


Article

# A Combined Method for MEMS Gyroscope Error Compensation Using a Long Short-Term Memory Network and Kalman Filter in Random Vibration Environments

Chenhao Zhu <sup>1,2</sup> , Sheng Cai <sup>1</sup>, Yifan Yang <sup>1,2</sup>, Wei Xu <sup>1</sup>, Honghai Shen <sup>3</sup> and Hairong Chu <sup>1,\*</sup>

<sup>1</sup> Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China; zhuchenhao17@mails.ucas.ac.cn (C.Z.); caisheng@ciomp.ac.cn (S.C.); yangyifan17@mails.ucas.ac.cn (Y.Y.); xuweihi@ciomp.ac.cn (W.X.)

<sup>2</sup> University of Chinese Academy of Sciences, Beijing 100049, China

<sup>3</sup> Key Laboratory of Airborne Optical Imaging and Measurement, Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China; shenh@ciomp.ac.cn

\* Correspondence: chuhr@ciomp.ac.cn

**Abstract:** In applications such as carrier attitude control and mobile device navigation, a micro-electro-mechanical-system (MEMS) gyroscope will inevitably be affected by random vibration, which significantly affects the performance of the MEMS gyroscope. In order to solve the degradation of MEMS gyroscope performance in random vibration environments, in this paper, a combined method of a long short-term memory (LSTM) network and Kalman filter (KF) is proposed for error compensation, where Kalman filter parameters are iteratively optimized using the Kalman smoother and expectation-maximization (EM) algorithm. In order to verify the effectiveness of the proposed method, we performed a linear random vibration test to acquire MEMS gyroscope data. Subsequently, an analysis of the effects of input data step size and network topology on gyroscope error compensation performance is presented. Furthermore, the autoregressive moving average-Kalman filter (ARMA-KF) model, which is commonly used in gyroscope error compensation, was also combined with the LSTM network as a comparison method. The results show that, for the *x*-axis data, the proposed combined method reduces the standard deviation (STD) by 51.58% and 31.92% compared to the bidirectional LSTM (BiLSTM) network, and EM-KF method, respectively. For the *z*-axis data, the proposed combined method reduces the standard deviation by 29.19% and 12.75% compared to the BiLSTM network and EM-KF method, respectively. Furthermore, for *x*-axis data and *z*-axis data, the proposed combined method reduces the standard deviation by 46.54% and 22.30% compared to the BiLSTM-ARMA-KF method, respectively, and the output is smoother, proving the effectiveness of the proposed method.

**Keywords:** MEMS gyroscope; random vibration environments; long short-term memory network; Kalman filter; expectation-maximization algorithm



**Citation:** Zhu, C.; Cai, S.; Yang, Y.; Xu, W.; Shen, H.; Chu, H. A Combined Method for MEMS Gyroscope Error Compensation Using a Long Short-Term Memory Network and Kalman Filter in Random Vibration Environments. *Sensors* **2021**, *21*, 1181. <https://doi.org/10.3390/s21041181>

Academic Editor: Stefano Mariani  
Received: 2 January 2021  
Accepted: 4 February 2021  
Published: 8 February 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Fiber optic gyroscopes and laser gyroscopes have excellent performance, but they are too large and expensive for portable devices [1,2]. Micro-electro-mechanical-system (MEMS) gyroscopes have, in recent years, been used in low-cost inertial navigation systems (INS) due to their small size and low cost. However, the MEMS gyroscope has a significant error due to the manufacturing technology and structural composition [3,4]. The error of the MEMS gyroscope can be divided into deterministic error and random error. The deterministic error mainly refers to perturbation errors such as zero offsets and the scale factor, which can be corrected by a calibration test [5,6]. Random error refers to the random drift caused by uncertain factors, usually determined by the device's accuracy level [7], with no precise repeatability. Therefore, it is difficult to accurately compensate for random error, which hinders the further improvement of MEMS gyroscope performance.

In MEMS gyroscope error compensation research, the MEMS gyroscope data are generally treated as time-series data. Scholars have proposed methods such as the autoregressive moving average (ARMA) model, the Allan variance (AV), the wavelet threshold (WT), the support vector machine (SVM), and the artificial neural network (ANN), and all of them have achieved excellent results [7–14]. Recently, various variants of the recurrent neural network (RNN), which has strong processing power for time-series data, have been shown to be superior to traditional methods in the research of error compensation in MEMS gyroscopes [15–18].

However, most of the research mentioned above has acquired data by placing the MEMS gyroscope in a static environment. In practical applications of the MEMS gyroscope, it is inevitably that it is affected by random vibration [19]. In random vibration environments, the MEMS gyroscope is interfered with by both internal device noise and external vibration noise [20], which dramatically affects the performance of the MEMS gyroscope. The degradation of performance in vibrating environments is a fatal problem for MEMS gyroscopes [21,22], so it is essential to research error compensation methods in random vibration environments.

Most of the current research on improving the performance of the MEMS gyroscope in random vibration environments is to fix the MEMS gyroscope on a vibration isolation platform [23–25]. However, this kind of method is not universal [26]. There is not much research based on time-series models—the windowed measurement error covariance (WMEC) method has been applied to compensate for the effects of the vibration environments [27], singular spectrum analysis (SSA) was proposed to remove the low-frequency vibration noise perturbations of MEMS accelerometers [28], and the third-order autoregressive (AR) model was used to estimate the Kalman filter to compensate for the MEMS gyroscope’s attitude angle error caused by random vibration [29].

Considering the dramatic perturbation of the MEMS gyroscope in random vibration environments, in this paper, a combined method of a long short-term memory (LSTM) network and Kalman filter is proposed for error compensation, with the Kalman smoother and expectation-maximization (EM) algorithm to dynamically adjust the predicted values of the LSTM network to improve the performance in error compensation. The main contributions of this paper are as follows:

- (1) The combination of LSTM network and Kalman filter is applied to MEMS gyroscope error compensation in random vibration environments;
- (2) The proper input data step and the network topology are explored, and the error compensation performance of the bidirectional LSTM (BiLSTM) network and other recurrent neural network (RNN) variants are compared;
- (3) In designing the Kalman filter, the EM algorithm is used to estimate the parameters. It is compared with the ARMA model, a parameter estimation method commonly used in research of the MEMS gyroscope error compensation problem.

The remainder of this paper is organized as follows: (1) Section 2 introduces the methods, including BiLSTM network, Kalman filter, ARMA-KF model, and EM-KF model, and gives the illustration of this paper proposed method; (2) Section 3 presents the experiment, results, and comparisons; and (3) the remaining sections are the conclusion, appendix, and references.

## 2. Method

### 2.1. Multi-Layer BiLSTM Network and Kalman Filter

The long short-term memory network is a variant of the recurrent neural network used to solve the gradient vanishing or gradient explosion problem of RNNs [30,31]. A detailed description of LSTM units can be found in references [15–18].

The basic LSTM network only considers the historical and current inputs and ignores future inputs [32]. Therefore, the LSTM network can perform the reverse operation, superimpose the forward and reverse information flows, and fully utilize the front and back inputs at the current time to improve the error compensation performance. In addition,

the previous hidden layer’s output is used as the input of the following layer to explore the more in-depth features of the time-series data, thus enhancing the model’s nonlinear fitting ability. The multi-layer BiLSTM network information flow is shown in Figure 1.

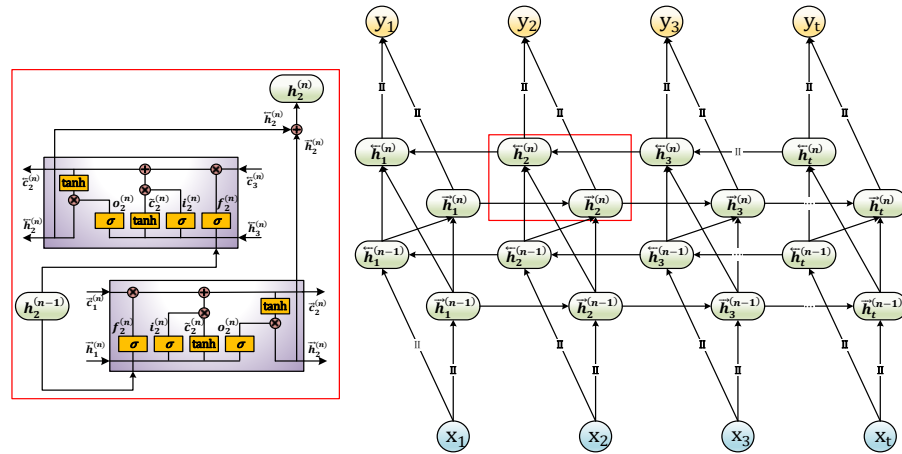


Figure 1. Information flow of multi-layer bidirectional long short-term memory (BiLSTM) network.

The cell state of the Layer  $n$  BiLSTM network at time  $t$  can be presented as:

$$\begin{bmatrix} i_t^{(n)} \\ f_t^{(n)} \\ o_t^{(n)} \\ c_t^{(n)} \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} \begin{bmatrix} W_{i,x}^{(n)} & W_{i,h}^{(n)} \\ W_{f,x}^{(n)} & W_{f,h}^{(n)} \\ W_{o,x}^{(n)} & W_{o,h}^{(n)} \\ W_{c,x}^{(n)} & W_{c,h}^{(n)} \end{bmatrix} \begin{bmatrix} h_t^{(n-1)} \\ h_t^{(n)} \end{bmatrix} \quad (1)$$

where  $h_t^{(n-1)}$  is the hidden state of the Layer  $n - 1$  at time  $t$ . Each hidden state is composed of forward and reverse superposition. The related equations are denoted as follows:

$$h_t^{(n)} = \overset{\rightarrow}{h}_t^{(n)} \oplus \overset{\leftarrow}{h}_t^{(n)} \quad (2)$$

The Kalman filter is an optimal state estimation method that can be applied to dynamic systems with random disturbances. It estimates the system state based on discrete measurement that contain noise [33,34]. Suppose the state–space model is built as:

$$\hat{x}_k = \Phi \hat{x}_{k-1} + \Gamma \omega_{k-1} \quad (3)$$

$$y_k = Hx_k + v_k \quad (4)$$

where  $\Phi$  is the system state transition matrix,  $\Gamma$  is the system noise-driven matrix,  $H$  is the measurement matrix,  $\hat{x}_k$  is the system state vector,  $y_k$  is the measurement vector,  $\omega_k$  is the system noise vector, and  $v_k$  is the measurement noise vector.

The noise of the system models and measurement models are assumed to have normal distribution in the Kalman filter, such that [35]:

$$\omega_k \sim \mathcal{N}(0, Q) \quad (5)$$

$$v_k \sim \mathcal{N}(0, R) \quad (6)$$

where  $Q$  is the covariance matrix of the system models and  $R$  is the covariance matrix of measurement models.

The Kalman filter is composed of two-stage optimization. In the predicted stage, the current system state vector is predicted based on the system state vector at the previous time, such that:

$$\hat{x}_{k/k-1} = \Phi \hat{x}_{k-1} \quad (7)$$

$$P_{k/k-1} = \Phi P_{k-1} \Phi^T + \Gamma Q \Gamma^T \quad (8)$$

where  $\hat{x}_{k/k-1}$  is the predicted value of the system state vector and  $P_{k/k-1}$  is the predicted covariance matrix of the system state vector.

In the updated stage of the Kalman filter, the current system state vector is updated by using the measurement vector, such that:

$$K_k = P_{k/k-1} H^T (H P_{k/k-1} H^T + R)^{-1} \quad (9)$$

$$\hat{x}_k = \hat{x}_{k/k-1} + K_k (y_k - H \hat{x}_{k/k-1}) \quad (10)$$

$$P_k = (I - K_k H) P_{k/k-1} \quad (11)$$

where  $K_k$  is the Kalman filter gain matrix and  $P_k$  is the updated covariance matrix of the system state vector.

## 2.2. Kalman Filter Design with ARMA Model

The autoregressive moving average model is the most widespread model used in time-series analysis, and it is derived and developed on the basis of the linear regression model. The ARMA model can be described as [36]:

$$x_t = \sum_{i=1}^p \varphi_i x_{t-i} - \sum_{j=1}^q \theta_j \varepsilon_{t-j} + \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, \delta_\varepsilon^2) \quad (12)$$

where  $p$  and  $q$  are the order of the ARMA model;  $\varphi_i$  and  $\theta_j$  are coefficients that satisfy stationary and invertible conditions, respectively [37]; and  $\varepsilon_t$  is white noise, which is an uncorrelated random variable with mean zero and constant variance. The model expresses that the measured values of the stochastic process  $\{x_t\}$  at time  $t$  are correlated with the previous  $p$  measurements and the previous  $q$  white noise.

The steps to design a Kalman filter using the ARMA model as follows: (1) test the stationarity and normality of the measurement data, (2) determine the model type according to the autocorrelation function and partial autocorrelation function, (3) determine the order and parameters of the model according to the Akaike information criterion (AIC) [38], and (4) perform adaptive testing of the designed model.

## 2.3. Kalman Filter Design with EM Algorithm

The expectation-maximization algorithm is an iterative method proposed by Shumway and Stoffer to compute maximum likelihood estimates based on incomplete data [39]. It is convergent and can identify parameters and states in the model [40]. Andrieu and Doucet introduced the EM algorithm for parameter estimation for linear state-space models was introduced [41]. The EM algorithm is an iterative numerical algorithm for computing the maximum likelihood estimation (MLE). The linear Gaussian state-space model used for the EM algorithm can be expressed as follows [42]:

$$x_k = \Phi x_{k-1} + \omega_{k-1} \quad (13)$$

$$y_k = H x_k + v_k \quad (14)$$

The conditional probability densities of the state equation and the measurement equation are obtained from Equations (13) and (14), respectively:

$$p(x_k|x_{k-1}) = \exp\left\{-\frac{1}{2}[x_k - \Phi x_{k-1}]^T Q^{-1}[x_k - \Phi x_{k-1}]\right\} (2\pi)^{-\frac{n}{2}} |Q|^{-\frac{1}{2}} \quad (15)$$

$$p(y_k|x_k) = \exp\left\{-\frac{1}{2}[y_k - Hx_k]^T R^{-1}[y_k - Hx_k]\right\} (2\pi)^{-\frac{m}{2}} |R|^{-\frac{1}{2}} \quad (16)$$

It is assumed that the likelihood of the system state data and the evolution of the states is Gaussian, which are defined by the following equations [43]:

$$p(Y, X|\Theta) = p(x_1) \prod_{k=1}^N p(y_k|x_k) \prod_{k=2}^N p(x_k|x_{k-1}) \quad (17)$$

where  $Y$  is the measurement data,  $X$  is the unknown system state data, and  $\Theta$  is the parameter set of linear Gaussian state-space model.  $\Theta$  can be represented as follows:

$$\Theta = \{\Phi, H, Q, R\} \quad (18)$$

By taking the log of the likelihood we arrive at the following formula:

$$\ln p(Y, X|\Theta) = -\frac{1}{2} \sum_{k=2}^N \left\{ \ln|Q| + [x_k - \Phi x_{k-1}]^T Q^{-1} [x_k - \Phi x_{k-1}] \right\} - \frac{1}{2} \sum_{k=1}^N \left\{ \ln|R| + [y_k - Hx_k]^T R^{-1} [y_k - Hx_k] \right\} \quad (19)$$

Depending on the maximum likelihood method, the linear Gaussian state-space model can be identified through an EM algorithm [42]. The algorithm alternates between two steps—the E-step (expectation) and the M-step (maximization) [44]. In general, the likelihood density function based on the measurement data, denoted by  $p(\Theta|Y)$ , is called the posterior distribution of the measurement. The EM algorithm aims to compute the maximum likelihood estimation of  $p(\Theta|Y)$ .  $\Theta_i$  is denoted as the estimate of the likelihood function at the beginning of the  $i$ th iteration.

In the E-step, the expectation for the conditional distribution of  $\ln p(Y, X|\Theta)$  concerning  $X$ , is calculated such that:

$$\Omega(\Theta|\Theta_i, Y) E_X \{ \ln p(\Theta|Y, X) | \Theta_i, Y \} = \int [\ln p(\Theta|Y, X)] (X|\Theta_i, Y) dX \quad (20)$$

In the M-step,  $\Omega(\Theta|\Theta_i, Y)$  is maximized to find  $\Theta_{i+1}$  such that:

$$\Omega(\Theta_{i+1}|\Theta_i, Y) = \operatorname{argmax}_{\Theta} [\Omega(\Theta|\Theta_i, Y)] \quad (21)$$

The E-step and the M-step are iterated until,

$$\|L(\Theta_{i+1}) - L(\Theta_i)\| < \tau \quad (22)$$

where  $\tau$  is the predefined threshold. Equation (22) means that it has satisfied the convergence criterion. The specific process of designing a Kalman filter using the EM algorithm is given as follows:

The value of  $\Omega(\Theta|\Theta_i, Y)$  is determined by the following [45]:

$$E_X(x_k|Y) = \hat{x}_{k|N} \quad (23)$$

$$E_X(x_k x_{k-1}^T | Y) = P_{k,k-1|N} + \hat{x}_{k|N} \hat{x}_{k-1|N}^T \quad (24)$$

$$E_X(x_k x_k^T | Y) = P_{k|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T \quad (25)$$

where  $\hat{x}_{k|N}$  is the smoothed value of the system state vector and  $P_{k|N}$  is the smoothed covariance matrix of the system state vector.  $P_{k,k-1|N}$  is initialized by:

$$P_{k,k-1|k} = (I - K_k H) \Phi P_{k-1} \quad (26)$$

$$P_{k,k-1|N} = P_{k,k-1|k} + [P_{k|N} - P_k] P_{k|k}^{-1} P_{k,k-1|k} \quad (27)$$

$\hat{x}_{k|N}$  and  $P_{k|N}$  can be obtained by smoothing the outputs of Kalman filter using backward-pass methods such as the Rauch–Tung–Striebel (RTS) smoother [46]. This method is summarized in the following equations:

$$J_{k-1} = P_{k-1} \Phi^T P_{k|k-1}^{-1} \quad (28)$$

$$\hat{x}_{k-1|N} = \hat{x}_{k-1} + J_{k-1} (\hat{x}_{k|N} - \hat{x}_{k/k-1}) \quad (29)$$

$$P_{k-1|N} = P_{k-1} - J_{k-1} (P_{k|N} - P_{k|k-1}) J_{k-1}^T \quad (30)$$

Then, the model parameters are re-estimated by maximizing the  $\Omega(\Theta|\Theta_i, Y)$  over  $\Theta$  using partial derivatives of  $\Omega(\Theta|\Theta_i, Y)$  and setting them to zero. Solving these equations yields the updated parameters (in the  $i$ th iteration) as follows:

$$\frac{\partial L(\Theta)}{\partial \Phi} = - \sum_{k=2}^N Q^{-1} (P_{k,k-1|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T) + \sum_{k=2}^N Q^{-1} \Phi (P_{k-1|N} + \hat{x}_{k-1|N} \hat{x}_{k-1|N}^T) = 0 \quad (31)$$

$$\Phi_{i+1} = \left( \sum_{k=2}^N P_{k,k-1|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T \right) \left( \sum_{k=2}^N P_{k-1|N} + \hat{x}_{k-1|N} \hat{x}_{k-1|N}^T \right)^{-1} \quad (32)$$

$$\frac{\partial L(\Theta)}{\partial H} = - \sum_{k=1}^N R^{-1} y_k \hat{x}_{k|N}^T + \sum_{k=1}^N R^{-1} H (P_{k|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T) = 0 \quad (33)$$

$$H_{i+1} = \left( \sum_{k=1}^N y_k \hat{x}_{k|N}^T \right) \left[ \sum_{k=1}^N (P_{k|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T) \right]^{-1} \quad (34)$$

$$\frac{\partial L(\Theta)}{\partial Q^{-1}} = \frac{N}{2} Q - \frac{1}{2} \sum_{k=2}^N (P_{k|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T) + \Phi \left[ \frac{1}{N} \sum_{k=2}^N (P_{k,k-1|N} + \hat{x}_{k|N} \hat{x}_{k-1|N}^T) \right] = 0 \quad (35)$$

$$Q_{i+1} = \frac{1}{N} \left( \sum_{k=1}^N (P_{k|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T) - \Phi_{i+1} \sum_{k=2}^N (P_{k,k-1|N} + \hat{x}_{k|N} \hat{x}_{k-1|N}^T) \right) \quad (36)$$

$$\frac{\partial L(\Theta)}{\partial R^{-1}} = \frac{N+1}{2} R - \sum_{k=1}^N \left( \frac{1}{2} y_k y_k^T - H \hat{x}_{k|N} y_k^T + \frac{1}{2} H (P_{k|N} + \hat{x}_{k|N} \hat{x}_{k|N}^T) H^T \right) = 0 \quad (37)$$

$$R_{i+1} = \frac{1}{N+1} \left( \sum_{k=1}^N y_k y_k^T \right) - H_{i+1} \left( \frac{1}{N+1} \sum_{k=1}^N y_k \hat{x}_{k|N} \right)^T \quad (38)$$

In this paper, based on the EM algorithm, the proposed LSTM and Kalman filter combination method is illustrated in Figure 2.

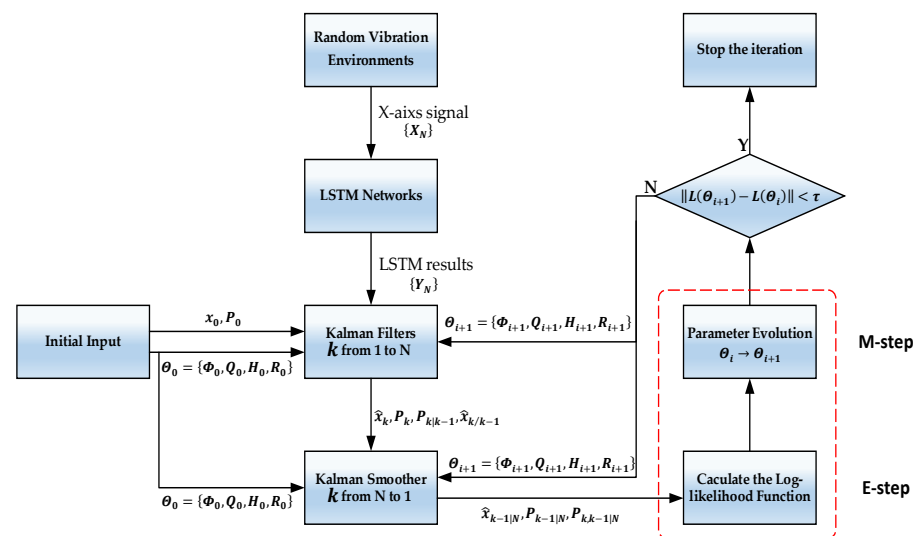


Figure 2. An illustration of this paper's proposed method.

### 3. Experiments and Results

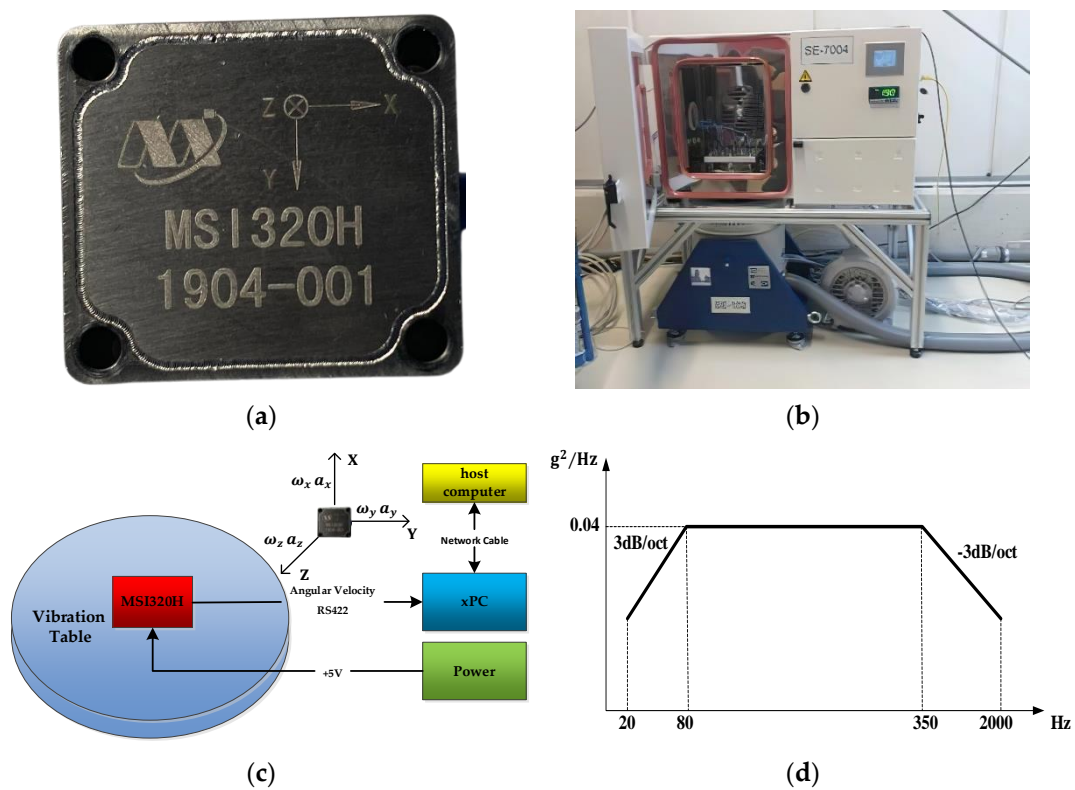
In this section, the designed experiments and the analysis of the results are presented to verify the effectiveness of the proposed method.

#### 3.1. Data Acquisition

The MSI320H MEMS Inertial Measurement Unit (IMU) was employed for experiments. This consists of a three-axis MEMS gyroscope and a three-axis MEMS accelerometer. The real picture and the gyroscope specifications of MSI320H are shown in Figure 3a and Table 1, respectively. The MSI320H was fixed on the vibration table. A picture of the vibration table is shown in Figure 3b. The data acquisition procedure of the MSI320H is shown in Figure 3c. Data from the MSI320H was sent to the xPC via the RS422 communication interface with a Baud of 921,600 bps. The xPC decoded the gyroscope data and sent it to the host computer via the network cable. The MSI320H was preheated at room temperature with power for 20 minutes. Then, linear vibration experiments were performed. The vibration direction of the vibration table is the  $y$ -axis of the gyroscope, and the power spectral density (PSD) of the linear random vibration loads is shown in Figure 3d.

Table 1. Specifications of MSI320H gyroscope.

GYRO	Input range	$\pm 1800^\circ/\text{s}$
	Bias instability (Allan variance)	$36^\circ/\text{h}$
	Angular random walk (Allan variance)	$0.4^\circ/\sqrt{\text{h}}$
	Bandwidth ( $-3$ dB)	$\geq 220$ Hz
GENERAL	Sample rate	100~1000 Hz
	Weight	$\leq 25$ g
	Supply voltage	$5.0 \pm 0.5$ V
	RS422 transmission bit rate	921600 bps
	Mechanical shock, any direction	$\geq 20,000$ g

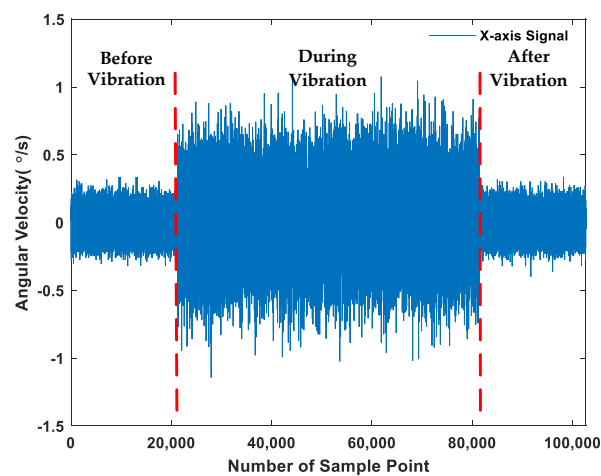


**Figure 3.** Experimental system. (a) MSI3200H inertial measurement unit, (b) vibration table, (c) data acquisition procedure, and (d) power spectral density of linear random vibration loads.

As illustrated in Figure 3d, the acceleration of the applied vibration loads can be expressed as follows:

$$a_v(t) = \zeta_v \sin(\omega_v t), \quad \omega_v \in [20 \cdot 2\pi, 2000 \cdot 2\pi] \quad (39)$$

where  $\zeta_v = 6$  g. The sample rate was set to 200 Hz, and approximately 60,000 data were acquired in the random vibration environment. The variation of gyroscope  $x$ -axis signal affected by random vibration is shown in Figure 4, and the error of the gyroscope increased significantly in random vibration environments.



**Figure 4.** The variation of gyroscope  $x$ -axis signal.



### 3.2. Comparison of BiLSTM and Other RNN Variants

The outliers of the acquired data in random vibration environments were eliminated using the Puata criterion [47]. Because the linear random vibration test's direction was the  $y$ -axis of the gyroscope, the  $x$ -axis and  $z$ -axis directions were in a random vibration environment. For the consideration of model generality, we took the last 80% of the processed  $x$ -axis data as the training set and the first 20% of the  $x$ -axis and  $z$ -axis data were used as the testing set.

If the hidden layer structure of the designed network is too simple, it is not easy to characterize the time-series model of gyroscope data. Conversely, it increases the complexity of the network, reduces the learning speed of the network, and tends to fall into local minima during the learning process. With the above considerations, in this paper, the proposed BiLSTM network is shown in Figure 5. The dense layer transformed the high-dimensional stacked sequence of the hidden layer into an output sequence with the same shape as the input sequence. Moreover, considering a large number of network parameters, dropout was set in the dense layer to prevent the overfitting phenomenon [48].

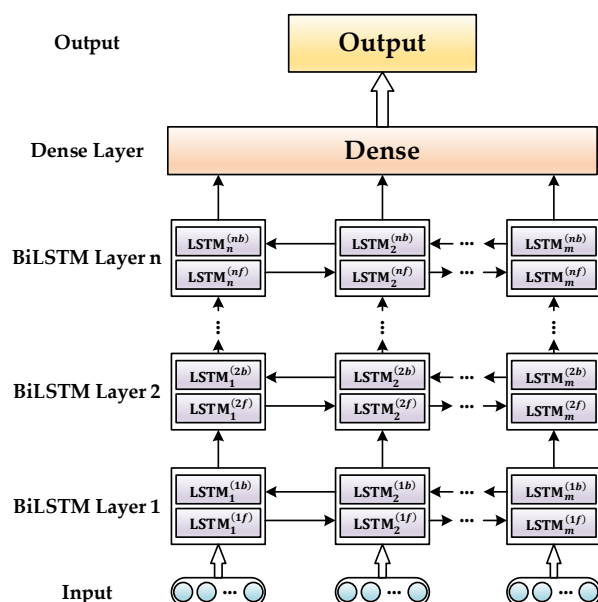


Figure 5. Multi-layer BiLSTM network.

The adaptive moment estimation (Adam) optimization algorithm was used to update the network parameters [49]. The Adam algorithm uses default parameters. The activation function of the Dense layer is rectified linear unit (ReLU). The specifications used for network training are illustrated in Table 2. Moreover, the root mean square error (RMSE) was used as the loss function. The network training was performed on Tensorflow 2.0.0 and Keras 2.3.1 over the Ubuntu 16.04-LTS-x86 64 operating system. The heterogeneous computing platform was equipped with Intel Xeon E5-1620 and GeForce RTX-2080Ti GPUs.

Table 2. Specifications used for network training.

The output dimension of dense layer	1
Activation function of dense layer	ReLU
Dropout rate	0.5
Batch size	256
Training epoch	50
Learning rate	0.001

In order to verify the performance of BiLSTM for gyroscope error compensation in random vibration environments, proper values for the input data step size, the number of hidden units, and the number of hidden layers was first explored using the *x*-axis testing set. Subsequently, training was performed using the identified values. The BiLSTM network results were compared with the LSTM network, gated recurrent unit (GRU) network, and bidirectional GRU (BiGRU) network using the *x*-axis and *z*-axis testing sets, respectively.

As shown in Tables 3–5, when the input data step size and the number of hidden layers are more extensive, the training time per epoch will be longer. So we needed to make a trade-off between the results and the computational performance. According to the results, the best results were obtained when taking the input data step of 20, the number of hidden units of 128, and the number of hidden layers of 10. Although it does not indicate that this is the optimal parameter for the network, it will be the proper value to be obtained considering the computational resources.

**Table 3.** Performance with varying values of the input data step.

Number of Hidden Layers	Number of Hidden Units	Input Data Step	STD (°/s)	Time/Epoch
10	64	5	0.1551	23 s
10	64	10	0.1481	38 s
10	64	15	0.1483	60 s
10	64	20	0.1346	82 s
10	64	25	0.1368	98 s
10	64	30	0.1501	115 s

**Table 4.** Performance with varying values of the number of hidden units.

Number of Hidden Layers	Number of Hidden Units	Input Data Step	STD (°/s)	Time/Epoch
10	8	20	0.1504	82 s
10	16	20	0.1459	82 s
10	32	20	0.1559	81 s
10	64	20	0.1346	82 s
10	128	20	0.1326	81 s
10	256	20	0.1468	88 s

**Table 5.** Performance with varying values of the number of hidden layers.

Number of Hidden Layers	Number of Hidden Units	Input Data Step	STD (°/s)	Time/Epoch
1	128	20	0.1493	10 s
2	128	20	0.1513	19 s
3	128	20	0.1597	27 s
4	128	20	0.1557	34 s
5	128	20	0.1658	42 s
6	128	20	0.1505	50 s
7	128	20	0.1470	58 s
8	128	20	0.1459	66 s
9	128	20	0.1542	75 s
10	128	20	0.1326	81 s
11	128	20	0.1405	90 s
12	128	20	0.1393	98 s

The results are shown in Figures 6 and 7 and Tables 6 and 7. Figure 6 shows the training loss within 50 epochs, and convergence was achieved for all networks. Tables 6 and 7 show that the standard deviations of the BiLSTM network results for the *x*-axis and *z*-axis were

reduced by 46.81% and 43.63%, respectively, compared to the raw data, proving that the BiLSTM network is feasible for the application in the research of MEMS gyroscope error compensation. Furthermore, compared with the results of the LSTM network, the BiGRU network, and the GRU network, the standard deviation values of BiLSTM results in the  $x$ -axis were reduced by 14.06%, 11.66%, and 17.33%, respectively, and the standard deviation values of BiLSTM results in the  $z$ -axis were reduced by 12.71%, 10.04%, and 14.04%, respectively. This indicates that the error compensation performance of the BiLSTM network is better than these three networks.

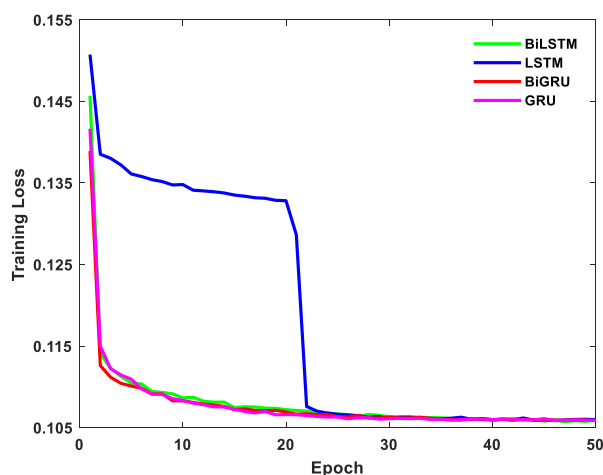


Figure 6. Training loss of BiLSTM, LSTM, BiGRU, and GRU.

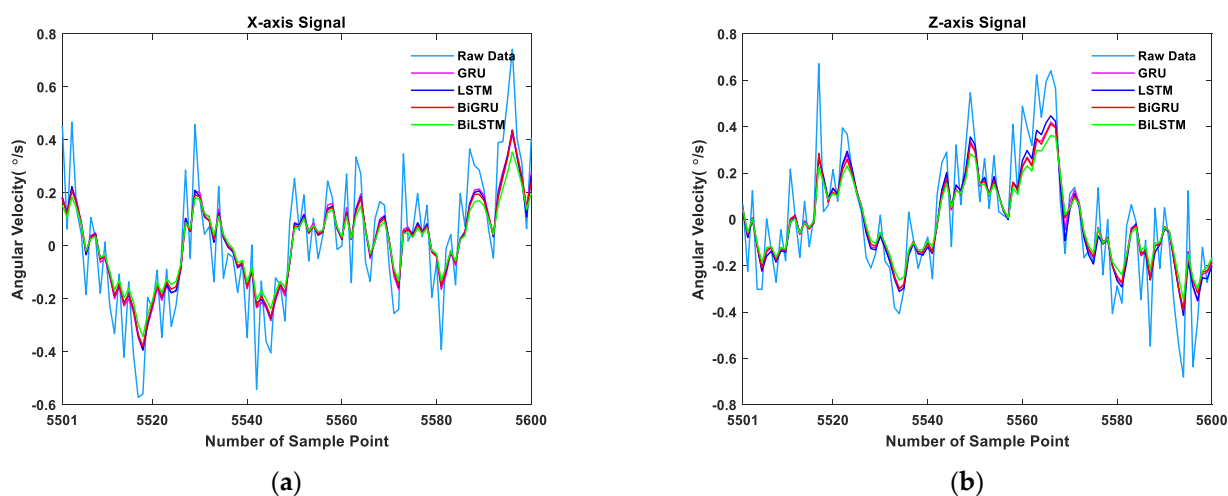


Figure 7. Comparison of error compensation performance of BiLSTM, LSTM, BiGRU, and GRU. (a) Part of the  $x$ -axis data zoomed-in and (b) part of the  $z$ -axis data zoomed-in.

Table 6. Comparison of raw data, BiLSTM, LSTM, BiGRU, and GRU standard deviation values for  $x$ -axis data.

$x$ -axis	STD ( $^{\circ}$ /s)	Percentage
Raw data	0.2493	—
BiLSTM	0.1326	53.19%
LSTM	0.1543	61.89%
BiGRU	0.1501	60.21%
GRU	0.1604	64.34%

**Table 7.** Comparison of raw data, BiLSTM, LSTM, BiGRU, and GRU standard deviation values for z-axis data.

z-axis	STD (°/s)	Percentage
Raw data	0.2400	—
BiLSTM	0.1353	56.38%
LSTM	0.1550	64.58%
BiGRU	0.1504	62.67%
GRU	0.1574	65.58%

### 3.3. Comparison of LSTM-EM-KF and LSTM-ARMA-KF

In this section, the raw data and BiLSTM network results of the  $x$ -axis and  $z$ -axis are used as the measurement, respectively. The Kalman filter parameters are estimated by the ARMA model and EM algorithm, and the filter results are compared.

#### 3.3.1. Estimating Kalman Filter Parameters Using the ARMA Model

When modeling time-series data using the ARMA model, the time-series data must meet stationarity and normality requirements. Therefore, a polynomial fitting method was used to eliminate the trend term before modeling. In this paper, the stationarity was tested using the run test, and the normality was tested by calculating the skewness,  $\zeta$ , and the kurtosis,  $\nu$ .

According to the test, after eliminating the trend term, the raw data and BiLSTM network results of the  $x$ -axis and  $z$ -axis met the stationarity and normality requirements. The test process is shown in Appendix A. Moreover, as illustrated in Figure A1, the autocorrelation function and partial autocorrelation function diagrams exhibit trailing properties, and all models can be identified as an ARMA ( $p, q$ ) model.

The next step is to determine the order of the model. If the order is increased, the identified model will be more realistic, but the computational difficulty will also increase as the order increases. Therefore, the maximum order was set to 3, which means the maximum value of  $p$  and  $q$  was set to 3. Furthermore, the Akaike information criterion (AIC) was used for determining model order. Determining the model order process and the Durbin–Watson test results are shown in Appendix B.

For  $x$ -axis raw data, the model identified is identified as ARMA(3,3):

$$x_n = -0.3126x_{n-1} + 0.8168x_{n-2} + 0.1520x_{n-3} + \varepsilon_n + 0.6885\varepsilon_{n-1} - 0.3241\varepsilon_{n-2} - 0.0366\varepsilon_{n-3} \quad (40)$$

For  $x$ -axis BiLSTM network results, the model is identified as ARMA(3,3):

$$x_n = 0.8505x_{n-1} + 0.8891x_{n-2} - 0.7745x_{n-3} + \varepsilon_n + 0.1354\varepsilon_{n-1} - 0.8476\varepsilon_{n-2} - 0.0815\varepsilon_{n-3} \quad (41)$$

For  $z$ -axis raw data, the model identified is identified as ARMA(1,3):

$$x_n = 0.8593x_{n-1} + \varepsilon_n - 0.51915\varepsilon_{n-1} + 0.0328\varepsilon_{n-2} - 0.0236\varepsilon_{n-3} \quad (42)$$

For  $z$ -axis BiLSTM network results, the model identified is identified as ARMA(3,3):

$$x_n = 2.0000x_{n-1} - 1.2268x_{n-2} + 0.2031x_{n-3} + \varepsilon_n - 1.0446\varepsilon_{n-1} - 0.0809\varepsilon_{n-2} - 0.1086\varepsilon_{n-3} \quad (43)$$

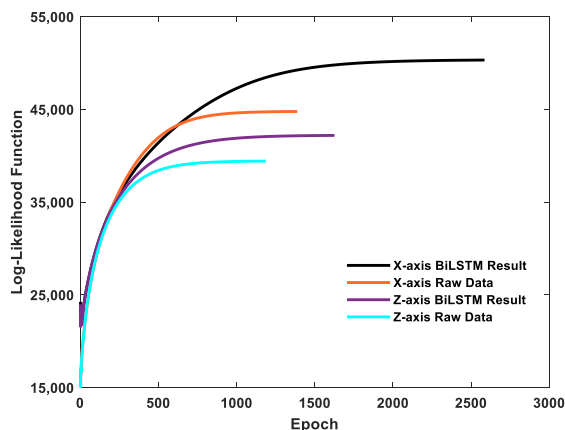
where  $x_n$  is the output of the ARMA model,  $\varepsilon_n$  is the driving white noise (with mean, 0, and variance,  $\hat{\delta}_\varepsilon^2$ ). The Kalman filter parameters are presented in Table 8. The value of  $R$  is the variance of the measurement. The initial value of the Kalman filter is set as follows:  $x_1 = [0; 0; 0; 0]$ , and  $P_1$  is the fourth-order identity matrix.

**Table 8.** Kalman filter parameters of all measurements.

	$\Phi$	$\Gamma$	$H$	$Q$	$R$
x-axis raw data	$\begin{bmatrix} -0.3126 & 0.8168 & 0.1520 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0.6885 & -0.3241 & -0.0366 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$[ 1 \ 0 \ 0 \ 0 ]$	$\begin{bmatrix} 0.0370 & 0 & 0 & 0 \\ 0 & 0.0370 & 0 & 0 \\ 0 & 0 & 0.0370 & 0 \\ 0 & 0 & 0 & 0.0370 \end{bmatrix}$	0.0604
x-axis BiLSTM	$\begin{bmatrix} 0.8505 & 0.8891 & -0.7745 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0.1354 & -0.8476 & -0.0815 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$[ 1 \ 0 \ 0 \ 0 ]$	$\begin{bmatrix} 0.0036 & 0 & 0 & 0 \\ 0 & 0.0036 & 0 & 0 \\ 0 & 0 & 0.0036 & 0 \\ 0 & 0 & 0 & 0.0036 \end{bmatrix}$	0.0175
z-axis raw data	$\begin{bmatrix} 0.8593 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & -0.5191 & 0.0328 & 0.0236 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$[ 1 \ 0 \ 0 \ 0 ]$	$\begin{bmatrix} 0.0405 & 0 & 0 & 0 \\ 0 & 0.0405 & 0 & 0 \\ 0 & 0 & 0.0405 & 0 \\ 0 & 0 & 0 & 0.0405 \end{bmatrix}$	0.0596
z-axis BiLSTM	$\begin{bmatrix} 2.0000 & -1.2268 & 0.2031 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & -1.0446 & 0.0809 & 0.1086 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$	$[ 1 \ 0 \ 0 \ 0 ]$	$\begin{bmatrix} 0.0043 & 0 & 0 & 0 \\ 0 & 0.0043 & 0 & 0 \\ 0 & 0 & 0.0043 & 0 \\ 0 & 0 & 0 & 0.0043 \end{bmatrix}$	0.0183

### 3.3.2. Estimating Kalman Filter Parameters Using the EM Algorithm

When using the EM algorithm to estimate the Kalman filter parameters, only the iteration convergence conditions and initial parameters need to be set. The M-step convergence constant  $\tau$  in Equation (22) was set to 0.1. The Kalman filter’s initial values were set to  $x_1 = 0$  and  $P_1 = 1$ , and the Kalman filter’s initial parameters were set to  $\Phi_1 = 1$ ,  $H_1 = 1$ ,  $Q_1 = 1$ , and  $R_1 = 1$ . The change of the log-likelihood function during the iteration of the EM algorithm is shown in Figure 8. The parameter estimation results are presented in Table 9.



**Figure 8.** Log-likelihood value change of the iteration epoch.

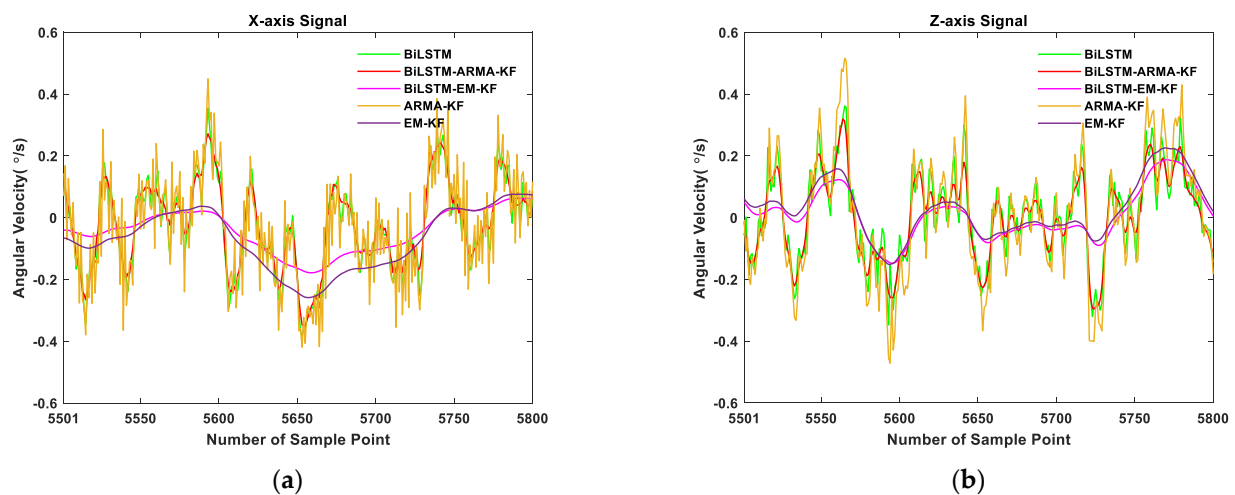
**Table 9.** The parameter estimation results.

	$\Phi$	$H$	$Q$	$R$
x-axis raw data	0.9723	0.1350	0.0016	0.1181
x-axis BiLSTM	0.9738	0.3422	0.0016	0.2852
z-axis raw data	0.9471	0.1327	0.0065	0.1155
z-axis BiLSTM	0.9530	0.2983	0.0044	0.2430

### 3.3.3. Kalman Filtering Results

The results are illustrated in Tables 10 and 11. For the x-axis data, the standard deviation of the BiLSTM-EM-KF results was reduced by 51.58% and 31.92% compared to the BiLSTM network and EM-KF, respectively. For the z-axis data, the standard deviation of the BiLSTM-EM-KF results was reduced by 29.19% and 12.75% compared to the BiLSTM network and EM-KF, respectively. Therefore, the combined method proposed in this paper can be demonstrated to improve the gyroscope error compensation performance of the

BiLSTM network and EM algorithm. Moreover, compared with BiLSTM-ARMA-KF results, the standard deviation of the BiLSTM-EM-KF was reduced by 46.54% and 22.30% in  $x$ -axis and  $z$ -axis, respectively. It indicates the proposed method's superior performance to that of BiLSTM-ARMA-KF. Furthermore, according to Figure 9, the curves of BiLSTM-EM-KF results are smoother, which proves that the proposed combined method is effective.



**Figure 9.** Comparison of error compensation performance of BiLSTM-EM-KF and BiLSTM-ARMA-KF. (a) Part of the  $x$ -axis data zoomed-in and (b) part of the  $z$ -axis data zoomed-in.

**Table 10.** Comparison of raw data, BiLSTM, ARMA-KF, EM-KF, BiLSTM-ARMA-KF, and BiLSTM-EM-KF standard deviation values for  $x$ -axis data.

$x$ -axis	STD (°/s)	Percentage
Raw data	0.2493	—
BiLSTM	0.1326	53.19%
ARMA-KF	0.1618	64.90%
EM-KF	0.0943	37.83%
BiLSTM-ARMA-KF	0.1201	48.17%
BiLSTM-EM-KF	0.0642	25.75%

**Table 11.** Comparison of raw data, BiLSTM, ARMA-KF, EM-KF, BiLSTM-ARMA-KF, and BiLSTM-EM-KF standard deviation values for  $z$ -axis data.

$z$ -axis	STD (°/s)	Percentage
Raw data	0.2400	—
BiLSTM	0.1353	56.38%
ARMA-KF	0.1853	77.21%
EM-KF	0.1098	45.75%
BiLSTM-ARMA-KF	0.1233	51.38%
BiLSTM-EM-KF	0.0958	39.92%

#### 4. Conclusions

In this paper, a combined method of an LSTM network and Kalman filter is proposed for MEMS gyroscope error compensation in random vibration environments. Through the results, the following conclusions were obtained:

- (1) After exploring proper input data step size and network topology, the network was trained, and the test results showed that the BiLSTM network outperformed the LSTM network, the GRU network, and the BiGRU network in gyroscope error compensation;

- (2) Combining the BiLSTM network with the EM-KF method can improve their gyroscopic error compensation performance;
- (3) In the classical gyroscope error compensation method, the ARMA-KF method, tedious data testing and model checking are required. In contrast, the EM-KF method only needs to set the initial parameters and the convergence value, which is much easier to apply. Moreover, the ARMA-KF method parameters cannot be updated through the filtering process, which means that satisfactory results cannot be obtained if the parameters are not defined correctly before the filtering process. From the filtering results, compared with BiLSTM-ARMA-KF, the standard deviation of the BiLSTM-EM-KF results were 46.54% and 22.30% lower, in  $x$ -axis and  $z$ -axis, and the output curve was smoother, which proves the effectiveness of the proposed method in this paper.

Future work should include conducting dynamic field experiments to obtain MEMS gyroscope outputs, as well as combining neural networks with more state-of-the-art Kalman filter methods for MEMS gyroscope error compensation and using fiber optic gyroscopes or laser gyroscopes as benchmarks for comparison.

**Author Contributions:** Conceptualization, C.Z. and H.C.; methodology, C.Z., S.C. and W.X.; software, C.Z. and Y.Y.; validation, S.C. and H.S.; formal analysis, C.Z. and S.C.; investigation, H.C. and W.X.; resources, S.C. and W.X.; data curation, C.Z.; writing—original draft preparation, C.Z.; writing—review and editing, C.Z.; visualization, C.Z.; supervision, S.C.; project administration, H.S.; funding acquisition, W.X. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Natural Science Foundation of Jilin Province, China, grant number 20200201170JC.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Appendix A. Stationarity and Normality Tests

For stationarity, the most common test method is the run test. It was divided equally into 20 groups  $\{X_1, X_2, \dots, X_{20}\}$ , and the mean square value of each group was  $\sigma_i$  where  $i \in [1, 20]$ . The mean of  $\{\sigma_i\}$  was  $\sigma_{mean}$ , and the difference between each  $\sigma_i$  and  $\sigma_{mean}$  formed  $\{\sigma_{ai}\}$ :

$$\sigma_{ai} = \sigma_i - \sigma_{mean} \quad (A1)$$

The total number of positive values in  $\{\sigma_{ai}\}$  was recorded as  $n_1$ , and the total number of negative values in  $\{\sigma_{ai}\}$  was recorded as  $n_2$ . The number of positive and negative alternations in order and plus 1 more was the Run  $r$ . According to the run test table, whether  $r$  was within the confidence interval was determined at the significance level  $\alpha = 0.05$ .

After eliminating the trend term, the raw data and BiLSTM network results of the  $x$ -axis and  $z$ -axis met the stationarity requirements. The test process is presented in Tables 1–4.

**Table 1.** Run test process of  $x$ -axis raw data.

$\sigma_{mean} = 0.0604, n_1 = 8, n_2 = 12, r = 14, \text{Significance Level } \alpha = 0.05, \text{Confidence Interval } [6, 16].$										
	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_7$	$X_8$	$X_9$	$X_{10}$
$\sigma_i$	0.0615	0.0583	0.0516	0.0621	0.0664	0.0572	0.0640	0.0545	0.0571	0.0628
$\sigma_{ai}$	0.0011	−0.0021	−0.0088	0.0017	0.0060	−0.0032	0.0036	−0.0059	−0.0033	0.0024
$r$	1	2		3		4	5	6		7
	$X_{11}$	$X_{12}$	$X_{13}$	$X_{14}$	$X_{15}$	$X_{16}$	$X_{17}$	$X_{18}$	$X_{19}$	$X_{20}$
$\sigma_i$	0.0601	0.0602	0.0565	0.0743	0.0585	0.0655	0.0532	0.0693	0.0588	0.0557
$\sigma_{ai}$	−0.0003	−0.0002	−0.0039	0.0139	−0.0019	0.0051	−0.0072	0.0089	−0.0016	−0.0047
$r$	8			9	10	11	12	13	14	

**Table 2.** Run test process of x-axis BiLSTM network results.

$\sigma_{mean} = 0.0175, n_1 = 11, n_2 = 9, r = 14, \text{Significance Level } \alpha = 0.05, \text{Confidence Interval } [6, 16].$										
	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_7$	$X_8$	$X_9$	$X_{10}$
$\sigma_i$	0.0189	0.0177	0.0144	0.0192	0.0198	0.0163	0.0191	0.0154	0.0159	0.0192
$\sigma_{ai}$	0.0014	0.0002	−0.0031	0.0017	0.0023	−0.0012	0.0016	−0.0022	−0.0016	0.0017
$r$	1		2	3		4	5	6		7
	$X_{11}$	$X_{12}$	$X_{13}$	$X_{14}$	$X_{15}$	$X_{16}$	$X_{17}$	$X_{18}$	$X_{19}$	$X_{20}$
$\sigma_i$	0.0176	0.0178	0.0159	0.0224	0.0149	0.0191	0.0144	0.0205	0.0166	0.0150
$\sigma_{ai}$	0.0001	0.0003	−0.0016	0.0049	−0.0026	0.0016	−0.0031	0.0030	−0.0009	−0.0025
$r$			8	9	10	11	12	13	14	

**Table 3.** Run test process of z-axis raw data.

$\sigma_{mean} = 0.0596, n_1 = 10, n_2 = 10, r = 10, \text{Significance Level } \alpha = 0.05, \text{Confidence Interval } [6, 16].$										
	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_7$	$X_8$	$X_9$	$X_{10}$
$\sigma_i$	0.0673	0.0565	0.0639	0.0694	0.0611	0.0570	0.0489	0.0533	0.0556	0.0539
$\sigma_{ai}$	0.0077	−0.0031	0.0043	0.0098	0.0015	−0.0026	−0.0107	−0.0063	−0.0040	−0.0057
$r$	1	2	3			4				
	$X_{11}$	$X_{12}$	$X_{13}$	$X_{14}$	$X_{15}$	$X_{16}$	$X_{17}$	$X_{18}$	$X_{19}$	$X_{20}$
$\sigma_i$	0.0577	0.0671	0.0642	0.0645	0.0585	0.0619	0.0516	0.0622	0.0620	0.0560
$\sigma_{ai}$	−0.0019	0.0075	0.0046	0.0049	−0.0011	0.0023	−0.0080	0.0026	0.0024	−0.0036
$r$		5			6	7	8	9		10

**Table 4.** Run test process of z-axis BiLSTM network results.

$\sigma_{mean} = 0.0183, n_1 = 9, n_2 = 11, r = 10, \text{Significance Level } \alpha = 0.05, \text{Confidence Interval } [6, 16].$										
	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_7$	$X_8$	$X_9$	$X_{10}$
$\sigma_i$	0.0216	0.0159	0.0200	0.0227	0.0180	0.0168	0.0139	0.0160	0.0158	0.0158
$\sigma_{ai}$	0.0033	−0.0024	0.0017	0.0044	−0.0003	−0.0015	−0.0044	−0.0023	−0.0025	−0.0025
$r$	1	2	3		4					
	$X_{11}$	$X_{12}$	$X_{13}$	$X_{14}$	$X_{15}$	$X_{16}$	$X_{17}$	$X_{18}$	$X_{19}$	$X_{20}$
$\sigma_i$	0.0173	0.0217	0.0213	0.0208	0.0175	0.0197	0.0154	0.0199	0.0191	0.0162
$\sigma_{ai}$	−0.0010	0.0034	0.0030	0.0025	−0.0008	0.0014	−0.0029	0.0016	0.0008	−0.0021
$r$		5			6	7	8	9		10

For normality, the most common test method is calculating skewness,  $\zeta$ , and kurtosis,  $v$ . When the calculated  $\zeta$  is close to 0, and  $v$  is close to 3, the data are considered to satisfy normality. For the valuation of  $\zeta$  and  $v$ :

$$\zeta = E \left[ \left( \frac{x_N - e_x}{\sigma_x} \right)^3 \right] = \frac{E \left[ (x_N - e_x)^3 \right]}{\left\{ E \left[ (x_N - e_x)^2 \right] \right\}^{\frac{3}{2}}} = \frac{\frac{1}{N} \sum_{n=1}^N (x_N - e_x)^3}{\left[ \frac{1}{N} \sum_{n=1}^N (x_N - e_x)^2 \right]^{\frac{3}{2}}} \quad (\text{A2})$$

$$v = E \left[ \left( \frac{x_N - e_x}{\sigma_x} \right)^4 \right] = \frac{E \left[ (x_N - e_x)^4 \right]}{\left\{ E \left[ (x_N - e_x)^2 \right] \right\}^2} = \frac{\frac{1}{N} \sum_{n=1}^N (x_N - e_x)^4}{\left[ \frac{1}{N} \sum_{n=1}^N (x_N - e_x)^2 \right]^2} \quad (\text{A3})$$

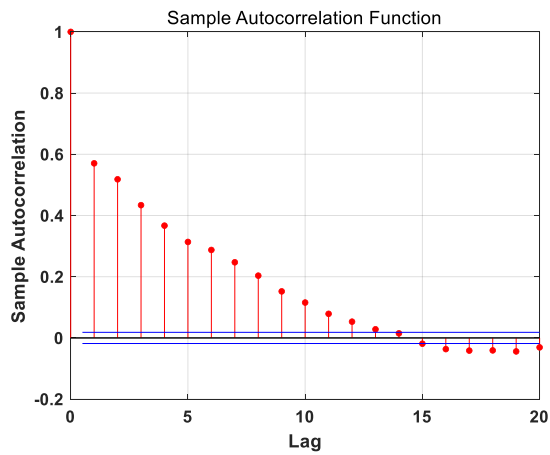
where  $e_x$  and  $\sigma_x$  are the mean and standard deviation of the data, respectively.

After eliminating the trend term, the raw data and BiLSTM network results of the x-axis and z-axis met the normality requirements. The results are presented in Table 5.

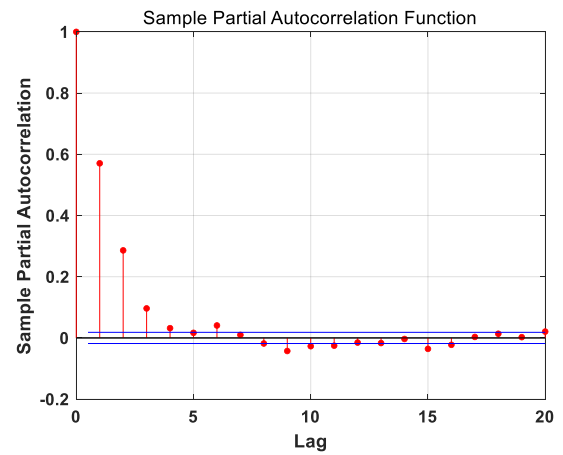


**Table 5.** The normality test results.

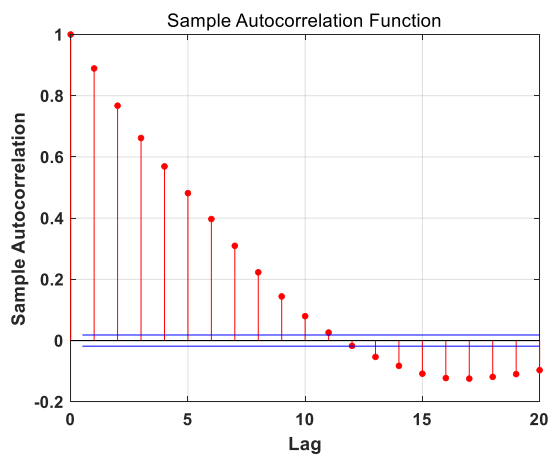
	<i>x</i> -axis Raw Data	<i>x</i> -axis BiLSTM Network Results	<i>z</i> -axis Raw Data	<i>z</i> -axis BiLSTM Network Results
Skewness $\xi$	2.8329	2.7283	2.7294	2.8156
Kurtosis $\nu$	0.0177	-0.1077	-0.0631	-0.0170

**B. Determine the ARMA Model Order**

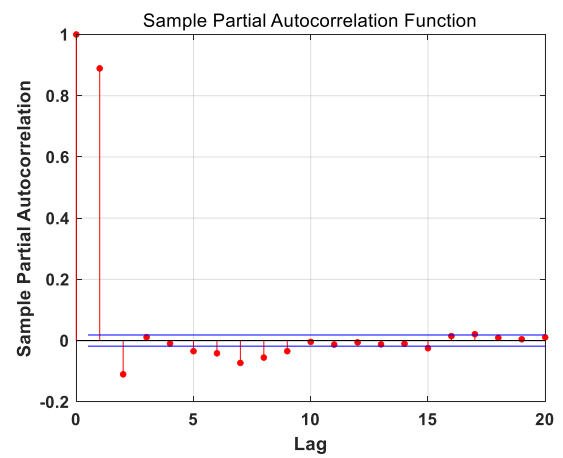
(a)



(b)

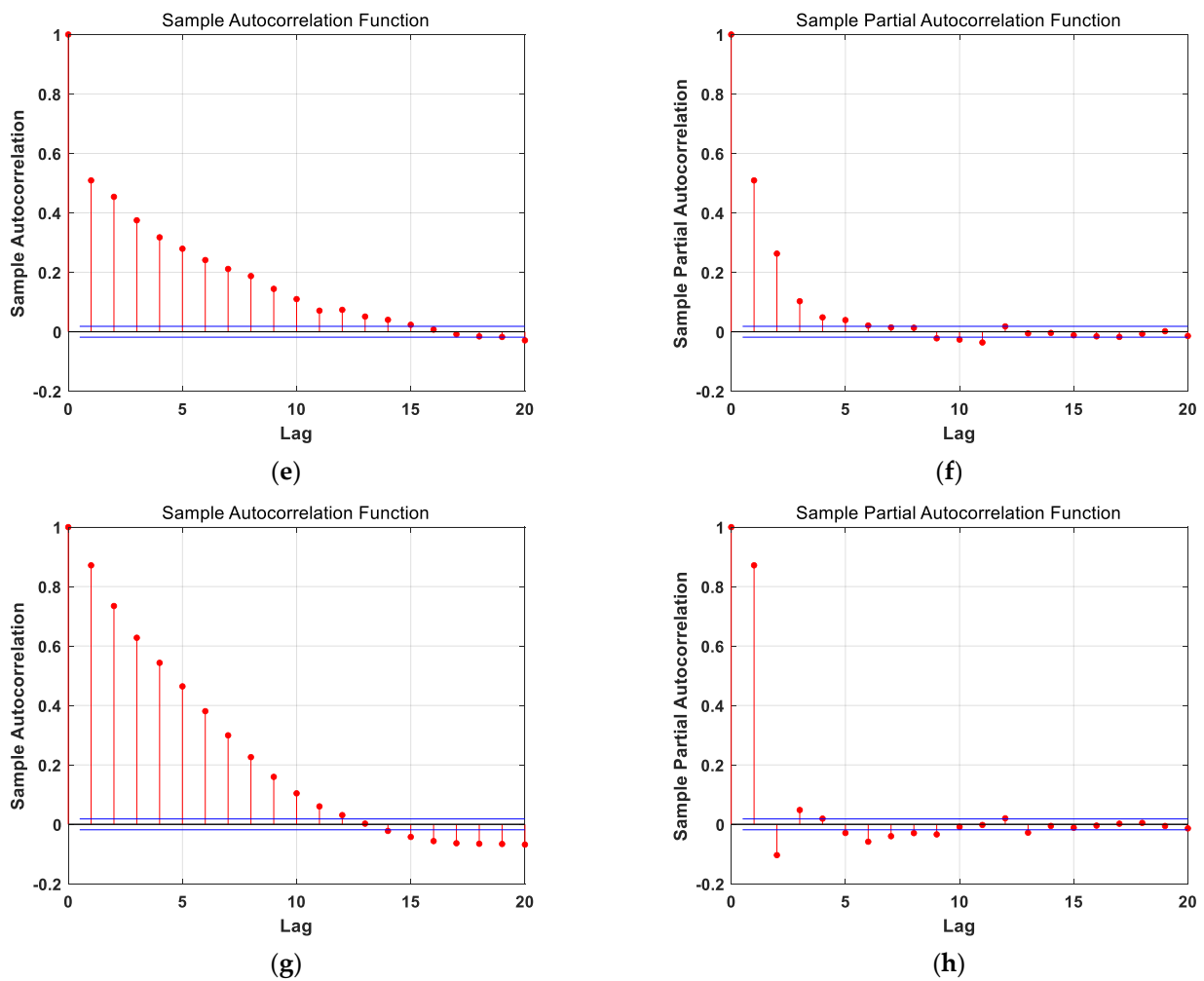


(c)



(d)

**Figure A1.** Cont.



**Figure A1.** Autocorrelation function and partial autocorrelation function diagram. (a,b)  $x$ -axis raw data, (c,d)  $x$ -axis BiLSTM network results, (e,f)  $z$ -axis raw data, and (g,h)  $z$ -axis BiLSTM network results.

The Akaike information criterion can be expressed as follows:

$$AIC(p, q) = N \ln(\delta^2(p, q)) + 2(p + q + 1) \quad (A4)$$

where  $\delta^2$  is the variance of driving white noise under each order and  $N$  is the total number of data samples. When AIC obtains the minimum value, the fitted model is the optimal model. In the case that the maximum values of  $p$  and  $q$  are set to 3, the AIC values at each order are shown in Tables A6–A9.

**Table A6.** AIC values of  $x$ -axis raw data at each order.

	$p = 0$	$p = 1$	$p = 2$	$p = 3$
$q = 0$	–	–4350.7006	–5373.4026	–5483.9134
$q = 1$	–2378.8661	–5480.4267	–5503.6485	–5503.2634
$q = 2$	–3784.2642	–5501.9738	–5502.2446	–5501.9462
$q = 3$	–4524.5464	–5504.1306	–5502.9798	–5514.9860

**Table A7.** AIC values of  $x$ -axis BiLSTM network results at each order.

	$p = 0$	$p = 1$	$p = 2$	$p = 3$
$q = 0$	–	–33245.8540	–33390.6679	–33390.1658
$q = 1$	–24407.6844	–33392.3127	–33390.3534	–33387.4822
$q = 2$	–28837.8009	–33390.3543	–33391.2919	–33390.0297
$q = 3$	–30742.3669	–33388.3649	–33390.0548	–33456.4704

**Table A8.** AIC values of  $z$ -axis raw data at each order.

	$p = 0$	$p = 1$	$p = 2$	$p = 3$
$q = 0$	–	–3384.5655	–4241.9726	–4367.2696
$q = 1$	–1992.4063	–4410.0138	–4414.9553	–4417.2529
$q = 2$	–3107.6730	–4414.3979	–4414.5702	–4415.3240
$q = 3$	–3612.4223	–4417.6140	–4415.7666	–4414.6351

**Table A9.** AIC values of  $z$ -axis BiLSTM network results at each order.

	$p = 0$	$p = 1$	$p = 2$	$p = 3$
$q = 0$	–	–31059.1193	–31186.5670	–31212.5235
$q = 1$	–23478.3934	–31198.8844	–31202.3767	–31212.2053
$q = 2$	–27430.5011	–31206.1886	–31221.9048	–31228.0632
$q = 3$	–29112.3646	–31212.9557	–31211.5881	–31262.0166

The first-order autocorrelation of the identified model residuals  $\{\omega_t\}$  was tested by the Durbin–Watson method. Assuming that the first-order correlation of  $\{\omega_t\}$  can be defined as:

$$\omega_t = \rho\omega_{t-1} + v_t \quad (\text{A5})$$

when  $\rho = 0$ , there is no first-order autocorrelation in  $\{\omega_t\}$ . Then the Durbin–Watson test value  $d$ :

$$d = \frac{\sum_{n=2}^N (\omega_n - \omega_{n-1})^2}{\sum_{n=1}^N \omega_n^2} \approx 2(1 - \rho) \quad (\text{A6})$$

The identified model's Durbin–Watson test value is shown in Table A10, indicating that the estimated model satisfies the requirements.

**Table A10.** D-W test results.

	$x$ -axis Raw Data	$x$ -axis BiLSTM Network Results	$z$ -axis Raw Data	$z$ -axis BiLSTM Network Results
Durbin–Watson test value	1.9997	2.0064	1.9998	1.9903

## References

1. Brown, A.K. Gps/ins uses low-cost mems imu. *IEEE Aerosp. Electron. Syst. Mag.* **2005**, *20*, 3–10. [[CrossRef](#)]
2. Noureldin, A.; Karamat, T.B.; Eberts, M.D.; El-Shafie, A. Performance enhancement of MEMS-based INS/GPS integration for low-cost navigation applications. *IEEE Trans. Veh. Technol.* **2008**, *58*, 1077–1096. [[CrossRef](#)]
3. Chia, J.; Low, K.; Goh, S.; Xing, Y. In A low complexity Kalman filter for improving MEMS based gyroscope performance. In Proceedings of the 2016 IEEE Aerospace Conference, Big Sky, MO, USA, 5–12 March 2016; pp. 1–7.
4. Mohammadi, Z.; Salarieh, H. Investigating the effects of quadrature error in parametrically and harmonically excited MEMS rate gyroscopes. *Measurement* **2016**, *87*, 152–175. [[CrossRef](#)]
5. Zhang, H.; Wu, Y.; Wu, W.; Wu, M.; Hu, X. Improved multi-position calibration for inertial measurement units. *Meas. Sci. Technol.* **2009**, *21*, 015107. [[CrossRef](#)]

6. Fong, W.; Ong, S.; Nee, A. Methods for in-field user calibration of an inertial measurement unit without external equipment. *Meas. Sci. Technol.* **2008**, *19*, 085202. [[CrossRef](#)]
7. Huang, L. Auto regressive moving average (ARMA) modeling method for Gyro random noise using a robust Kalman filter. *Sensors* **2015**, *15*, 25277–25286. [[CrossRef](#)] [[PubMed](#)]
8. Narasimhappa, M.; Nayak, J.; Terra, M.H.; Sabat, S.L. ARMA model based adaptive unscented fading Kalman filter for reducing drift of fiber optic gyroscope. *Sens. Actuators* **2016**, *251*, 42–51. [[CrossRef](#)]
9. Seong, S.M.; Lee, J.G.; Park, C.G. Equivalent ARMA model representation for RLG random errors. *IEEE Trans. Aerosp. Electron. Syst.* **2000**, *36*, 286–290. [[CrossRef](#)]
10. Quinchia, A.G.; Ferrer, C.; Falco, G.; Falletti, E.; Dosis, F. In Analysis and modelling of MEMS inertial measurement unit. In Proceedings of the 2012 International Conference on Localization and GNSS, Starnberg, Germany, 25–27 June 2012; pp. 1–7.
11. Kang, C.H.; Kim, S.Y.; Park, C.G. Improvement of a low cost MEMS inertial-GPS integrated system using wavelet denoising techniques. *Int. J. Aeronaut. Space Sci.* **2011**, *12*, 371–378. [[CrossRef](#)]
12. Zhang, Y.S.; Yang, T. Modeling and compensation of MEMS gyroscope output data based on support vector machine. *Measurement* **2012**, *45*, 922–926. [[CrossRef](#)]
13. Bhatt, D.; Aggarwal, P.; Bhattacharya, P.; Devabhaktuni, V. An enhanced mems error modeling approach based on nu-support vector regression. *Sensors* **2012**, *12*, 9448–9466. [[CrossRef](#)] [[PubMed](#)]
14. El-Rabbany, A.; El-Diasty, M. An efficient neural network model for de-noising of MEMS-based inertial data. *J. Navig.* **2004**, *57*, 407. [[CrossRef](#)]
15. Jiang, C.; Chen, S.; Chen, Y.; Zhang, B.; Feng, Z.; Zhou, H.; Bo, Y. A MEMS IMU de-noising method using long short term memory recurrent neural networks (LSTM-RNN). *Sensors* **2018**, *18*, 3470. [[CrossRef](#)]
16. Jiang, C.; Chen, S.; Chen, Y.; Bo, Y.; Han, L.; Guo, J.; Feng, Z.; Zhou, H. Performance analysis of a deep simple recurrent unit recurrent neural network (SRU-RNN) in MEMS gyroscope de-noising. *Sensors* **2018**, *18*, 4471. [[CrossRef](#)]
17. Jiang, C.; Chen, Y.; Chen, S.; Bo, Y.; Li, W.; Tian, W.; Guo, J. A mixed deep recurrent neural network for MEMS gyroscope noise suppressing. *Electronics* **2019**, *8*, 181. [[CrossRef](#)]
18. Zhu, Z.; Bo, Y.; Jiang, C. A MEMS Gyroscope Noise Suppressing Method Using Neural Architecture Search Neural Network. *Math. Probl. Eng.* **2019**, *2019*, 5491243. [[CrossRef](#)]
19. Barbour, N.M. *Inertial Navigation Sensors*; Charles Stark Draper Lab Inc.: Cambridge, MA, USA, 2010.
20. Jafari, M.; Najafabadi, T.A.; Moshiri, B.; Tabatabaei, S.S.; Sahebameyan, M. PEM stochastic modeling for MEMS inertial sensors in conventional and redundant IMUs. *IEEE Sens. J.* **2014**, *14*, 2019–2027. [[CrossRef](#)]
21. Cho, J.Y. High-Performance Micromachined Vibratory Rate and Rate-Integrating Gyroscopes. Ph.D. Thesis, University of Michigan, Ann Arbor, MI, USA, 2012.
22. Park, B.S.; Han, K.; Lee, S.; Yu, M. Analysis of compensation for a g-sensitivity scale-factor error for a MEMS vibratory gyroscope. *J. Micromech. Microeng.* **2015**, *25*, 115006. [[CrossRef](#)]
23. Dean, R.; Flowers, G.; Hodel, S.; MacAllister, K.; Horvath, R. In Vibration Isolation of MEMS Sensors for Aerospace Applications. In *Proceedings of the SPIE Proceedings Series*; SPIE: Bellingham, WA, USA, 2002; pp. 166–170.
24. Reid, J.R.; Bright, V.M.; Kosinski, J.A. A micromachined vibration isolation system for reducing the vibration sensitivity of surface transverse wave resonators. *IEEE Trans. Ultrason. Ferroelectr. Freq. Control* **1998**, *45*, 528–534. [[CrossRef](#)]
25. Reid, J.R.; Bright, V.M.; Stewart, J.T.; Kosinski, J.A. In Reducing the normal acceleration sensitivity of surface transverse wave resonators using micromachined isolation systems. In Proceedings of the 1996 IEEE International Frequency Control Symposium, Honolulu, HI, USA, 5–7 June 1996; pp. 464–472.
26. Dean, R.; Flowers, G.; Sanders, N.; Horvath, R.; Johnson, W.; Kranz, M.; Whitley, M. Experimental validation and testing of components for active damping control for micromachined mechanical vibration isolation filters using electrostatic actuation. In *Smart Structures and Materials 2006: Smart Electronics, MEMS, BioMEMS, and Nanotechnology*; International Society for Optics and Photonics: Bellingham, WA, USA, 2006; p. 61721C.
27. Kim, J.M.; Mok, S.H.; Leeghim, H.; Lee, C.Y. Vibration-Robust Attitude and Heading Reference System Using Windowed Measurement Error Covariance. *Int. J. Aeronaut. Space Sci.* **2017**, *18*, 555–564. [[CrossRef](#)]
28. Wu, Z.; Yao, M.; Ma, H.; Jia, W. De-noising MEMS inertial sensors for low-cost vehicular attitude estimation based on singular spectrum analysis and independent component analysis. *Electron. Lett.* **2013**, *49*, 892–893. [[CrossRef](#)]
29. Hao, X.Y.; Li, M.; Han, X.F.; Jia, H.G. In Analysis on the influence of random vibration on MEMS gyro precision and error compensation. In *Applied Mechanics and Materials*; Trans Tech Publications: Stafa, Switzerland, 2012; pp. 4164–4168.
30. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
31. Bengio, Y.; Simard, P.; Frasconi, P. Learning long-term dependencies with gradient descent is difficult. *IEEE Trans. Neural Netw.* **1994**, *5*, 157–166. [[CrossRef](#)] [[PubMed](#)]
32. Graves, A.; Schmidhuber, J. In Framewise phoneme classification with bidirectional LSTM networks. In Proceedings of the 2005 IEEE International Joint Conference on Neural Networks, Montreal, QC, Canada, 31 July–4 August 2005; pp. 2047–2052.
33. Kalman, R.E. A new approach to linear filtering and prediction problems. *J. Basic Eng. Mar* **1960**, *82*, 35–45. [[CrossRef](#)]
34. Kustiawan, I.; Chi, K.H. Handoff decision using a Kalman filter and fuzzy logic in heterogeneous wireless networks. *IEEE Commun. Lett.* **2015**, *19*, 2258–2261. [[CrossRef](#)]

35. Hosseinyalamdary, S. Deep Kalman filter: Simultaneous multi-sensor integration and modelling; A GNSS/IMU case study. *Sensors* **2018**, *18*, 1316. [[CrossRef](#)]
36. Zhang, Y.; Peng, C.; Mou, D.; Li, M.; Quan, W. An adaptive filtering approach based on the dynamic variance model for reducing MEMS gyroscope random error. *Sensors* **2018**, *18*, 3943. [[CrossRef](#)]
37. Yan, Y.; Guo, P.; Liu, L. In A novel hybridization of artificial neural networks and ARIMA models for forecasting resource consumption in an IIS web server. In Proceedings of the 2014 IEEE International Symposium on Software Reliability Engineering Workshops, Naples, Italy, 3–6 November 2014; pp. 437–442.
38. Lőrincz, I.; Tajmar, M. Identification of error sources in high precision weight measurements of gyroscopes. *Measurement* **2015**, *73*, 453–461. [[CrossRef](#)]
39. Shumway, R.H.; Stoffer, D.S. An approach to time series smoothing and forecasting using the EM algorithm. *J. Time Ser. Anal.* **1982**, *3*, 253–264. [[CrossRef](#)]
40. Wu, C.J. On the convergence properties of the EM algorithm. *Ann. Stat.* **1983**, *11*, 95–103. [[CrossRef](#)]
41. Andrieu, C.; Doucet, A. In Online expectation-maximization type algorithms for parameter estimation in general state space models. In Proceedings of the 2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, Hong Kong, China, 6–10 April 2003; p. VI-69.
42. Wen, Q.; Ge, Z.; Song, Z. Data-based linear Gaussian state-space model for dynamic process monitoring. *AIChE J.* **2012**, *58*, 3763–3776. [[CrossRef](#)]
43. Mirikitani, D.; Nikolaev, N. Nonlinear maximum likelihood estimation of electricity spot prices using recurrent neural networks. *Neural Comput. Appl.* **2011**, *20*, 79–89. [[CrossRef](#)]
44. Moore, T.J.; Sadler, B.M.; Kozick, R.J. Maximum-likelihood estimation, the Cramér–Rao bound, and the method of scoring with parameter constraints. *IEEE Trans. Signal. Process.* **2008**, *56*, 895–908. [[CrossRef](#)]
45. Hesar, H.D.; Mohebbi, M. An Adaptive Kalman Filter Bank for ECG Denoising. *IEEE J. Biomed. Health Inform.* **2020**, *25*, 13–21. [[CrossRef](#)]
46. Hartikainen, J.; Solin, A.; Särkkä, S. *Optimal Filtering with Kalman Filters and Smoothers—A Manual for MATLAB Toolbox EKF/UKF*; Aalto University School of Science: Espoo, Finland, 2011.
47. Rousseeuw, P.J.; Leroy, A.M. *Robust Regression and Outlier Detection*; John Wiley & Sons: Hoboken, NJ, USA, 2005; Volume 589.
48. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
49. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.