

Article

A MEMS IMU De-Noiseing Method Using Long Short Term Memory Recurrent Neural Networks (LSTM-RNN)

Changhui Jiang ^{1,2} , Shuai Chen ^{1,*}, Yuwei Chen ², Boya Zhang ¹, Ziyi Feng ², Hui Zhou ³ and Yuming Bo ¹

¹ School of Automation, Nanjing University of Science and Technology, Nanjing 210094, China; changhui.jiang1992@gmail.com (C.J.); soochow_njust@sina.com (B.Z.); byuming@mail.njust.edu.cn (Y.B.)

² Centre of Excellence in Laser Scanning Research, Finnish Geospatial Research Institute (FGI), Geodeetinrinne 2, FI-02431 Kirkkonummi, Finland; yuwei.chen@nls.fi (Y.C.); ziyi.feng@nls.fi (Z.F.)

³ Department of Photogrammetry and Remote Sensing, Wuhan University, 129 Luoyu Road, Wuhan 430079, China; zhouhui@whu.edu.cn

* Correspondence: c1492@163.com; Tel.: +86-138-1391-5826

Received: 29 September 2018; Accepted: 13 October 2018; Published: 15 October 2018



Abstract: Microelectromechanical Systems (MEMS) Inertial Measurement Unit (IMU) containing a three-orthogonal gyroscope and three-orthogonal accelerometer has been widely utilized in position and navigation, due to gradually improved accuracy and its small size and low cost. However, the errors of a MEMS IMU based standalone Inertial Navigation System (INS) will diverge over time dramatically, since there are various and nonlinear errors contained in the MEMS IMU measurements. Therefore, MEMS INS is usually integrated with a Global Positioning System (GPS) for providing reliable navigation solutions. The GPS receiver is able to generate stable and precise position and time information in open sky environment. However, under signal challenging conditions, for instance dense forests, city canyons, or mountain valleys, if the GPS signal is weak and even is blocked, the GPS receiver will fail to output reliable positioning information, and the integration system will fade to an INS standalone system. A number of effects have been devoted to improving the accuracy of INS, and de-noising or modelling the random errors contained in the MEMS IMU have been demonstrated to be an effective way of improving MEMS INS performance. In this paper, an Artificial Intelligence (AI) method was proposed to de-noise the MEMS IMU output signals, specifically, a popular variant of Recurrent Neural Network (RNN) Long Short Term Memory (LSTM) RNN was employed to filter the MEMS gyroscope outputs, in which the signals were treated as time series. A MEMS IMU (MSI3200, manufactured by MT Microsystems Company, Shijiazhuang, China) was employed to test the proposed method, a 2 min raw gyroscope data with 400 Hz sampling rate was collected and employed in this testing. The results show that the standard deviation (STD) of the gyroscope data decreased by 60.3%, 37%, and 44.6% respectively compared with raw signals, and on the other way, the three-axis attitude errors decreased by 15.8%, 18.3% and 51.3% individually. Further, compared with an Auto Regressive and Moving Average (ARMA) model with fixed parameters, the STD of the three-axis gyroscope outputs decreased by 42.4%, 21.4% and 21.4%, and the attitude errors decreased by 47.6%, 42.3% and 52.0%. The results indicated that the de-noising scheme was effective for improving MEMS INS accuracy, and the proposed LSTM-RNN method was more preferable in this application.

Keywords: microelectromechanical systems; inertial measurement unit; long short term memory recurrent neural networks; artificial intelligence

1. Introduction

Global Navigation Satellite System (GNSS) and Inertial Navigation System (INS) have been set up in various vehicles and carriers for navigation and tracking [1–5]. A GNSS receiver is usually a chip, which is small, low-cost and precise. With continuously receiving the signal from the navigation satellites in orbit, the GNSS receiver is able to provide reliable and constant positioning, navigation and timing (PNT) information [6–10]. However, limited by the principle that at least four satellites are essential for computing positioning and velocity, the GNSS receiver will fail to work normally under challenging signal conditions [6–10]. The navigation signal transmits from the satellites a long way, and becomes too weak when reaching the ground, therefore, it can be easily blocked temporarily by the environment. For bridging the signal outages, Inertial Navigation System (INS) is employed to output positioning information during the signal outage. Traditional fiber or laser Inertial Measurement Units (IMU) are precise, but too big and expensive for vehicles or handheld devices [11–13].

Recently, Microelectromechanical Systems (MEMS) IMU has gained a boom in applications of position and navigation, especially, vehicles, handheld devices, and precise-guidance bombs, due to its low cost and small size brought by the advanced MEMS manufacturing technology [14–16]. With proper circuit and structure design, the accuracy of MEMS IMU has gradually improved [14–16]. Although, the MEMS IMU obtained a reduction in volume, and cost compared with conventional fiber or laser IMU, the MEMS IMU experiences more non-linear or random errors, which leads to the MEMS INS navigation solutions diverging dramatically over time [14–18]. Commonly, the GNSS receiver is employed as an outer aiding to calibrating the INS, thus, the GNSS and INS integrated navigation system is able provide reliable and continuous navigation solutions, even during short-term signal outage. During the outage, the problem is that the errors of INS increase quickly without outer sensors or reference aiding. Under this condition, the modeling or de-noising of the MEMS IMU outputs will be a key step to improve the accuracy of MEMS INS. Researchers are devoted to identifying and modeling the errors contained in the MEMS IMU raw signals, which can be divided into two parts: System errors and random errors. The system part refers to the bias, and scale factor errors. These errors can be calibrated or quantified by certain experiments in a laboratory. The calibration process has been investigated and reported by a number of researchers [19–22]. However, the random part can lead to the drifts and instabilities in bias or scale factor over time, which is the key component leading to the INS errors divergence [17–26].

Therefore, before deploying MEMS IMU for navigation, an accurate model of the random and systemic is requisite to ensure the accuracy. For identifying and modeling the random errors, researchers have devoted to proposing some de-noising techniques in this random signals processing, and overall, the approaches can be divided to statistical modeling methods represented by Wavelet De-noising (WD), Allan Variance (AV), Auto Regressive and Moving Average (ARMA), and Artificial Intelligence (AI) methods represented by Support Vector Machine (SVM), Neural Networks (NN) [17–26]. Generally, the random errors contain high frequency (long-term) and low frequency (short-term) parts. Among statistical methods, the WD method performs significantly in removing the high frequency part [17–26], but it has restricted ability in removing low frequency errors. The AV method is another statistical method, and has been widely used in MEMS IMU errors analysis in time domain [17,18]. In the AV method, the stability of MEMS IIMU measurements are presented as a function of average time, and the intrinsic noise is described by five basic parts termed as: Quantization noise, angle random walk, bias instability, rate random walk, and rate ramp [17,18]. Usually, the AV method is employed to exploit the noise characteristics and obtain first-order Gauss Markov (GM) or ARMA parameters [17,18]. Traditional or conventional approaches are unsatisfactory for this application. Moreover, the unsatisfactory estimation or compensation of the random errors will lead to the failure to provide reliable navigation information estimation in short time. Another approach is AI methods including SVM and NN, which have been utilized in MEMS IMU modeling, and found to be better than other conventional methods [27–36]. These methods operate the signal de-noising or modeling as sequence prediction problem, and the MEMS IMU measurements are treated as time series.

Generally, in the data science community, sequence prediction problems have been around for a long period of time in a wide range of applications, including stock price prediction and sales pattern finding, language translation and speech recognition [37–41]. Recently, a new breakthrough has happened in the data science community, and a Long Short Term Memory Recurrent Neural Networks (LSTM-RNN) has been proposed and has been demonstrated more effective for almost all of these sequence prediction problems [37–41]. Compared with conventional RNN, LSTM-RNN introduces the “gate structure” to address the long-term memory, which allows it to have the pattern of selectively remembering for a long time. This special design or structure makes it more suitable for predicting or processing time based series data. In this paper, the LSTM-RNN is incorporated in MEMS IMU gyroscope raw signal de-noising. LSTM-RNN has performed excellently in time series signal processing, for instance stock price prediction, speech single processing, and others [37–41]. A MEMS Inertial Measurement Unit (IMU) manufactured by MT Microsystems Company known as MSI2000 IMU is employed in the experiments for testing [42]. Firstly, a common ARMA is employed to process the raw signal, then the order and parameters are determined through the auto-correlation and partial correlation operation; secondly, a single LSTM and multi-layer LSTM are compared in the MEMS gyroscope raw signal de-noising in aspects of average training loss, training time and de-noising performance; finally, the three-axis attitude errors of raw signals, ARMA, and LSTM-RNN are compared and analyzed.

The remainder of this paper is organized as: (1) Section 2 introduces the methods including Auto Regressive Moving Average Method (ARMA), and the proposed LSTM-RNN; (2) Section 3 presents the experiments, results and comparison (3) the following are the discussion, conclusion and future work.

2. Method

In this section, the conventional ARMA representing the statistical methods and the proposed LSTM-RNN representing AI methods are presented. The principles, basic equations and information flow are briefly introduced.

2.1. ARMA Model

As illustrated in previous papers [25,26], the following two steps are essential for setting up an ARMA model: (1) After the obtaining the raw gyroscope signals, auto-correlation and partial correlation are operated to characterize the noise and select the suitable time series model; (2) estimating the parameters of the ARMA model.

The auto-correlation of a signal is a product operation of the signal and a time-shifted version of the signal itself. Assuming $r(t)$ is a random signal sequence, and the auto-correlation can be modelled as [25,26]:

$$R(\tau) = E(r(t)r(t + \tau)) \quad (1)$$

where, $E(\cdot)$ is the expectation operator, τ is the time delay or shift. The partial correlation process is defined as [25,26]:

$$P(\tau) = \frac{\sum (r(t) - E(r(t)))(r(t + \tau) - E(r(t + \tau)))}{\sqrt{\sum (r(t) - E(r(t)))^2 \sum (r(t + \tau) - E(r(t + \tau)))^2}} \quad (2)$$

where, $E(\cdot)$ is the expectation operator, τ is the time delay or shift.

The ARMA model is defined as:

$$z(k) = \sum_{i=1}^p a_i z(k - i) + \sum_{j=1}^q b_j \varepsilon(k - j) + \varepsilon(k) \quad (3)$$

where, $\varepsilon(k)$ is a zero mean and unknown white noise, p and q are the order of the ARMA model, $z(k - i)$ is the input time series data, and the a_i and b_j are the related parameters, some approaches are

published for obtaining the values of these parameters, for instance Kalman filter and the least square estimation method [25,26]. Generally, the auto-correlation and partial correlation function is used to decide the order of the AMRA model.

2.2. LSTM-RNN Method

Long Short Term Memory (LSTM) is a popular variant of the common Recurrent Neural Network (RNN). An RNN composed of LSTM units is often called an LSTM network. Different from RNN, new structure termed as “gate” is added to LSTM. Commonly, a LSTM unit is composed of a cell, an “input gate”, “output gate” and a “forget gate.” The basic structure of a single layer LSTM unit is shown as Figure 1. The cell remembers values over arbitrary time intervals and the three different gates regulate and control the flow of information into and out of the cell [38–40]. Following is the detailed description of the different gates and relative equations.

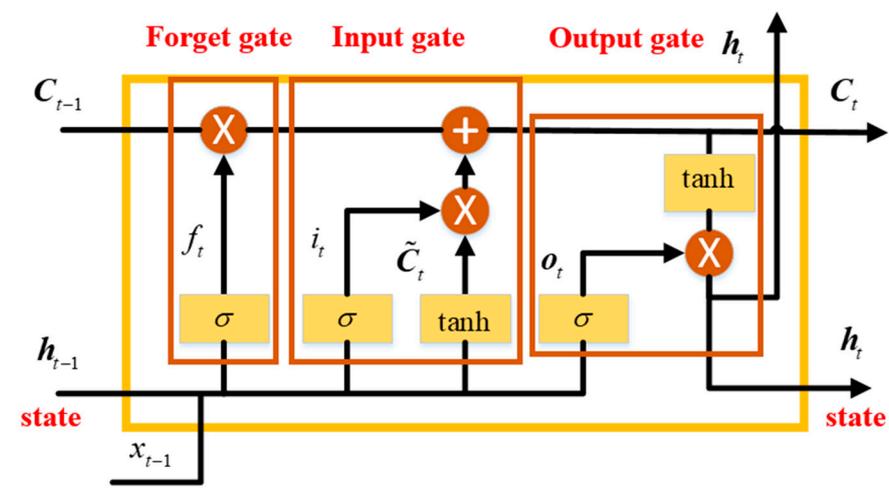


Figure 1. Basic structure of a Long Short Term Memory (LSTM) Unit.

As illustrated in Figure 1, the first part of the LSTM is the “forget gate”, which is employed to decide what information is going to get thrown away from the cell state, the decision is made by a sigmoid layer called “forget gate layer”. h_{t-1} and x_t are input to the function, and outputs a value ranging from 0 to 1 for each number in the cell state C_{t-1} . The values represent the forgetting degree of each number in the cell state, and “1” represents “completely keep this” while “0” represents “completely get rid of this”. The operation equation f_t is as:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (4)$$

where, $\sigma(\cdot)$ is a sigmoid function, W_f is the updating weights, b_f is the bias, h_{t-1} is the hidden state, and x_t is the input vector.

After deciding the memory of the previous hidden state, and the second part is the “input gate”, which is utilized to decide what new information is going to be stored in the current cell state. This gate is composed of two parts: (1) A sigmoid layer to decide what values are going to be updated, the output values i_t range from 0 to 1, and they represent the updating degree of each number in input; (2) another part is a \tanh layer which creates a vector of new candidate values \tilde{C}_t , which will be added to the cell state after multiplied with the decision vector i_t . The relative equations are as following:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (5)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (6)$$

where, $\sigma(\cdot)$ is a sigmoid function, W_i is the updating weights, b_i is the bias in the input gate, h_{t-1} is the hidden state at time $t - 1$, W_C is the updating weights, b_C is the bias, and x_t is the input vector.

The last part is the “output gate”, which is employed to decide what is going to output, similarly, a sigmoid layer outputs values o_t , which is employed to decide what parts of the cell state will be output, then the cell state is put through a \tanh function. After this operation, the cell state values are pushed to be between -1 and 1 . Finally, the results are multiplied by the output of the sigmoid gate, and the output parts are decided. The related equations are as:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t * \tanh(C_t) \tag{8}$$

where, W_o is the updating weights, and b_o is the bias in the output gate, C_t is the cell state at time t .

The above Equations (4)–(8) describes the basic LSTM unit for RNN, which is just a single LSTM unit. Figure 2 presents a sequence of LSTM-RNN units in time domain. In Figure 2a, is a simple description of the LSTM-RNN working flow. The output of LSTM-RNN is decided by not only current state, but a long-term memory. Figure 2b gives the details. The cell state and hidden layer is covered to the next LSTM Unit, and the inner gate will decide the memory degree of past information. In addition, before it is employed for prediction, a training procure is necessary for determining the unknown parameters in the above Equations [37–40].

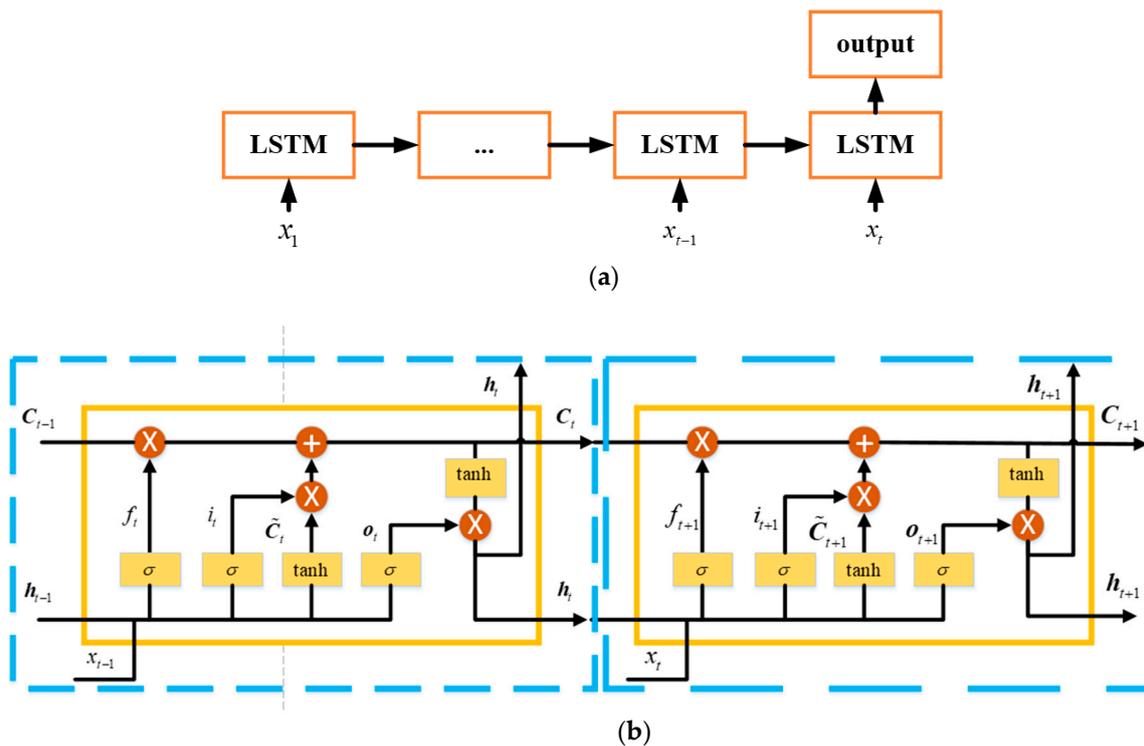


Figure 2. Working flow of the LSTM-RNN. (a) brief working flow of LSTM-RNN (Recurrent Neural Network); (b) a sequence of LSTM Unit.

3. Experiments and Results

This section will present experiments and the relative analyses for evaluating the performance of the proposed LSTM-RNN method. The laboratory experiments are conducted using the data which was collected from a MEMS IMU (MSI3200) manufactured by MT Microsystems Company, Shijiazhuang, China [29]. The real picture and the specifications of the IMU are as Figure 3 and Table 1 respectively. The gyroscope bias stability is $\leq 10^\circ/h$, and the random walk is $\leq 10^\circ/\sqrt{h}$. The

accelerometer bias was 0.5 mg, and the bias stability was 0.5 mg. The IMU was placed statically on the table, and the sampling frequency was 400 Hz. Thus, the amount of data was 48,000. The gyroscope output data unit was in degree/s. The raw noisy data of the X-axis gyroscope output is shown as Figure 3 (red line representing the raw data), and the bias was reduced before modeling the errors. After removing the bias, the result is presented in Figure 4 (blue line representing the data excluding bias). Note that the program in this experiment was developed in Python with the Tensorflow package, which is operated in an Alienware R2 PC installed an i7 Intel CPU and 16 GB random memory.



Figure 3. MSI3200 Inertial Measurement Unit.

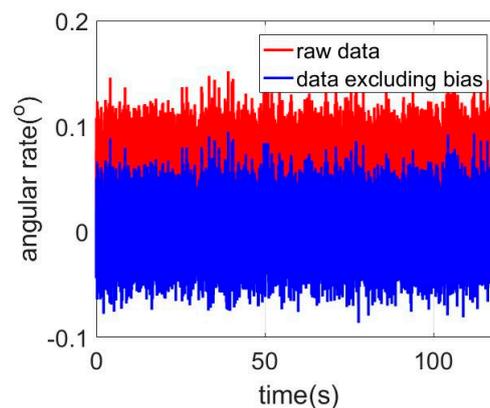


Figure 4. X-axis gyroscope signals.

Table 1. Specifications of MSI3200 IMU (Inertial Measurement Unit).

	Gyroscope	Range	$\pm 300^\circ/\text{s}$
		Bias stability (1σ)	$\leq 10^\circ/\text{h}$
		Bias stability (Allan)	$\leq 2^\circ/\text{h}$
		Angle random walk	$\leq 10^\circ/\sqrt{\text{h}}$
MEMS IMU	Accelerometer	range	$\pm 15 \text{ g}$
		Bias stability (1σ)	0.5 mg
		Bias stability (Allan)	0.5 mg
	Power consumption		1.5 W
	Weight		250 g
	Size		70 mm × 54 mm × 39 mm
	Sampling rate		400 Hz

The remainder of this section is divided into three parts: (1) The first part describes the results using the ARMA method to model the errors with presenting the auto-correction and partial correction results. The ARMA models are given according to the auto-correction and partial correction results. The standard deviation (STD) values of the signals are compared with the corresponding raw gyroscope

signals; (2) the second part is the results using the LSTM-RNN to model the errors, and presenting the training time and prediction accuracy for different input vector length. Moreover, a multi-layer LSTM-RNN is designed and compared with a single-layer LSTM-RNN in terms training time, computation load and performance; (3) the last part presents the comparisons conducted between the ARMA and LSTM-RNN, including statistical results and position results.

3.1. Error Modeling Using ARMA

For time series analysis, Auto-Correlation Function (ACF) and Partial Auto-Correlation Function (PACF) characteristics are usually employed to select the proper model. As aforementioned in Section 2, the ACF and PACF are presented as Equations (1) and (2), and then Figures 4–6 show the ACF and PACF results of the X-axis, Y-axis, and Z-axis gyroscope raw signals respectively, which are processed according to the Equations (1) and (2). From Figures 5–7, it is evident and obvious that ACF and PACF of the three-axis gyroscope are tail off. Thus, the ARMA model is suitable for this application, and the order of the ARMA model is determined using the results. More details about parameters determination can be found in the references [25,26]. Therefore, ARMA models for these three-axis gyroscope signals are presented as:

$$z(k) = 0.3475z(k-1) + 0.163z(k-2) - 0.0508\varepsilon(k-1) + \varepsilon(k) \quad (9)$$

$$z(k) = 0.5065z(k-1) + 0.2583z(k-2) - 0.2371\varepsilon(k-1) + \varepsilon(k) \quad (10)$$

$$z(k) = 0.3309z(k-1) + 0.2592z(k-2) + 0.132\varepsilon(k-1) + \varepsilon(k) \quad (11)$$

where, the $z(k)$ is the data at time $\varepsilon(k)$ is the white noise at time k . The results from the ARMA is listed in Table 2. Compared with raw signal, the standard deviation (STD) of the three-axis gyroscope outputs decrease by 31.1%, 20.0% and 25.0%. The results show the ARMA performs effectively for de-noising MEMS gyroscope raw signals. Specifically, the parameters or orders are fixed in this experiment.

Table 2. Standard deviation of Auto Regressive and Moving Average (ARMA) modeling results for the three-axis gyroscope.

	X	Y	Z
Raw data	0.0247	0.035	0.056
ARMA	0.017	0.028	0.042

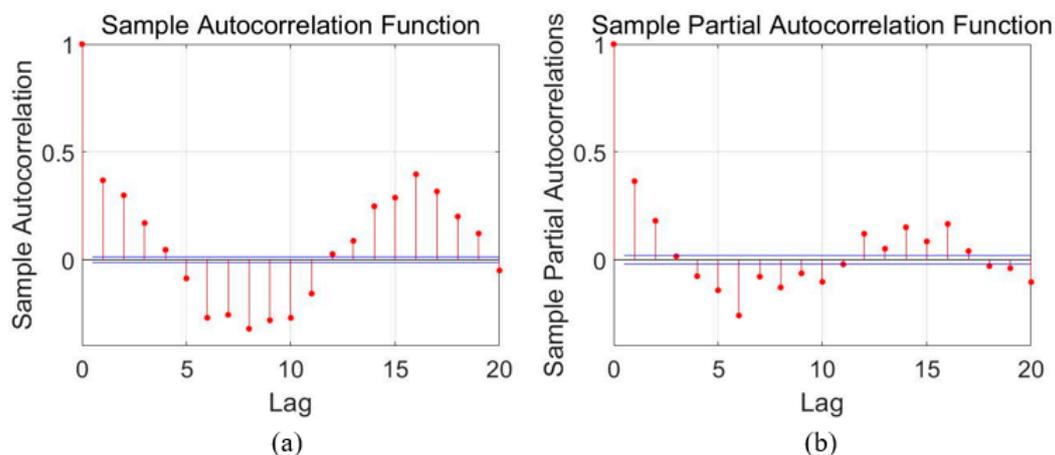


Figure 5. Auto-correlation and partial correlation analysis results of X-axis gyroscope signals. (a) autocorrelation analysis diagram; (b) Partial correlation analysis diagram.

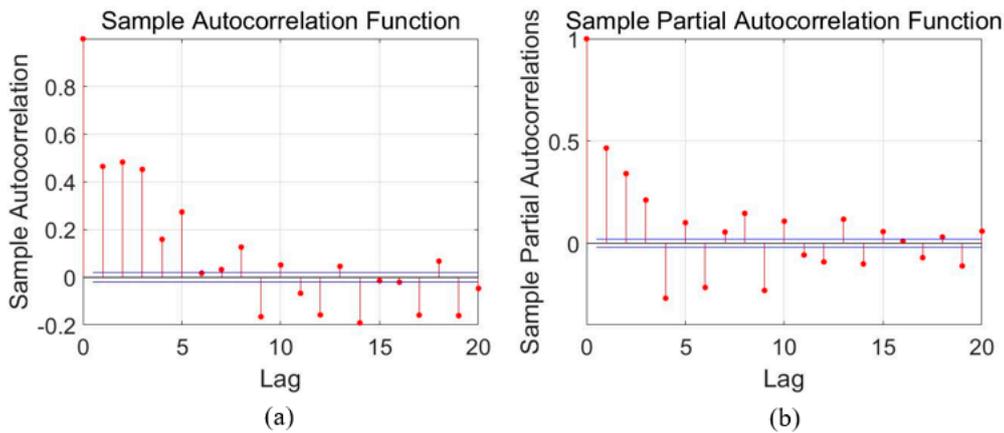


Figure 6. Auto-correlation and partial correlation analysis results of Y-axis gyroscope signals. (a) autocorrelation analysis diagram; (b) Partial correlation analysis diagram.

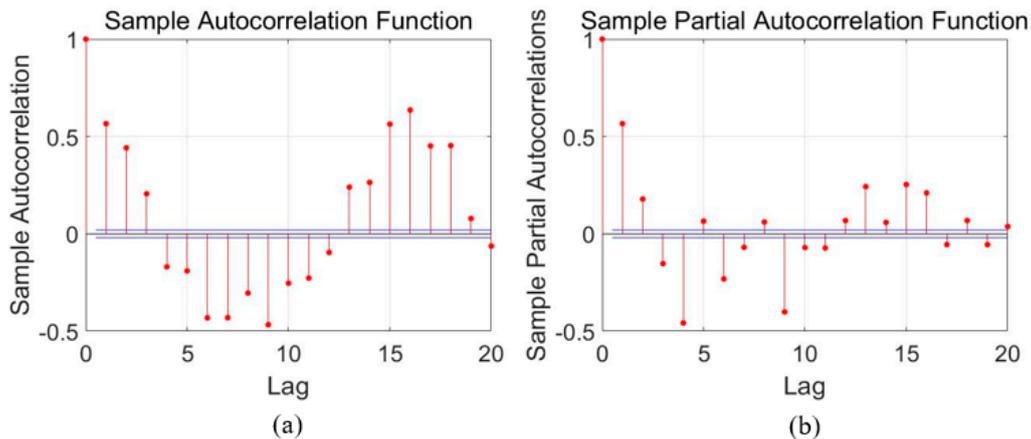


Figure 7. Auto-correlation and partial correlation analysis results of Z-axis gyroscope signals. (a) autocorrelation analysis diagram; (b) Partial correlation analysis diagram.

3.2. Error Modeling Using LSTM-RNN

In this proposed LSTM-RNN method, the employed MEMS gyroscope dataset is labeled as $[x_1, x_2, \dots, x_N]$, the subscript N is termed as the amount of the IMU data samples. The dataset is divided into a training and testing part. The training part is used to build the model and the testing part is used to verify the model. The input data vector for training is defined as:

$$Input_i = [x_i, x_{i+1}, \dots, x_{i+step}], i \in [1, N - step] \quad (12)$$

The output data vector is defined as:

$$Onput_i = [x_{i+step+1}], i \in [1, N - step] \quad (13)$$

In above equations, the variable $step$ is the length of the input data vector for training procedure. Suitable values of the input vector length $step$ is identified to realize a tradeoff between training time and the prediction performance. Table 3 shows the three axis gyroscope data training results and standard deviation (STD) of the prediction. In this test, the 5, 10, 15, 20 and 30 are the selected values of the vector length. Table 4 shows the comparison results. The training dataset length is 1000, and the testing dataset length is 48,000 (2 min with 400 Hz sampling rate). The specifications of the LSTM-RNN are presented in Table 5. The consumption time increases with the input vector length, and the STD values decrease first, and then increase. Hence, 20 is selected as the length of input vector, which is the best tradeoff between the STD and the computation time.

Table 3. Standard deviation of the raw and predicted datasets by single LSTM-RNN.

	X	Y	Z
Raw data	0.0247	0.035	0.056
Single LSTM-RNN	0.0098	0.022	0.031

Table 4. Performance of gyroscope X-axis with varying values of input vector.

Length	STD	Time (s)
5	0.0096	1.25
10	0.0095	1.33
15	0.0063	2.13
20	0.0052	2.90
30	0.0094	2.98

Table 5. Specifications of LSTM-RNN.

Batch size	128
Training epoch	50
Learning rate	0.01
Hidden unit amount	1

As shown in Table 5, the noises are considerably decreased using the LSTM-RNN, and the STD values decrease by 60.3%, 37%, and 44.6%. As aforementioned, the training epoch was set to 50, and a multi-layer LSTM-RNN was designed and compared with single LSTM-RNN. Table 6 shows the results. The multi-layer LSTM-RNN (two hidden units) has a lower average training loss at epoch of 50, which decreases by 55.4%, 34.2% and 32.1%. However, it seems that there is no obvious advance in filtering performance. The STD values of the filters data have no reduction, and are even a little higher. The operation time consumption is almost twice that of single LSTM-RNN.

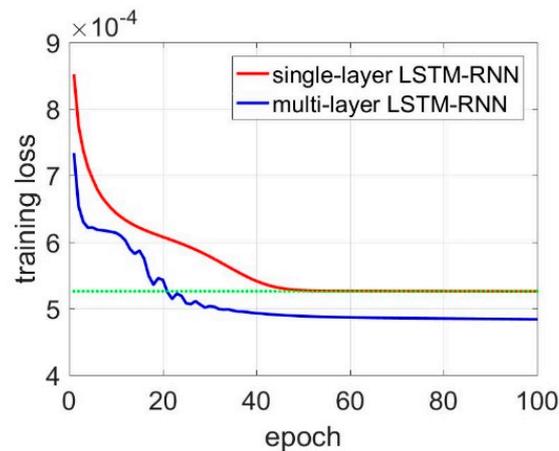
Figure 8 shows the training loss comparison of single-layer LSTM RNN and multi-layer LSTM-RNN, and they have identical accuracy at the 20th epochs. Thus, multi-layer LSTM-RNN was trained with 20 epochs, and Table 7 shows the results for the multi-layer LSTM-RNN. The results were compared with single LSTM-RNN with 50 training epochs. The average training losses have a slight increase, and the STD values are almost identical to that of multi-layer LSTM-RNN with 50 training epochs. However, compared with the single LSTM-RNN, the time consumption of multi-layer LSTM-RNN is less than that of single LSTM-RNN, which is initiated by the training epochs reducing in multi-layer LSTM-RNN. In aspects of the STD values, the de-noised three-axis gyroscope outputs have an improvement of 22.4%, 9.1%, and 22.6% respectively compared with single-layer LSTM-RNN trained after 50 epochs. This decline in accuracy means the multi-layer LSTM-RNN with fewer training epochs have weaker generation ability, since the multi-layer LSTM-RNN has more parameters which need more training epochs. Thus, while the training epochs are set to 20, the multi-layer has slightly worse STD values compared with the single-layer LSTM-RNN.

Table 6. Comparison of Single LSTM-RNN and multi-layer LSTM-RNN.

	X			Y			Z		
	Training Loss	STD	Time	Training Loss	STD	Time	Training Loss	STD	Time
Single LSTM-RNN	0.00053	0.0098	4.52	0.0010	0.022	4.94	0.022	0.031	4.56
Multi-layer LSTM-RNN	0.000467	0.011	9.31	0.0009	0.023	9.21	0.014	0.038	8.65
Raw data	/	0.0246	/	/	0.0352	/	/	0.056	/

Table 7. Comparison of single LSTM-RNN with 50 training epochs and multi-layer LSTM-RNN with 20 training epochs.

	X			Y			Z		
	Training Loss	STD	Time	Training Loss	STD	Time	Training Loss	STD	Time
Single LSTM-RNN	0.00053	0.0098	4.52	0.0010	0.022	4.94	0.022	0.031	4.56
Multi-layer LSTM-RNN	0.00045	0.012	3.68	0.0009	0.024	3.82	0.017	0.038	3.76
Raw data	/	0.0246	/	/	0.0352	/	/	0.056	/

**Figure 8.** Training loss comparison between single-layer LSTM and multi-layer LSTM.

3.3. Comparisons of ARMA and LSTM-RNN

This part presents the comparisons between ARMA and LSTM-RNN. Table 8 shows the STD results from the ARMA and LSTM-RNN de-noising methods. Compared with raw signals, STD values of the three-axis gyroscope data from the ARMA method perform a 31.2%, 20.0% and 25.0% improvement, and the STD values of the single-layer LSTM-RNN de-noised signals decrease by 42.4%, 21.4% and 21.4% respectively. With the same MEMS IMU dataset, the LSTM-RNN has an obvious improvement of 42.3%, 21.4% and 26.2% respectively for the three-axis gyroscope dataset.

Further, Figure 9 shows the attitude errors of the ARMA and LSTM-RNN de-noised MEMS IMU data. In the Figure 9, the blue line represents the position errors of the raw signals from MEMS IMU, the green line represents the position errors of ARMA de-noised MEMS IMU, and the red line represents the position errors of the designed single LSTM-RNN de-noised MEMS IMU. Table 9 shows the maximum errors of the three-axis gyroscope, compared with raw signal. The pitch, roll angle and yaw angle errors decreased by 15.8%, 18.3% and 51.3% respectively. Specifically, the pitch error decreases from -5.07° to -4.27° , the roll error decreases from -1.95° to -1.60° , and the yaw angle error decreases from -3.85° to -1.87° . Moreover, the errors from signals de-noised by LSTM-RNN decreased by 47.6%, 42.3% and 52.0%, further compared with ARMA results. To be specific, the pitch, roll and yaw angles have an extra improvement of 2.04° , 0.66° and 0.96° compared with that of ARMA. The ARMA employed in this experiment was operated with fixed parameters with selected part of the dataset. Thus, in the testing dataset, more feasible parameters are available and suitable for better performance. However, the single-layer LSTM and ARMA were tested with the identical dataset, this might demonstrate that LSTM-RNN have better generation ability in this application. This is what we think might account for the accuracy improvement of the single-layer LSTM-RNN compared with the common ARMA method. In addition, the yaw errors from the LSTM-RNN de-noised signals have an upward trend which is different from the errors from the ARMA and raw signals. We think the principle of the LSTM-RNN may account for this, and the specifications need more and further

investigation. Overall, the results demonstrate the effectiveness of LSTM-RNN in MEMS gyroscope signals de-noising.

Table 8. Standard deviation of ARMA modeling results for the three-axis gyroscope.

	X	Y	Z
Raw data	0.0247	0.035	0.056
ARMA	0.017	0.028	0.042
Single LSTM-RNN	0.0098	0.022	0.031

Table 9. Maximum attitude errors from raw, ARMA and LSTM-RNN

	Pitch	Roll	Yaw
Raw data	-5.070	-1.952	-3.853
ARMA	-4.268	-1.601	-1.873
Single LSTM-RNN	-2.231	-0.942	0.826

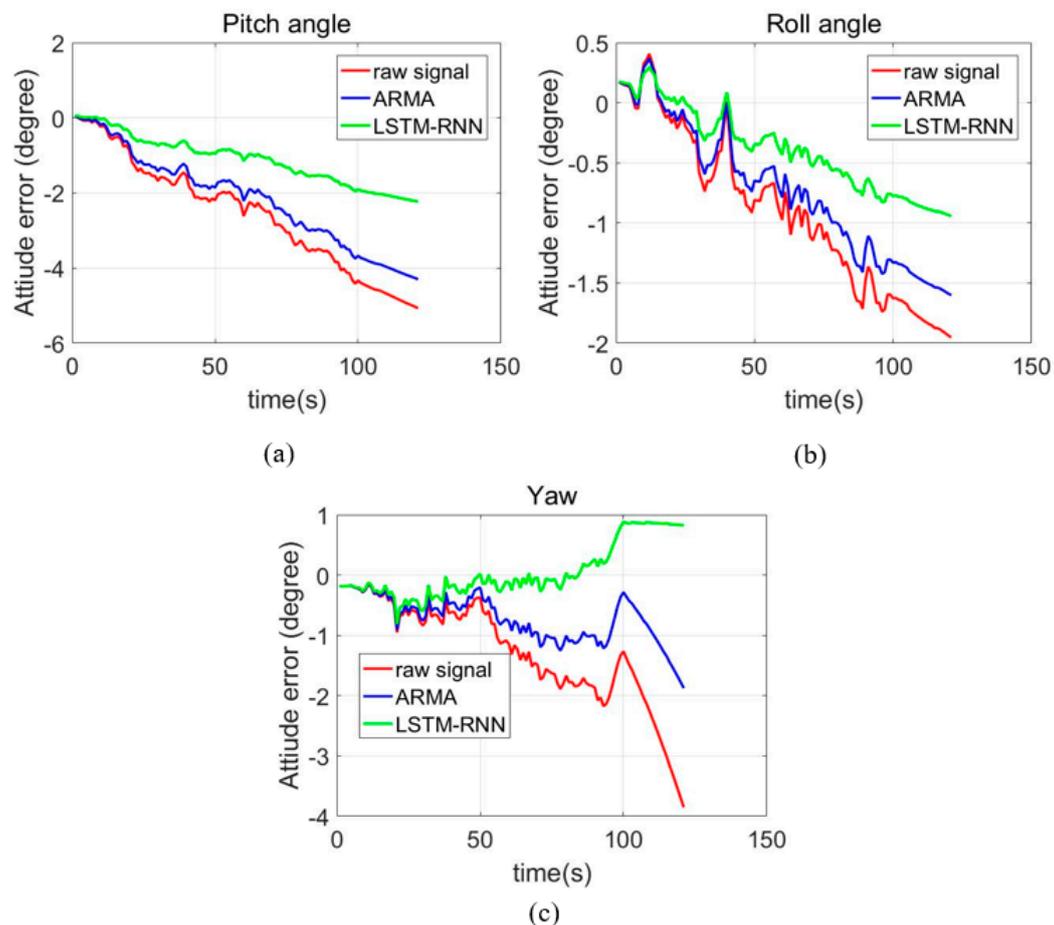


Figure 9. LSTM-RNN MEMS IMU attitude errors. (a) Pitch angle; (b) Roll angle; (c) Yaw angle

4. Discussion

1. In this paper, limited by the computing capacity of the employed computer, the LSTM-RNN had a limited amount of layers, which might have a negative influence on the generation ability of LSTM-RNN and the prediction performance in the long term.
2. In this paper, just one of the RNN variants LSTM-RNN were employed and evaluated in this application, and it has significant meaning to explore different LSTM-RNN structures more suitable for MEMS IMU errors modeling and de-noising.

3. This method was tested only using static data, and dynamic trajectory data should be included for fully evaluating the proposed method. The noise characteristics in dynamic environment may be different from that in dynamics.

5. Conclusions

This paper discussed a LSTM-RNN based MEMS IMU errors modelling method. A MEMS IMU (MSI 3200) was employed for testing the proposed method. Through the comparisons, three major conclusions were drawn as: (1) LSTM-RNN outperformed the ARMA in this application. Compared with the ARMA model, the standard deviation of the single LSTM-RNN de-noised signals decreased by 42.4%, 21.4% and 21.4% respectively, and the attitude errors decreased by 47.6%, 42.3% and 52.0%; (2) multi-layer LSTM-RNN was able to realize the settled average training loss with less training epochs. However, the multi-layer LSTM-RNN did not outperform the single LSTM-RNN in standard deviation values of the prediction. When the training epoch was set to 20, the multi-layer had a slightly better prediction accuracy with less computation time than the single LSTM-RNN.

In addition, we think some more details are worthy of being investigated further in the future: (1) It is meaningful to investigate the deep LSTM-RNN network, which should be trained with a large amount data. Well trained deep LSTM-RNN has been demonstrated to be more feasible in some applications. A deep LSTM-RNN will be implemented and presented in future; (2) many variants of RNN are published and have been demonstrated effectively in solving time series prediction problems. It is meaningful to further investigate and compare their performance, and find more preferable neural networks suitable for this particular application. Comparison of several popular variants of RNN will be presented in future.

Author Contributions: C.J. proposed the idea, developed the LSTM-RNN Python software, and written the first version of this paper. S.C. guided the paper writing, reviewed the paper and offered the funding. Y.C. revised the paper, discussed the idea and guided the paper writing. B.Z. collected and processed the data. Z.F., H.Z. and Y.B. reviewed the paper before submission.

Funding: This research was funded by the Fundamental Research Funds for the Central Universities (Grant No. 30917011105); the National Defense Basic Scientific Research program of China (Grant No. JCKY2016606B004); Jiangsu Planned Projects for Postdoctoral Research Funds, grant number (Grant No. 1501050B); the China Postdoctoral Science Foundation, grant number (Grant No. 2015M580434) and the special grade of the financial support from the China Postdoctoral Science Foundation with grant number (Grant No. 2016T90461).

Acknowledgments: The author gratefully acknowledges the financial support from China Scholarship Council (CSC, Grant No. 201806840087) and excellent doctor training fund of Nanjing University of Science and Technology (NJUST). If you want the source code, please e-mail me, and I will share the all the materials.

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

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.* **2009**, *58*, 1077–1096. [[CrossRef](#)]
3. Chen, Y.; Tang, J.; Jiang, C.; Zhu, L.; Lehtomäki, M.; Kaartinen, H.; Kaijaluoto, R.; Wang, Y.; Hyyppä, J.; Hyyppä, H.; et al. The accuracy comparison of three simultaneous localization and mapping (SLAM)-Based indoor mapping technologies. *Sensors* **2018**, *18*, 3228. [[CrossRef](#)] [[PubMed](#)]
4. Zhang, X.; Zhu, F.; Tao, X.; Duan, R. New optimal smoothing scheme for improving relative and absolute accuracy of tightly coupled GNSS/SINS integration. *GPS Solut.* **2017**, *21*, 861–872. [[CrossRef](#)]
5. Tang, J.; Chen, Y.; Chen, L.; Liu, J.; Hyyppä, J.; Kukko, A.; Kaartinen, H.; Hyyppä, H.; Chen, R. Fast fingerprint database maintenance for indoor positioning based on UGV SLAM. *Sensors* **2015**, *15*, 5311–5330. [[CrossRef](#)] [[PubMed](#)]
6. Jiang, C.; Chen, S.; Bo, Y.; Sun, Z.; Lu, Q. Implementation and performance evaluation of a fast relocation method in a GPS/SINS/CSAC integrated navigation system hardware prototype. *IEICE Electron. Express* **2017**, *14*, 20170121. [[CrossRef](#)]

7. Ma, L.; You, Z.; Liu, T.; Shi, S. Coupled integration of CSAC, MIMU, and GNSS for improved PNT performance. *Sensors* **2016**, *16*, 682. [[CrossRef](#)] [[PubMed](#)]
8. Petritoli, E.; Leccese, F. Improvement of altitude precision in indoor and urban canyon navigation for small flying vehicles. In Proceedings of the 2015 IEEE Metrology for Aerospace (MetroAeroSpace), Benevento, Italy, 4–5 June 2015.
9. Jiang, C.; Chen, S.; Bo, Y.; Sun, Z.; Lu, Q. Performance Analysis of GNSS Vector Tracking Loop Based GNSS/CSAC Integrated Navigation System. *J. Aeronaut. Astronaut. Aviat.* **2017**, *49*, 289–297.
10. Fernández, E.; Calero, D.; Parés, M.E. CSAC Characterization and Its Impact on GNSS Clock Augmentation Performance. *Sensors* **2017**, *17*, 370. [[CrossRef](#)] [[PubMed](#)]
11. Lee, B. Review of the present status of optical fiber sensors. *Opt. Fiber Technol.* **2003**, *9*, 57–79. [[CrossRef](#)]
12. Narasimhappa, M.; Sabat, S.L.; Nayak, J. Fiber-optic gyroscope signal denoising using an adaptive robust Kalman filter. *IEEE Sens. J.* **2016**, *16*, 3711–3718. [[CrossRef](#)]
13. 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 A Phys.* **2016**, *251*, 42–51. [[CrossRef](#)]
14. Jiang, C.; Chen, S.; Chen, Y.; Bo, Y. Research on Chip Scale Atomic Clock Driven GNSS/SINS Deeply Coupled Navigation System for Augmented Performance. *IET Radar Sonar Navig.* **2018**. [[CrossRef](#)]
15. Eling, C.; Klingbeil, L.; Kuhlmann, H. Real-time single-frequency GPS/MEMS-IMU attitude determination of lightweight UAVs. *Sensors* **2015**, *15*, 26212–26235. [[CrossRef](#)] [[PubMed](#)]
16. Jiang, C.; Chen, S.; Bo, Y.; Qu, Y.; Han, N. Research of fast relocation technology assisted by IMU in the GPS/SINS ultra-tightly coupled navigation system. *J. Aeronaut. Astronaut. Aviat.* **2016**, *48*, 253–259.
17. El-Sheimy, N.; Hou, H.; Niu, X. Analysis and modeling of inertial sensors using Allan variance. *IEEE Trans. Instrum. Meas.* **2008**, *57*, 140–149. [[CrossRef](#)]
18. Quinchia, A.G.; Falco, G.; Falletti, E.; Dovis, F.; Ferrer, C. A comparison between different error modeling of MEMS applied to GPS/INS integrated systems. *Sensors* **2013**, *13*, 9549–9588. [[CrossRef](#)] [[PubMed](#)]
19. Vaccaro, R.J.; Zaki, A.S. Statistical modeling of rate gyros. *IEEE Trans. Instrum. Meas.* **2012**, *61*, 673–684. [[CrossRef](#)]
20. Syed, Z.; Aggarwal, P.; Goodall, C.; Niu, X.; El-Sheimy, N. A new multi-position calibration method for MEMS inertial navigation systems. *Meas. Sci. Technol.* **2007**, *18*, 1897. [[CrossRef](#)]
21. Aggarwal, P.; Syed, Z.; Niu, X.; El-Sheimy, N. A standard testing and calibration procedure for low cost MEMS inertial sensors and units. *J. Navig.* **2008**, *61*, 323–336. [[CrossRef](#)]
22. Bekkeng, J.K. Calibration of a novel MEMS inertial reference unit. *IEEE Trans. Instrum. Meas.* **2009**, *58*, 1967–1974. [[CrossRef](#)]
23. 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](#)]
24. Chen, D.; Han, J. Application of wavelet neural network in signal processing of MEMS accelerometers. *Microsyst. Technol.* **2011**, *17*, 1–5. [[CrossRef](#)]
25. 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](#)]
26. Huang, L.; Li, Z.; Xie, F.; Feng, K. Novel time series modeling methods for gyro random noise used in Internet of Things. *IEEE Access* **2018**, *6*, 47911–47921. [[CrossRef](#)]
27. Wang, L.; Wei, G.; Zhu, Y.; Liu, J.; Tian, Z. Real-time modeling and online filtering of the stochastic error in a fiber optic current transducer. *Meas. Sci. Technol.* **2016**, *27*, 105103. [[CrossRef](#)]
28. Bhatt, D.; Aggarwala, P.; Devabhaktunia, V.; Bhattacharyab, P. A novel hybrid fusion algorithm to bridge the period of GPS outages using low-cost INS. *Expert Syst. Appl.* **2014**, *41*, 2166–2173. [[CrossRef](#)]
29. Kopáček, A.; Kajánek, P.; Lipták, I. Systematic error elimination using additive measurements and combination of two low cost IMs. *IEEE Sens. J.* **2016**, *16*, 6239–6248. [[CrossRef](#)]
30. Yang, H.; Li, W.; Luo, T.; Liang, H.; Zhang, H.; Gu, Y.; Luo, C. Research on the Strategy of Motion Constraint-Aided ZUPT for the SINS Positioning System of a Shearer. *Micromachines* **2017**, *8*, 340. [[CrossRef](#)]
31. Ning, Y.; Wang, J.; Han, H.; Tan, X.; Liu, T. An Optimal Radial Basis Function Neural Network Enhanced Adaptive Robust Kalman Filter for GNSS/INS Integrated Systems in Complex Urban Areas. *Sensors* **2018**, *18*, 3091. [[CrossRef](#)] [[PubMed](#)]
32. Jerath, K.; Brennan, S.; Lagoa, C. Bridging the gap between sensor noise modeling and sensor characterization. *Measurement* **2018**, *116*, 350–366. [[CrossRef](#)]

33. 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](#)]
34. Chu, Y.; Fei, J. Adaptive global sliding mode control for MEMS gyroscope using RBF neural network. *Math. Probl. Eng.* **2015**. [[CrossRef](#)]
35. El-Rabbany, A.; El-Diasty, M. An efficient neural network model for de-noising of MEMS-based inertial data. *J. Navig.* **2004**, *57*, 407–415. [[CrossRef](#)]
36. Xing, H.; Hou, B.; Lin, Z.; Guo, M. Modeling and Compensation of Random Drift of MEMS Gyroscopes Based on Least Squares Support Vector Machine Optimized by Chaotic Particle Swarm Optimization. *Sensors* **2017**, *17*, 2335. [[CrossRef](#)] [[PubMed](#)]
37. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
38. Gallicchio, C. Short-term memory of deep rnn. *arXiv* **2018**, arXiv:1802.00748.
39. Li, X.; Wu, X. Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, Australia, 19–24 April 2015.
40. Understanding LSTM Networks. Available online: <https://colah.github.io/posts/2015-08-Understanding-LSTMs> (accessed on 27 August 2015).
41. Hosseinyalamdary, S. Deep Kalman Filter: Simultaneous Multi-Sensor Integration and Modelling; A GNSS/IMU Case Study. *Sensors* **2018**, *18*, 1316. [[CrossRef](#)] [[PubMed](#)]
42. MT Microsystems. Available online: http://www.mtmems.com/product_view.asp?id=28 (accessed on 30 September 2018).



© 2018 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 (<http://creativecommons.org/licenses/by/4.0/>).