(i)} & -{}_{f}\boldsymbol{\omega}_{xk-1}^{(i)} & 0\end{bmatrix}, \tag{12}$$

where ${}_{f}\boldsymbol{\omega}_{k-1}^{(i)} = {}_{f}\boldsymbol{\omega}_{k-1} + \mathbf{w}_{\boldsymbol{\omega}}^{(i)}$ and $[\bullet]_{\times}$ represents matrix multiplication. The filter inputs ${}_{f}\boldsymbol{\omega}_{k-1}$ and ${}_{f}\mathbf{a}_{k-1}$ are the triaxial angular velocity under the body-fixed frame and specific force, respectively, and $\mathbf{w}_{\boldsymbol{\omega}}^{(i)}$, $\mathbf{w}_q^{(i)}$, $\mathbf{w}_a^{(i)}$, and $\mathbf{w}_{\mathbf{v}}^{(i)}$ are obtained from the proposal distribution to express the uncertainty in the prediction step. The uncertainty of the prediction step usually includes the following two cases:

(a) The uncertainty of filter input: ${}_{f}\boldsymbol{\omega}_{k-1}$ and ${}_{f}\mathbf{a}_{k-1}$are always with noises, and the noises belong to $\mathbf{w}_a^{(i)} \sim \mathcal{N}(0, \sigma_a^2)$ and $\mathbf{w}_{\boldsymbol{\omega}}^{(i)} \sim \mathcal{N}(0, \sigma_{\boldsymbol{\omega}}^2)$.

(b) The numerical integration error and the small angle approximation error: the simplification and approximation to the quaternion updating equation cause some errors in equations (5) and (7), and the error belongs to $\mathbf{w}_q^{(i)} \sim N(0, \sigma_q^2)$ and $\mathbf{w}_{\mathbf{v}}^{(i)} \sim N(0, \sigma_{\mathbf{v}}^2)$.

3.2. Updating Step of the Velocity Weight. In this updating step, the GPS velocity output is used to update the velocity weight. Assume that the discrete approximation $p(\mathbf{x}_k \mid \mathbf{y}_{1:k}, \mathbf{u}_{1:k})$ is only relevant to the current moment of the measurement $\mathbf{y}_k$. By using the following equation, the updating weight can be recured:

$$\mathbf{w}_k^{(i)} = \mathbf{w}_{k-1}^{(i)} p\left(\mathbf{y}_k \mid \mathbf{x}_k^{(i)}\right). \tag{13}$$

Input: observation $\mathbf{y}_k$, previous estimation $\boldsymbol{\mu}_{k-1}$, and Cholesky decomposition $\mathbf{C}_{k-1}$ of covariance $\boldsymbol{\Sigma}_{k-1}$;
Initial: mean $\boldsymbol{\mu}_0$ and $\boldsymbol{\Sigma}_0$ from initial estimation
Method:
(1) Drawing conditioning particles from $\mathcal{N}(\mathbf{x}_{k-1}; \boldsymbol{\mu}_{k-1}, \boldsymbol{\Sigma}_{k-1})$ to obtain $\left\{\mathbf{x}_{k-1}^{(i)}\right\}_{i=1}^{M}$
(2) Drawing particles: draw particles from $p(\mathbf{x}_k \mid \mathbf{x}_k = \mathbf{x}_{k-1}^{(i)})$ to obtain $\left\{\mathbf{x}_k^{(i)}\right\}_{i=1}^{M}$
(3) (a) Weight calculation $\widetilde{w}_n^{(i)} = p(\mathbf{y}_k \mid \mathbf{x}_k^{(i)})$
(b)Weight standardization $w_k^{(i)} = \widetilde{w}_k^{(i)} / \sum_{i=1}^{N} \widetilde{w}_k^{(i)}$
(4) Updating mean and covariance by
(a) $\boldsymbol{\mu}_k = \sum_{i=1}^{N} w_k^{(i)} \mathbf{x}_k^{(i)}$
(b) $\boldsymbol{\Sigma}_k = \sum_{i=1}^{N} w_k^{(i)} (\mathbf{x}_k^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}_k^{(i)} - \boldsymbol{\mu}_k)^T$
(5) Cholesky decomposition $\mathbf{C}_{k-1}$ calculation $\boldsymbol{\Sigma}_k = \mathbf{C}_k \mathbf{C}_k^T$

Pseudocode 1: Generic Gaussian particle filter algorithm.

Here, the observation $\mathbf{y}_k$ is obtained by GPS velocity vector ${}_g\widetilde{\mathbf{v}}_k$. The likelihood function is selected as

$$p\left({}_g\widetilde{\mathbf{v}}_k \mid {}_g\mathbf{v}_k^{(i)}\right) = \frac{1}{\sigma_{g\mathbf{v}}\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{\left\|{}_g\widetilde{\mathbf{v}}_k - {}_g\mathbf{v}_k^{(i)}\right\|}{\sigma_{g\mathbf{v}}}\right)^2\right], \quad (14)$$

where $\sigma_{g\mathbf{v}}$ is the standard deviation of ${}_g\widetilde{\mathbf{v}}_k$.

3.3. State Estimation for the UAV's Attitude and Velocity. The state estimation $\widehat{\mathbf{x}}_k = (\widehat{\mathbf{q}}_k, {}_g\widehat{\mathbf{v}}_k)$ is obtained from weighted average to particle sets. We choose the number of particles as N:

$$\begin{aligned} {}_g\widehat{\mathbf{v}}_k &= \sum_{i=1}^{N} \mathbf{w}_k^{(i)}\, {}_g\mathbf{v}_k^{(i)}, \\ \widehat{q}_k &= \frac{1}{\left\|\sum_{i=1}^{N} \mathbf{w}_k^{(i)} \mathbf{q}_k^{(i)}\right\|} \sum_{i=1}^{N} \mathbf{w}_k^{(i)} \mathbf{q}_k^{(i)}. \end{aligned} \quad (15)$$

It is worth noting that the quaternion rotation cannot be averaged because it is not in the same vector space. Furthermore, Gramkow has proved that the quaternion mean from calculating the minimum variance exactly corresponds to the weighted average given by equation (16). And when the rotation angle is less than 40 degrees, the relative error is less than 1% [8].

3.4. Way to Calculate Covariance. Markley has proved that the covariance matrix can also be calculated without loss of generality for nonstandardized quaternions [9]. Since the real quaternion cannot be obtained, the quaternion covariance matrix is represented by the quaternion state of the single particle and the estimated quaternion as shown in equation (17):

$$\boldsymbol{\Sigma}_k = \sum_{k=1}^{M} \mathbf{w}_k^{(m)} \left(\left(\mathbf{q}_k^{(i)} - \widehat{q}_k\right)\left(\mathbf{q}_k^{(i)} - \widehat{q}_k\right)^T\right). \quad (16)$$

The attitude estimation algorithm based on GPF is given as shown in Pseudocode 2.

4. Hardware Implementation

In this section, the attitude estimation algorithm based on the pipeline Gaussian particle filter will be verified on the system generator and finally implemented on ZYNQ-7000 platform.

The value $\widehat{q}_k$ is necessary for covariance calculation, and it can be obtained until the N particles finish the state estimation 4(a) step. In order to combine the covariance matrix calculation with the above steps, the covariance matrix can be modified such as in [10]:

$$\boldsymbol{\Sigma}_k = \frac{1}{W_k} \sum_{k=1}^{M} \mathbf{w}_k^{(m)} \mathbf{q}_k^{(i)} \left(\mathbf{q}_k^{(i)}\right)^T - \widehat{q}_k (\widehat{q}_k)^T. \quad (17)$$

The Gaussian particle filter is modularly decomposed into the following five modules: control module, sampling conditioning particles module, sampling particles module, weight calculation module, mean covariance calculation module, and Cholesky decomposition module.

The following mainly introduces the design of the sampling conditioning particle module, the mean and covariance calculation module, and the Cholesky decomposition module. Others have been realized in [11]. The particle filter modular flow diagram is shown in Figure 1, and the flowing timing diagram is given in Figure 2.

4.1. Sampling Conditioning Particles Module. Here, the $\widehat{q}_k$ and $\mathbf{C}_k$ obtained from the mean and covariance calculation module are used and all multipliers are pipelined. These multipliers generate M particles in a pipeline of $M + L1$ clock cycles. This module requires 4 Gaussian random number generators while there are 16 delays in this module.

4.2. Mean and Covariance Calculation Module. The job of this module is to calculate the mean $\widehat{q}_k$ and the covariance matrix $\sum_k$. $(N_s^2 + 3N_s)/2$ multipliers are enough for the mean and covariance calculation, where N_s is the dimension of the state equation as shown in Table 1. For quaternion attitude estimation, $N_s = 4$ and 14 multipliers are required

Input: particle $\{\mathbf{x}_{k-1}^{(i)}\}_{j=1}^{N}$, IMU output ${}_f\boldsymbol{\omega}_{k-1}^{(i)}$ and ${}_f\mathbf{a}_{k-1}$ GPS, out ${}_g\tilde{\mathbf{v}}_k$, and Cholesky decomposition $\mathbf{C}_{k-1}$ of covariance $\boldsymbol{\Sigma}_{k-1}$: $\boldsymbol{\Sigma}_{k-1} = \mathbf{C}_{k-1}\mathbf{C}_{k-1}^{T}$
Output: attitude and velocity $\hat{q}_k$ and covariance ${}_g\tilde{\mathbf{v}}_k$
(1) Sampling conditioning particles: $\mathbf{q}_{k-1}^{(i)} = \hat{\mathbf{q}}_{k-1} + \mathbf{C}_{k-1}\mathbf{w}_c^{(i)}$
(2) Sampling particles: $\begin{cases} \mathbf{q}_k^{(i)} = (\mathbf{I}_{4\times4} + \Delta t/2\Omega({}_f\boldsymbol{\omega}_{k-1}^{(i)}))\mathbf{q}_{k-1}^{(i)} + \mathbf{w}_q^{(i)} \\ {}_g\mathbf{v}_k^{(i)} = {}_g\mathbf{v}_{k-1}^{(i)} + (\mathbf{C}(\mathbf{q}_{k-1}^{(i)})_f^g({}_f\mathbf{a}_{k-1} + \mathbf{w}_a^{(i)}) + \mathbf{g})\Delta t + \mathbf{w}_{\mathbf{v}}^{(i)} \end{cases}$
(3) (a) Weight calculation: $\mathbf{w}_k^{(i)} = \mathbf{w}_{k-1}^{(i)}/\sigma_{g\mathbf{v}}\sqrt{2\pi}\exp[-(1/2)(\|{}_g\tilde{\mathbf{v}}_k - {}_g\mathbf{v}_k^{(i)}\|/\sigma_{g\mathbf{v}})^2]$
(b) Weight standardization: $\mathbf{w}_k^{(i)} = \mathbf{w}_k^{(i)}/W_k$
(4) (a) State estimation: $\begin{cases} {}_g\hat{\mathbf{v}}_k = \sum_{i=1}^{N}\mathbf{w}_k^{(i)}{}_g\mathbf{v}_k^{(i)} \\ \hat{q}_k = 1/\|\sum_{i=1}^{N}\mathbf{w}_k^{(i)}\mathbf{q}_k^{(i)}\|\sum_{i=1}^{N}\mathbf{w}_k^{(i)}\mathbf{q}_k^{(i)} \end{cases}$
(b) Covariance calculation: $\boldsymbol{\Sigma}_k = \sum_{k=1}^{M}\mathbf{w}_k^{(i)}((\mathbf{q}_k^{(i)} - \hat{q}_k)(\mathbf{q}_k^{(i)} - \hat{q}_k)^T)$
(5) Cholesky decomposition $\mathbf{C}_k$ calculation: $\boldsymbol{\Sigma}_k = \mathbf{C}_k\mathbf{C}_k^T$

Pseudocode 2: An attitude estimation algorithm based on pipeline GPF.

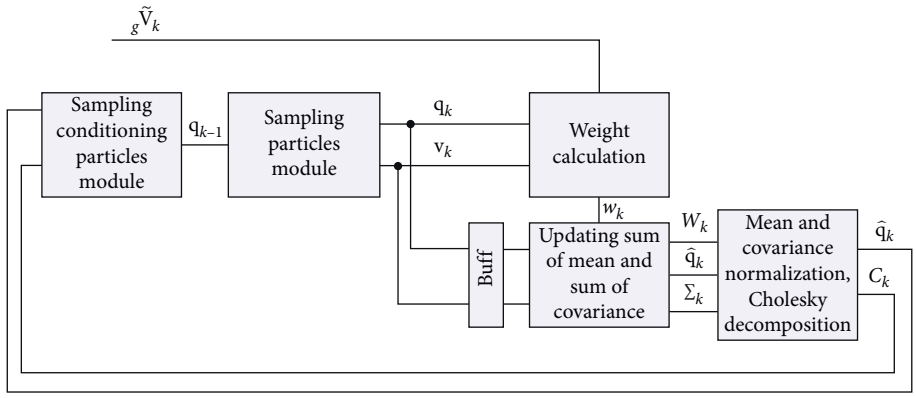

Figure 1: Pipeline Gaussian particle filter algorithm for attitude estimation.

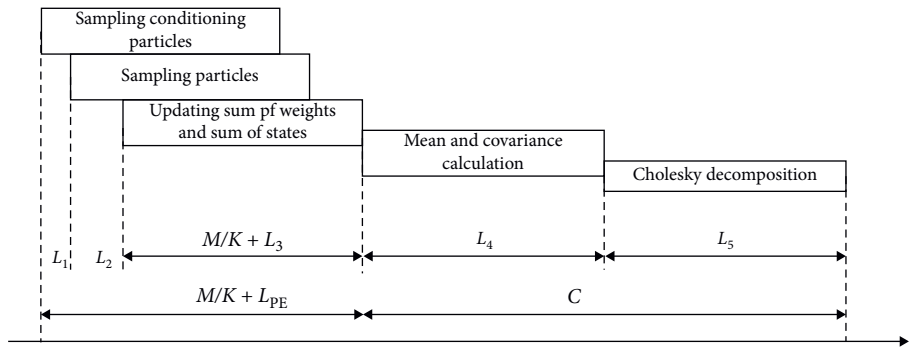

Figure 2: Particle filter modular flowing timing diagram.

too. Since all outputs require their accumulations, 14 accumulations are also required at the same time.

4.3. Cholesky Decomposition Module. The square root and division operation are used to realize the Cholesky decomposition here. This module contains 4 square operations and 6 division operations. Their operations are all implemented by the CORDIC algorithm.

After the above introduction, the final design can be obtained, and Table 2 shows the resource utilization information.

Table 1: Multipliers and accumulations on mean and covariance calculation.

Attitude mean calculation	Covariance calculation	Total
N_s	$(N_s^2 + N_s)/2$	$(N_s^2 + 3N_s)/2$

5. Simulation Analysis

In order to verify the effectiveness of the Gaussian particle filter, the attitude estimation algorithm is designed on the

Table 2: ZYNQ-7000 resource utilization of attitude estimation algorithm based on GPF.

Hardware: Zynq-7000 (7z020clg484-1)			
Resource	Utilization	Total	Utilization rate %
Slice LUTs	10764	53200	20.23
Slice registers	16767	106400	15.75
Block RAMs	6	140	4.29
DSPs	89	220	40.45

Table 3: Gaussian particle filter parameter settings.

Filter parameters	N	σ_w, (deg/s)	σ_a, (m/s^2)	σ_v, (m/s)
Numeric	2048	0.008	0.07	0.02

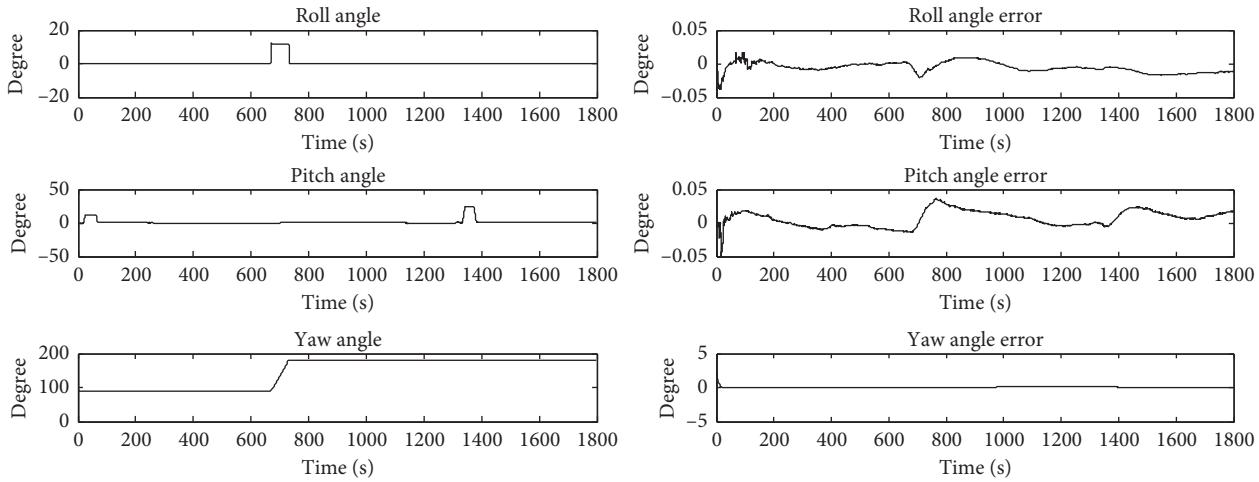

Figure 3: UAV attitude output and output error using Gaussian particle filter based on FPGA.

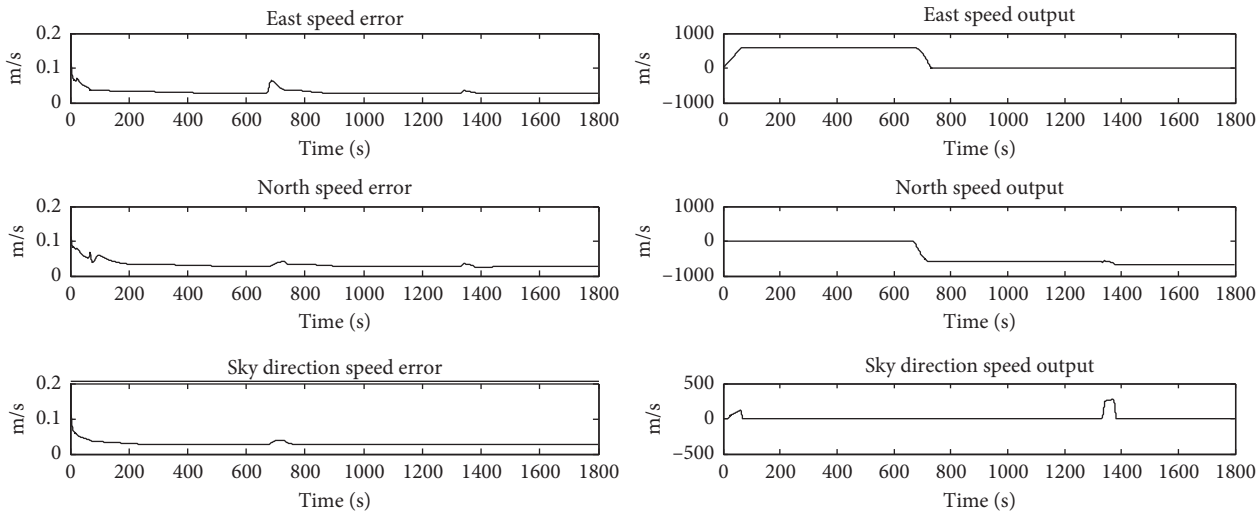

Figure 4: UAV velocity output and velocity output error of GPF based on FPGA.

FPGA. And an existing UAV track model is adopted for simulation. Numerical values are represented by 24-bit Fixed-Point numbers. The quaternion consists of 1 sign bit, 3 integer bits, and 20 decimal bits, and it is recorded as FIX24_20. Similarly, the geography velocity is recorded as FIX24_13 and the weight is UFIX24_23. Filter parameter settings are shown in Table 3. Simulation results represent the attitude output, attitude output error, velocity, and velocity error under geodetic frame based on FPGA platform, respectively, in Figures 3 and 4.

The above simulation results (Figures 3 and 4) show that the UAV attitude estimation algorithm of GPF based on FPGA is feasible and effective. While due to the setting of the fixed-point number and the approximate solution on the

FPGA platform, the hardware simulation result is with certain errors. The errors can be only improved by a lot of board-level debugging work.

6. Conclusion

The resampling algorithm is very complex and not difficult for parallel implementation. So it is considered to apply the Gaussian particle filter for attitude estimation algorithm in this paper. The quaternion discrete equation is regarded as the state equation here. This algorithm consists of sample adjustment particles, sampled particles, weight calculation, mean covariance calculation, Cholesky decomposition five modules, and the quaternion "average" value and covariance matrix which are calculated by nonstandardized weights. An improved pipeline Gaussian particle filtering algorithm is deduced through rewriting the covariance matrix calculation formula. The final simulation shows the result effectiveness. The Gaussian particle filter presented in this paper is not limited to the application of UAV attitude estimation but also in providing the reference value and practical applications on other nonlinear estimation problems.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Science Foundation of China (Grant no. 61922042).

References

[1] W. Ding, J. Wang, S. Han et al., "Adding optical flow into the GPS/INS integration for UAV navigation," in *Proceedings of International Global Navigation Satellite Systems Society Symposium*, pp. 1–13, Paradise, Australia, December 2009.

[2] K.-L. Yin, Y.-F. Pu, and L. Lu, "Combination of fractional FLANN filters for solving the van der pol-duffing oscillator," *Neurocomputing*, 2020.

[3] L. Lu, W. Wang, X. Yang, W. Wu, and G. Zhu, "Recursive Geman-McClure estimator for implementing second-order volterra filter," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 66, no. 7, pp. 1272–1276, 2019.

[4] K. Yin, H. Zhao, and L. Lu, "Functional link artificial neural network filter based on the q-gradient for nonlinear active noise control," *Journal of Sound and Vibration*, vol. 435, pp. 205–217, 2018.

[5] L. Lu, G. Zhu, X. Yang, K. Zhou, Z. Liu, and W. Wu, "Affine projection algorithm based high-order error power for partial discharge denoising in power cables," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 4, pp. 1821–1832, 2020.

[6] S. Hong, *Simplifying Physical Realization of Gaussian Particle Filters with Block. Level Pipeline Control*, Hindawi Publishing Corp., London, UK, 2005.

[7] J. H. Kotecha and P. M. Djuric, "Gaussian sum particle filtering," *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2602–2612, 2003.

[8] C. Gramkow, "On averaging rotations," *International Journal of Computer Vision*, vol. 42, no. 1-2, pp. 7–16, 2001.

[9] F. L. Markley, "Attitude estimation or quaternion estimation?" *Journal of the Astronautical Sciences*, vol. 52, no. 1-2, pp. 221–238, 2004.

[10] M. Bolic, *Architectures for Efficient Implementation of Particle filters*, State University of New York at Stony Brook, Stony Brook, NY, USA, 2004.

[11] J. Wang, Y. Wang, and Y. Xue, "Design of simplify particle filter algorithm using xilinx system generator," *Electronics Optics & Control*, in Chinese, 2019.