Hindawi
International Journal of Aerospace Engineering
Volume 2019, Article ID 9142694, 11 pages
https://doi.org/10.1155/2019/9142694

Research Article

Terrain Referenced Navigation Using a Multilayer Radial Basis Function-Based Extreme Learning Machine

Jungshin Lee, Changky Sung, and Juhyun Oh

The 3rd R&D Institute 4th Center, Agency for Defense Development, Daejeon 34060, Republic of Korea

Correspondence should be addressed to Jungshin Lee; jslee0534@add.re.kr

Received 9 November 2018; Revised 27 March 2019; Accepted 7 April 2019; Published 11 July 2019

Academic Editor: Joseph Morlier

A high-resolution digital elevation model (DEM) is an important element that determines the performance of terrain referenced navigation (TRN). However, the higher the resolution of the DEM, the bigger the memory size needed for storing it. It is difficult to secure such large memory spaces in small, low-priced unmanned aerial vehicles. In this study, a high-precision terrain regression model to fit the DEM is generated using the extreme learning machine technique based on the multilayer radial basis function. The TRN results using the proposed method are compared with existing studies on various DEM fitting methods. This study verifies that the proposed method obtains improved fitting accuracy and TRN performance over existing DEM fitting methods such as bilinear interpolation, SVM for regression, and bi-spline neural network, without the DEM storage space.

1. Introduction

Terrain referenced navigation (TRN) is a navigation technology that estimates the position of an aircraft by comparing the altitude measured by an altimeter with the digital elevation model (DEM). Currently, the global navigation satellite system (GNSS) is commonly used for precise navigation [1] but an alternative is needed for environments in which the GNSS is unavailable due to interference such as hostile jamming. Thus, studies on TRN are actively being carried out as an alternative that will ensure precise navigation performance when GNSS is denied [2–5].

The DEM is a map that stores the elevation values of terrain to represent a terrain's surface in three dimensions, and a high-resolution DEM is essential for improving the performance of TRN. However, large amounts of memory space are needed to store a high-resolution DEM and the loading time of the DEM in a separate storage space interferes with real-time computation. In addition, various interpolation methods are being used to obtain terrain elevation at the desired location using a discontinuous DEM. This method also generates additional computational errors, which can be larger as the resolution of the DEM decreases. To alleviate this problem, studies are continuously being conducted on devising a continuous terrain elevation model through a regression method based on the machine learning technique [6–10]. However, studies on DEM fitting using machine learning techniques up until now were conducted on DEM data in small areas of several hundred square meters or were unable to acquire higher accuracy than interpolation methods. Furthermore, excluding the method proposed by Liu et al. [6], there have been no studies that applied actual TRN using trained regression models. Also, in the case of Liu et al. [6], applying low-resolution DEM learning results in narrow areas would be inappropriate for actual high-performance TRN. In addition to the necessity of very high fitting accuracy for application in TRN, there is also the constraint of guaranteeing real-time operation. Taking into consideration such conditions, we applied the extreme learning machine (ELM) technique to train a DEM. The ELM has a simple structure and a fast learning speed, while it is still able to obtain high levels of accuracy and it is therefore currently being researched in various fields [11–14]. In addition, studies are actively being conducted on advanced algorithms to improve the performance of the

conventional ELM or to resolve the complex problems beyond the abilities of the conventional ELM [15–22]. Kan et al. [9] and Hu and Yuan [10] estimated the DEM using a conventional ELM, but it was conducted on a very narrow area, and it was unable to obtain higher accuracy than the Delaunay triangulation-based interpolation method.

Even with the advantage of not needing large memory space, unless it has high levels of fitting accuracy that can replace the DEM, it cannot be applied to TRN. Moreover, when fitting a DEM as a complex model to enhance accuracy, it can preclude the real-time operation for navigation. Considering such restrictions, we used the generalized ELM autoencoder [15, 20] and multilayer radial basis function (RBF) [16, 18] to design a model that enhances the fitting accuracy of the ELM and guarantees real-time computing. We compared systems that used the proposed model and bilinear interpolation, which demonstrated the highest TRN performance in past studies, and our results verified that the proposed technique is the most suitable for TRN.

In the next section, we introduce existing studies on DEM fitting methods, namely, bilinear interpolation, the bi-spline neural network (B-spline NN), and the support vector machine for regression (SVMR). Then we introduce DEM learning techniques that use the conventional ELM and propose the multilayer RBF-based ELM (ML-RBF-ELM) to overcome the limitations of the conventional ELM. Furthermore, we compare the existing various DEM fitting methods and use of the proposed ML-RBF-ELM regression method and bilinear interpolation. Finally, we verify the design of the proposed technique by comparing it with the TRN performance.

2. Conventional DEM Fitting Method

There are different types of DEM database: the shuttle radar topography mission (SRTM) elevation database, digital terrain elevation data (DTED), etc. In this study, we used 10 m resolution DTED level 3 and 30 m resolution DTED level 2. There are various interpolation methods and machine learning regression techniques that can be used to obtain terrain elevation at the desired position using discontinuous DEM data. This section introduces previous studies that are aimed at obtaining continuous terrain elevation information.

2.1. Bilinear Interpolation. Interpolation methods frequently used to obtain continuous elevation information include Delaunay triangulation [9] and bilinear and bicubic interpolations. This study introduces bilinear interpolation, which is generally used in TRN systems. Bilinear interpolation is a method for linearly determining the linear distance of the four nearby DEM grid points, $Q_{11}(x_1, y_1)$, $Q_{12}(x_1, y_2)$, $Q_{21}(x_2, y_1)$, and $Q_{22}(x_2, y_2)$, when estimating the elevation value at any point $P(x, y)$ within the four points as shown in Figure 1.

If the terrain elevation value estimated at the point $P(x, y)$ is $f(x, y)$, the elevation values stored in DTED,

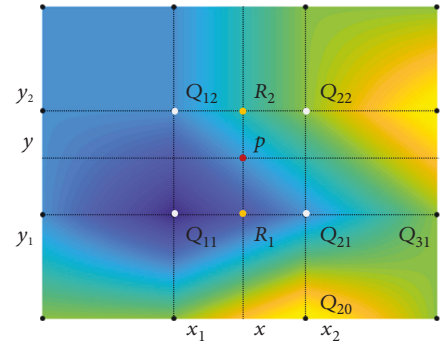

Figure 1: Bilinear interpolation scheme.

$h(Q_{11})$, $h(Q_{12})$, $h(Q_{21})$, and $h(Q_{22})$, can be used to compute as follows.

$$f(x, y) = \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \cdot \begin{bmatrix} h(Q_{11}) & h(Q_{12}) \\ h(Q_{21}) & h(Q_{22}) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}. \quad (1)$$

2.2. B-Spline Neural Network. As discussed by Liu et al. [6], the B-spline NN-based regression method was proposed to fit the 30 m resolution DEM. The method divides all the data into multiple bins to supplement the weaknesses of polynomial regression, and it fits the divided data with separate polynomial functions. The two adjacent polynomials' first and second differentials were made the same for smooth connection without making noncontinuous points in bins where division occurs. Liu et al. [6] designed the system as a single hidden layer where such spline basis functions are used for the layer's nodes and DEM data were learned as a neural network based on this. In this study, the B-spline NN was designed as 1971 hidden nodes using the spline basis function of order 3. This is compared with other fitting methods in the next section.

2.3. Support Vector Machine for Regression. Okwuashi and Ndehedehe [7] used SVMR to fit the DEM in the 225 m by 225 m field. The regression method using SVMR was compared with the artificial neural network and nearest neighbour method, and it showed the highest level of accuracy. SVMR uses the same principles as the SVM for classification with only a few minor differences. For regression, a margin of tolerance was set as the estimate of the SVMR, which would have already been determined from the problem [23]. SVMR should consider more design factors than SVM for classification. However, like SVM for classification, SVMR is learned to minimize the difference between the true value and the predicted value and maximize the margin between decision boundaries. For nonlinear SVMR, kernel functions map input data to higher dimensional feature space, simplifying it as a linear SVMR [23]. In our study, the Gaussian kernel function was used for the precision DTED regression model based on SVMR. In addition, the box constraint was used and set as 0.002178 to prevent

overfitting. The optimization issue of the SVMR model is usually solved with the Lagrange dual formulation for computational convenience, and when the Karush-Kuhn-Tucker (KKT) complementarity conditions are satisfied, the optimal solution can be obtained. In this study, such KKT tolerance was set as 0.5×10^{-4} and it was optimized using the iterative single data algorithm (ISDA), which updates a single multiplier per iteration.

3. DEM Fitting Method by ML-RBF-ELM

3.1. Conventional ELM. Since first being developed by Huang et al., the ELM has demonstrated good generalization ability and much faster learning than the backpropagation method of stochastic gradient descent and is therefore being studied in various fields. The ELM is a single hidden layer feedforward neural network. The input weights and hidden unit biases are chosen randomly and do not need to be adjusted [24]. As the output weights are analytically computed by the least squares method, this method provides a good generalization performance at an extremely fast learning speed. For M distinct training sample $\aleph = \{(x^{(m)}, d^{(m)}) | x^{(m)} \in R^N, d^{(m)} \in R^L, m = 1,\cdots,M\}$, the output of ELM with hidden layer nodes is as in [8, 9]:

$$\phi_k = \exp\left(-\beta_k \left\| x_i^{(m)} - a_{ki} \right\|^2\right), \quad k = 1, \cdots, K, i = 1, \cdots, N. \tag{2}$$

Here, ϕ_k is the activation function of the kth hidden node, which is modelled as a RBF. $x^{(m)} = \{x_i^{(m)}, i = 1,\cdots,N\}$ is the mth input data and N is set to 2 in this study. $x_1^{(m)}$ and $x_2^{(m)}$ are the normalized longitude and latitude, respectively. $d^{(m)}$ is the mth target (or true) data and means the normalized terrain height of the DEM in this study. α_{ki} and β_k are the center and width, respectively, of the kth RBF. By Huang et al. [24], the center and width are assigned randomly. $\|\bullet\|$ means the Euclidean distance. K is the number of hidden nodes and is set to 271 including a bias node in this study. The number of nodes was optimized to have the lowest training and test errors. The ELM can estimate any target function with zero error using equation (2). The relation is written as

$$HC = T, H = \begin{bmatrix} h\left(x^{(1)}\right) \\ \vdots \\ h\left(x^{(M)}\right) \end{bmatrix} = \begin{bmatrix} \phi_1\left(\alpha_1, \beta_1, x^{(1)}\right) & \cdots & \phi_K\left(\alpha_K, \beta_K, x^{(1)}\right) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \phi_1\left(\alpha_1, \beta_1, x^{(M)}\right) & \cdots & \phi_K\left(\alpha_K, \beta_K, x^{(M)}\right) & 1 \end{bmatrix}. \tag{3}$$

Here, $H[M \times K]$ is the hidden layer output matrix and $h(x^{(m)})$ is the hidden layer set of the mth sample data. The hidden layer set consists of K activation functions. $T[M \times L] = \{d^{(1)},\cdots,d^{(m)},\cdots,d^{(M)}\}^T$ is the target matrix of the ELM, and L is set to 1 because the normalized terrain elevation data is used as the true value of the training data. $C[K \times L]$ is the weight matrix of the output layer and the smallest norm least squares solution of equation (3) as follows:

$$C = H^{\dagger} T. \tag{4}$$

Here, $H^{\dagger}$ is the Moore-Penrose generalized inverse, and it is generally calculated as a singular value composition. In order to prevent the overfitting and improve the robustness, a ridge parameter is used [15, 16]. The optimization problem is described as

$$\min\left(\frac{1}{2} C^T C + \frac{1}{2\lambda} \sum_{m=1}^{M} \left\| \xi^{(m)} \right\|^2\right). \tag{5}$$

Here, λ is a regularization parameter and is set to 0.2. $\xi^{(m)}$ is the residual between the target value $d^{(m)}$ and estimated output value $y^{(m)}$ of the mth sample. According to KKT conditions, the solution of equation (5) is calculated as follows [9]:

$$C = H^{\dagger} T = \begin{cases} \left(H^T H + \lambda I\right)^{-1} H^T T, & M \geq K, \\ H^T \left(H H^T + \lambda I\right)^{-1} T, & M < K. \end{cases} \tag{6}$$

Here, I is a L-by-L identity matrix.

4. ML-RBF-ELM

The conventional ELM is easy to configure and the learning speed is fast, but it cannot be applied to fields that process big data with very large nonlinearity or require high levels of accuracy. To solve these problems, many studies are being actively conducted on various advanced ELM algorithms. In the constraint in which real-time computation must be guaranteed, three hidden layers were used as shown in Figure 2 to maximize the training and test accuracies. The first hidden layer was designed as an ELM-based autoencoder (ELM-AE), and the remaining hidden layers were designed as conventional ELMs. As the main difference between the ELM-AE and conventional ELM, the ELM is a supervised NN and the outputs are the target values but the ELM-AE is an unsupervised NN and the outputs are equal to the inputs [15].

The output weight matrix of the 1st hidden layer output C_1 is calculated as follows [15]:

$$C_1 = H_1^{\dagger} X = \begin{cases} \left(H_1^T H_1 + \lambda I\right)^{-1} H_1^T X, & M \geq K_1, \\ H_1^T \left(H_1 H_1^T + \lambda I\right)^{-1} X, & M < K_1. \end{cases} \tag{7}$$

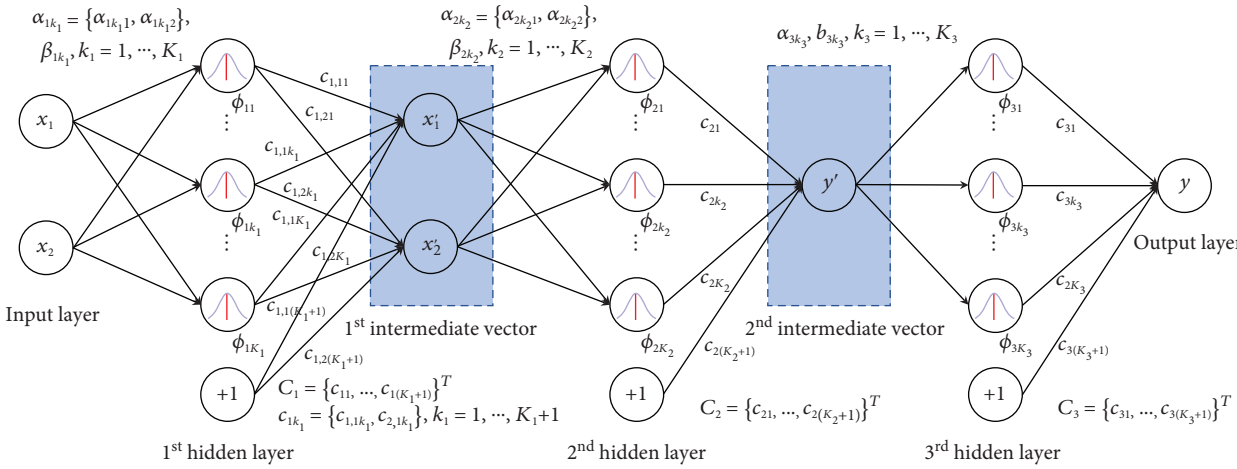

Figure 2: Proposed ML-RBF-ELM architecture.

(1) Prepare a training set $\aleph = \{(x^{(m)}, d^{(m)}) \mid x^{(m)} \in R^2, d^{(m)} \in R^1, m = 1,\cdots,M\}$ where $x^{(m)} = \{x_1^{(m)}, x_2^{(m)}\}$ is the input data with the normalized longitude and latitude and $d^{(m)}$ is the normalized terrain height in the DEM
(2) Set the activation function of the 1st hidden layer, $\Phi_1 = \{\phi_{11},\cdots,\phi_{1K_1}\}$, by equation (2) and assign the center $\alpha_{1k_1 i}$ and width β_{1k_1} of the RBF by K-means clustering
(3) Calculate the 1st hidden layer output matrix H_1 by equation (3)
(4) Calculate the output weight matrix of the 1st hidden layer, $C_1[(K_1+1)\times 2] = (H_1^T H_1 + \lambda I)^{-1} H_1^T X$ where $X[M\times 2] = \{x^{(1)},\cdots,x^{(M)}\}^T$
(5) Calculate the new input data of the 2nd hidden layer: $X'[M\times 2] = H_1 C_1$
(6) Set the activation function of the 2nd hidden layer, $\Phi_2 = \{\phi_{21},\cdots,\phi_{2K_2}\}$, by equation (2) and assign the center $\alpha_{2k_2 i}$ and width β_{2k_2} of the RBF by K-means clustering
(7) Calculate the 2nd hidden layer output matrix H_2 by equation (3)
(8) Calculate the output weight matrix of the 2nd hidden layer, $C_2[(K_2+1)\times 1] = (H_2^T H_2 + \lambda I)^{-1} H_2^T T$ where $T[M\times 1] = \{d^{(1)},\cdots,d^{(M)}\}^T$
(9) Calculate the new input data of the 3rd hidden layer: $y'[M\times 1] = H_2 C_2$
(10) Set the activation function of the 3rd hidden layer, $\Phi_3 = \{\phi_{31},\cdots,\phi_{3K_3}\}$, by equation (2) and assign the center $\alpha_{3k_3 i}$ and width β_{3k_3} of the RBF by K-means clustering
(11) Calculate the 3rd hidden layer output matrix H_3 by equation (3)
(12) Calculate the output weight matrix of the 3rd hidden layer, $C_3[(K_3+1)\times 1] = (H_3^T H_3 + \lambda I)^{-1} H_3^T T$
(13) Compute and verify the regression results, $y[M\times 1] = H_3 C_3$

Algorithm 1: Proposed ML-RBF-ELM algorithm.

Here, $X[M\times 2] = \{x^{(1)},\cdots,x^{(m)},\cdots,x^{(M)}\}^T$ and $x^{(m)} = \{x_i^{(m)}, i = 1,\cdots,N\}$. I is a N-by-N identity matrix and K_1 is the number of nodes in the 1st hidden layer. In this study, the ELM-AE was used to make the inputs more important in a rough terrain area than in a flat area. The 2nd and 3rd hidden layers shown in Figure 2 are similar to the conventional ELM. These layers were designed to represent the terrain height with refined accuracy. Unlike the conventional ELM, in the proposed ML-RBF-ELM method, the weights between the previous hidden layer and the current hidden layer were determined using K-means algorithms to enhance the training accuracy. The 1st and 2nd intermediate vectors are virtual vectors that represent the process in which the output weight matrices of the 1st and 2nd hidden layers explained in Algorithm 1 are learned. K-means clustering is used to partition M sample data into K clusters based on features in which each sample data belongs to the cluster with the nearest mean of a Gaussian function [25]. In this study, the mean and width values of the RBF nodes in the hidden layers are determined by the K-mean clustering based on the expectation-maximization algorithm [25]. The process of the proposed ML-RBF-ELM is given in Algorithm 1. The numbers of nodes in the three hidden layers, K_1, K_2, and K_3, are set to 384, 2301, and 193, respectively.

4.1. Comparison of Various DEM Fitting Methods. Figure 3 illustrates the DEM fitting area with a total area of about 689.70 km^2 for DTED level 3 and 4203.86 km^2 for DTED level 2. As mentioned above, we used 10 m resolution DTED level 3 and 30 m resolution DTED level 2 for the training and

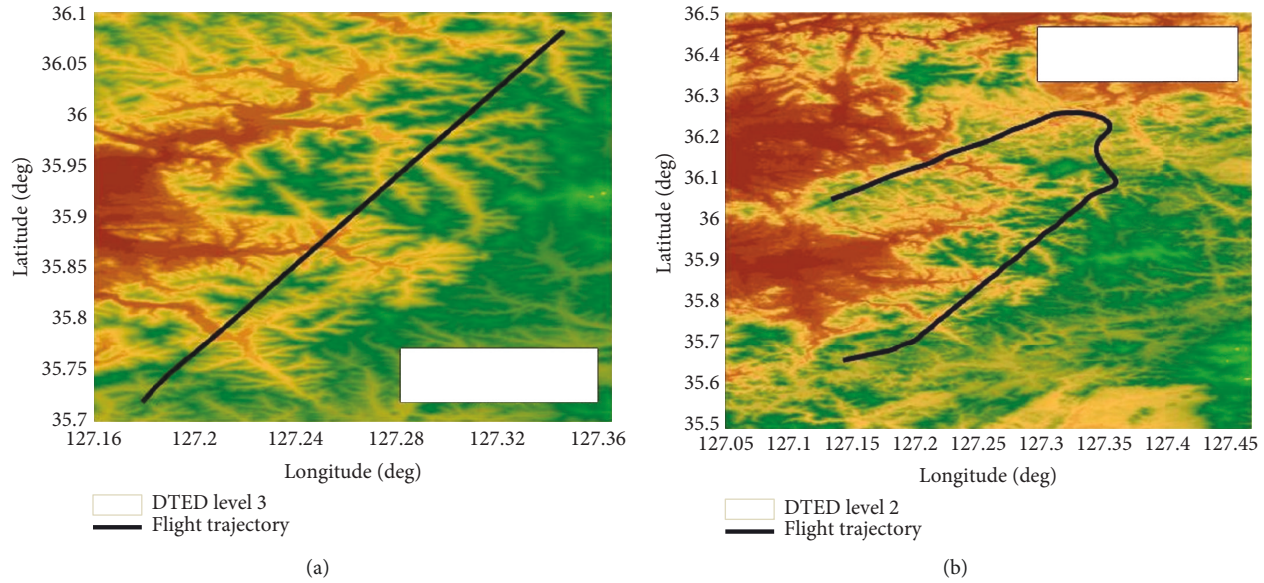

Figure 3: DEM fitting areas and flight trajectories. (a) DEM level 3 fitting area. (b) DEM level 2 fitting area.

Table 1: Results of various DEM fitting methods.

	Bilinear interpolation	B-spline NN	SVMR	Conventional ELM	ML-RBF-ELM
Number of hidden nodes	—	1971	—	271	3842301193
Training time	—	249.3 sec	3135.4 sec	8.10 sec	71.72 sec
Test time	—	99.8 μsec	115.3 sec	13.0 μsec	44.0 μsec
Training error	Approx. 1.02%	3.81%	8.76%	24.03%	0.98%
Test error	Approx. 1.01%	7.50%	8.64%	23.48%	1.10%

test data. The total obtainable number of sample data were 5391738 sets (161 MB) for level 3 and 5443200 sets (163 MB) for level 2. At a rate of 8 : 2, the sample data was used by dividing the training data (4313390 sets, 129 MB for level 3, and 4354560 sets, 130.4 MB for level 2) and test data (1078348 sets, 32 MB for level 3, and 1088640 sets, 32.6 MB for level 2). The sample data are acquired from the grid points such as Q_{11}, Q_{12}, Q_{21}, and Q_{22} in Figure 1. The learning was performed with an Intel® Core™ i7-6700K, CPU @4.00 GHz, RAM 16.0 GB computing environment. All simulations for the bilinear interpolation, B-spline NN, SVMR, conventional ELM, and the proposed ML-RBF-ELM were carried out in a MATLAB R2018a environment.

Table 1 compares the fitting accuracy and operation time according to the bilinear interpolation, B-spline NN, SVMR, conventional ELM, and the proposed ML-RBF-ELM for fitting DTED level 3. The fitting results of level 2 are also similar to those of Table 1. Each method was designed to maximize fitting accuracy. For the ML-RBF-ELM design, which was designed with three hidden layers, as shown in Table 1, the results confirmed that compared to the B-spline NN and conventional ELM designed as a single hidden layer, more nodes could be used in each hidden layer. Furthermore, we also confirmed that the ML-RBF-ELM, which was designed with three hidden layers, learned faster than the B-spline NN, which was designed as a single hidden layer. Likewise, it was evident that the ELM performed better in a shorter amount of time than the NN. The test time in Table 1 indicates the time needed for fitting one new sample to the terrain height. For the index to evaluate the fitting accuracy, the following mean value of the absolute percentage error was used. The value $\bar{E}$ is calculated as follows:

$$\bar{E} = \left[\sum_{m=1}^{M} \left| \frac{\left(y^{(m)} - d^{(m)} \right)}{d^{(m)}} \times 100 \right| \right] \times \frac{1}{M}. \tag{8}$$

Here, $y^{(m)}$ and $d^{(m)}$ are the estimated and target output value of the mth sample data, respectively. M is the total number of samples.

The training and test errors of the bilinear interpolation in Table 1 are approximate values. These values are about 25% of the absolute percentage errors when the terrain elevation value estimated at point Q_{21} was acquired from points Q_{11}, Q_{31}, Q_{20}, and Q_{22} in Figure 1. The conventional ELM had the smallest training and test time but the lowest fitting accuracy so it is not suitable for TRN. The SVMR did not satisfy the test time restrictions for real-time operation, so it is unsuitable. However, the proposed ML-RBF-

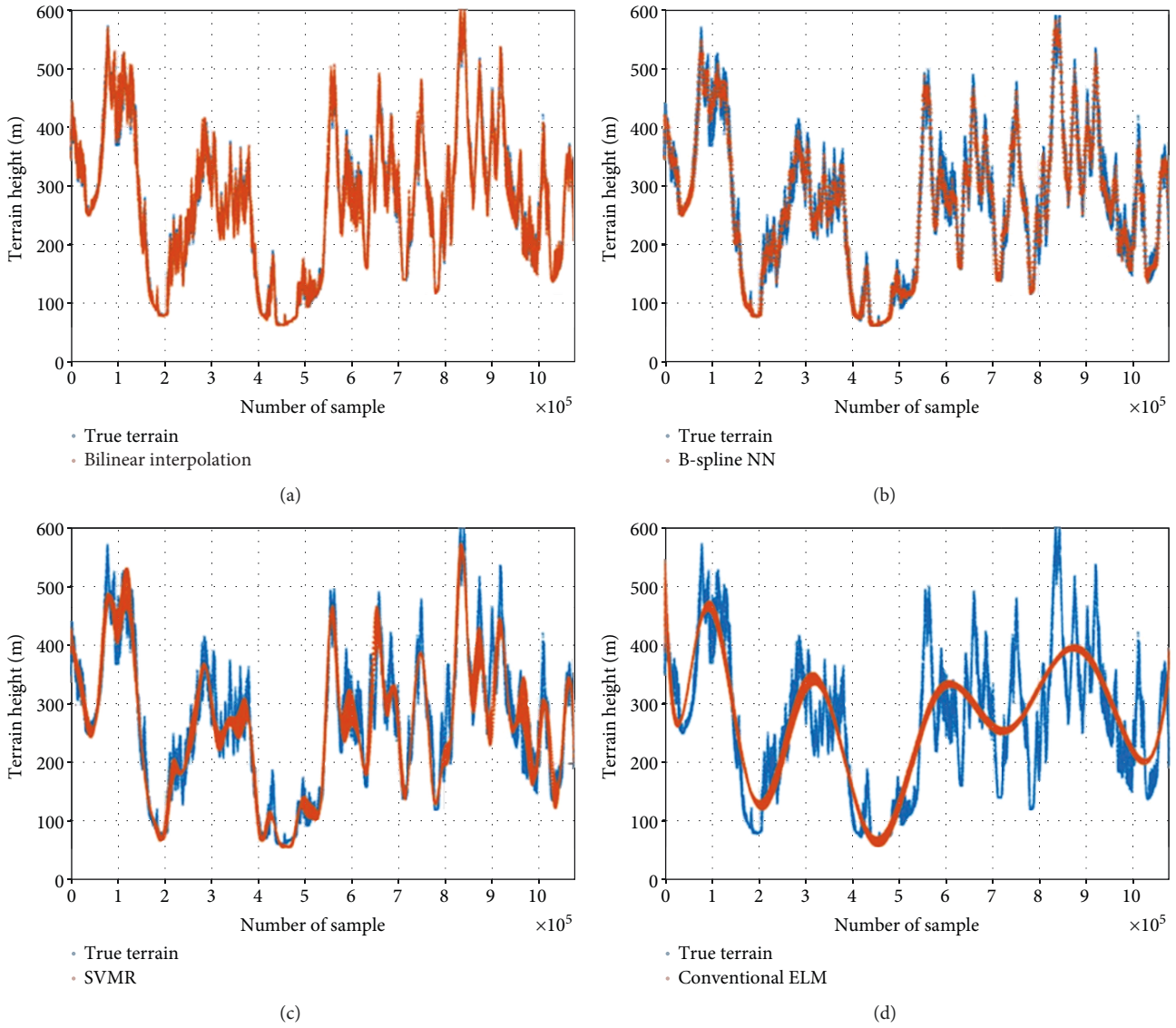

Figure 4: Results of the DEM fitting methods using DTED level 3. (a) Bilinear interpolation. (b) B-spline neural network. (c) SVMR. (d) Conventional ELM.

ELM had the best fitting accuracy and its short test time allowed for real-time operation, so we judged that it would be suitable in TRN. Figures 4 and 5 are illustrations of the terrain height fitting results for the test data. We found that previous research results did not reach the accuracy of bilinear interpolation, as shown in Figure 4. Despite the advantages of not requiring a large memory space to store the DEM and the time needed for loading, if fitting accuracy results cannot be obtained at the same levels of bilinear interpolation, then the performance of TRN cannot be guaranteed. On the other hand, as evident in Figure 5, the proposed ML-RBF-ELM demonstrates a fitting accuracy similar to the bilinear interpolation of Figure 4. To store the weights of the proposed ML-RBF-ELM, we need a 105 KB size memory. This is only about 1/1533 of the 161 MB memory size for DTED level 3 and about 1/1552 of the 163 MB memory size for DTED level 2.

5. Performance of TRN Using ML-RBF-ELM

5.1. BKF-Based TRN and INS/TRN Integration. We verified the ML-RBF-ELM by conducting TRN with the ML-RBF-ELM and bilinear interpolation, which is superior to the fitting methods explained in the previous section. TRN simulations were performed using the interferometric radar altimeter (IRA) measurements and a bank of Kalman filters (BKF), which were used in Heli/SITAN [26]. Figure 6 presents a schematic diagram of our system with an inertial navigation system (INS), IRA, barometer, DEM, TRN S/W, and INS/TRN S/W. The TRN progresses in two steps, the time propagation and the measurement update. If the IRA measurements do not satisfy validity check conditions; the TRN performs only the time propagation. To acquire all the navigation information, including the position, velocity, and attitude, the INS/TRN-integrated navigation is

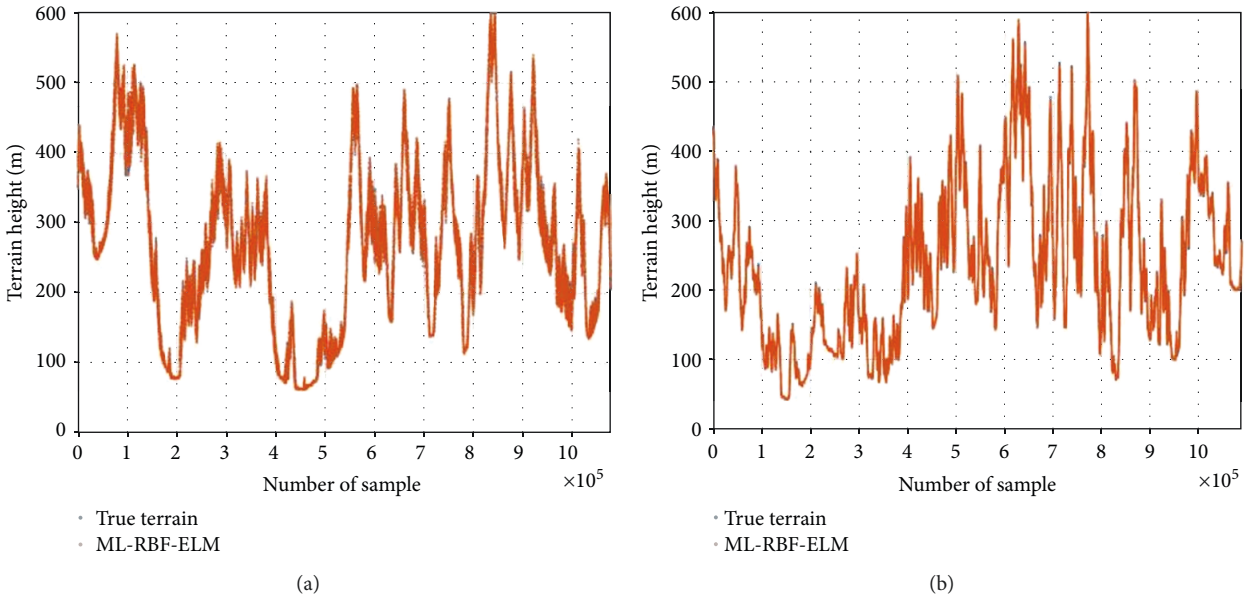

Figure 5: Fitting result of the proposed ML-RBF-ELM. (a) DTED level 3 result. (b) DTED level 2 result.

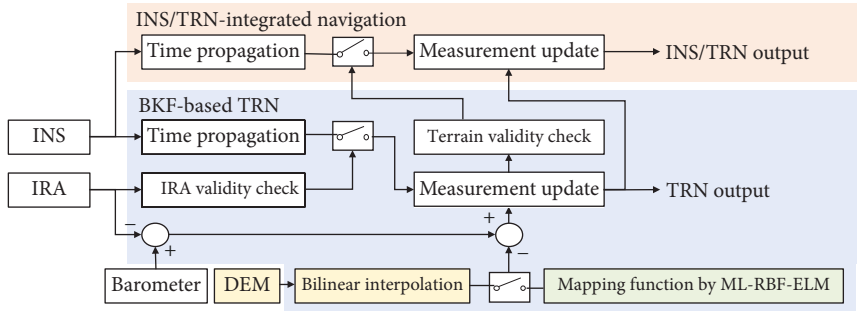

Figure 6: Schematic diagram of the TRN and INS/TRN system.

needed. Therefore, the loosely coupled INS/TRN-integrated navigation was designed with an extended Kalman filter (EKF) and uses the position estimated by the BKF-based TRN for measurement. Details of the INS/TRN-integrated navigation are given in section 4.3 of reference [27]. If the estimates of TRN do not satisfy terrain validity check conditions, the INS/TRN performs only the time propagation. Table 2 shows the simulation conditions and BKF design parameters. On June 21, 2018, and January 21, 2019, two captive flight tests were carried out on a CESSNA 172 Skyhawk. The INS/GNSS-integrated navigation data and IRA outputs were used as the true trajectory and measurements in this simulation, respectively. We conducted Monte Carlo simulations 100 times using bilinear interpolation and the proposed ML-RBF-ELM. The proposed ML-RBF-ELM was implemented by using a S/W without the DEM, as shown in Figure 6.

The BKF is a linear filter that uses the multiple-model adaptive estimation (MMSE) technique to acquire navigation information sequentially, even in the case of a large initial position error. A state variable of one-dimensional Kalman filters comprising the BKF is the vertical bias error, which gradually changes. The BKF forms grids that are centered on the INS/TRN position estimate and assigns a Kalman filter to each grid. In this study, 100×100 filters in the grid region corresponding to three times the initial position error were allocated. The time propagation for the jth

Table 2: Simulation conditions and BKF design parameters.

Parameter	Value
Initial covariance	15^2 m^2
Accelerometer bias	100 μg
Gyro bias	0.01 deg/hr
Gyro white noise	0.005 deg/$\sqrt{\text{hr}}$
Barometer bias	14 m
Barometer scale factor	0.2% of height
Barometer white noise	5 m
Process noise covariance	4^2 m^2
Measurement noise covariance	30^2 m^2
Update frequency	50 Hz

$(1 < j \le 100 \times 100)$ filter state variable x_{jk} and its covariance estimate p_{jk} assigned to each grid is described as

$$x_{jk} = x_{j(k-1)} + u_k,$$
$$p_{jk} = p_{j(k-1)} + Q_k \Delta t. \tag{9}$$

Here, Q_k is the process noise covariance and Δt is the sampling time. u_k is the upward velocity of the aircraft in the current step. In this study, the IRA was used to measure the angle perpendicular to the direction of flight and the range from the aircraft to the nearest terrain. It then converted these measurements to three-dimension position information. As shown in Figure 7, the relative position vector δx_{IRA} of the nearest point from the aircraft is calculated as follows:

$$\delta x_{\text{IRA}} = \begin{bmatrix} \delta\lambda_{\text{res}} \\ \delta\phi_{\text{res}} \\ h_{\text{res}} \end{bmatrix} = \begin{bmatrix} \rho \cos\xi \sin\alpha \sin\beta + \rho \sin\xi \cos\beta \\ \rho \cos\xi \sin\alpha \cos\beta - \rho \sin\xi \sin\beta \\ \rho \cos\xi \cos\alpha \end{bmatrix}. \tag{10}$$

Here, $\delta\lambda_{\text{res}}$, $\delta\phi_{\text{res}}$, and h_{res} are the relative distances in the directions of longitude, latitude, and height, respectively. ρ and ξ are the range and look angle output, respectively, of the IRA. The virtual pitch angle α and azimuth angle β of the zero Doppler line are determined by the velocity of the aircraft [27].

The jth filter gain k_{jk} and the measured values z_{jk} using the IRA measurement δx_{IRA} are as follows:

$$k_{jk} = p_{jk} \bullet \left(p_{jk} + R_k\right)^{-1},$$
$$z_{jk} = h_{jk}\left(\widehat{\lambda}_k + \delta\lambda_{\text{res}} R_{\text{ew}}^{-1}, \widehat{\phi}_k + \delta\phi_{\text{res}} R_{\text{ns}}^{-1}\right) + h_{\text{res}} - h_{\text{baro}}. \tag{11}$$

Here, $\widehat{\lambda}_k$ and $\widehat{\phi}_k$ are the longitude and latitude, respectively, estimated by INS/TRN and R_k is the measurement noise covariance. $\widehat{\lambda}_k + \delta\lambda_{\text{res}} R_{\text{ew}}^{-1}$ and $\widehat{\phi}_k + \delta\phi_{\text{res}} R_{\text{ns}}^{-1}$ are the longitude and latitude, respectively, at the nearest point T as shown in Figure 7. R_{ew} and R_{ns} represent the radius of the curvature of the Earth ellipsoid in the north-south and east-west, respectively. h_{jk} is the terrain height of the jth filter that is centered on the nearest point T. The terrain height is calculated using the bilinear interpolation from the stored DEM or the regression model by the proposed ML-RBF-ELM. h_{baro} is the aircraft altitude measured by the barometer. The weighted residual squares (WRS) of the jth filter at the kth step WRS_{jk} are calculated as follows [26]:

$$\text{WRS}_{jk} = \left(z_{jk} - x_{jk}\right)^2 \bullet \left(p_{jk} + R_k\right)^{-1}. \tag{12}$$

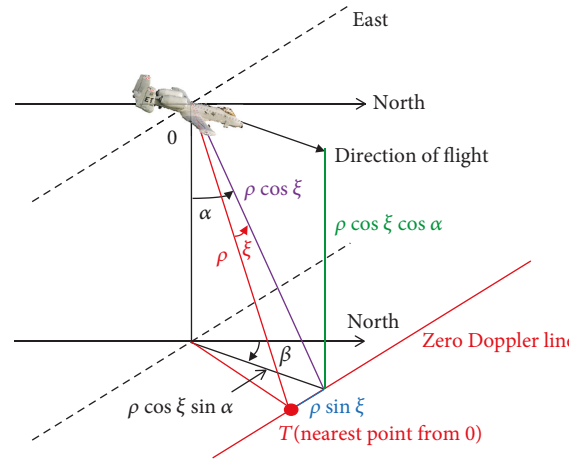

Figure 7: Relative position to the nearest point.

The WRS indicates how well each filter in the grids matches the designed model. The smoothed WRS (SWRS) is expressed in the same way as the moving average of the WRS as follows [26]:

$$\text{SWRS}_{jk} = \alpha \text{WRS}_{jk} + (1 - \alpha)\text{SWRS}_{j(k-1)}. \tag{13}$$

Here, $\text{SWRS}_{j0} = 1$ and α is the smoothing constant $(0.0 < \alpha \le 1.0)$ and is determined by calculating how fast the INS estimate gets out of the 3×3 filter exclusion region by the INS navigation error. In this study, α was set as 0.018. The estimated covariance NSWRS_i of each SWRS value in the 3×3 filter exclusion region centered on the grid with the minimum SWRS value SWRS_{min} among total grids is as follows:

$$\text{NWRS}_i = \text{SWRS}_i \text{SWRS}_{\text{min}}^{-1}, \quad 1 \le i \le 9. \tag{14}$$

Using the estimated covariance, the longitude and latitude $x_{\lambda k}$ and $x_{\phi k}$ estimated by the BKF are calculated as follows:

$$x_{\lambda k} = x_{\lambda 0} + (v_e \Delta t + p_e) R_{\text{ew}}^{-1},$$
$$x_{\phi k} = x_{\phi 0} + (v_n \Delta t + p_n) R_{\text{ns}}^{-1},$$
$$p_i = \exp\left(-0.5\text{NWRS}_i\right) \bullet \left[\sum_{i=1}^{9} \exp\left(-0.5\text{NWRS}_i\right)\right]^{-1}, \tag{15}$$
$$p_e = \sum_{i=1}^{9} p_{e_i} p_i,$$
$$p_n = \sum_{i=1}^{9} p_{n_i} p_i.$$

Here, $x_{\lambda 0}$ and $x_{\phi 0}$ are the initial longitude and latitude, respectively. p_{e_i} and p_{n_i} represent the east and north axis

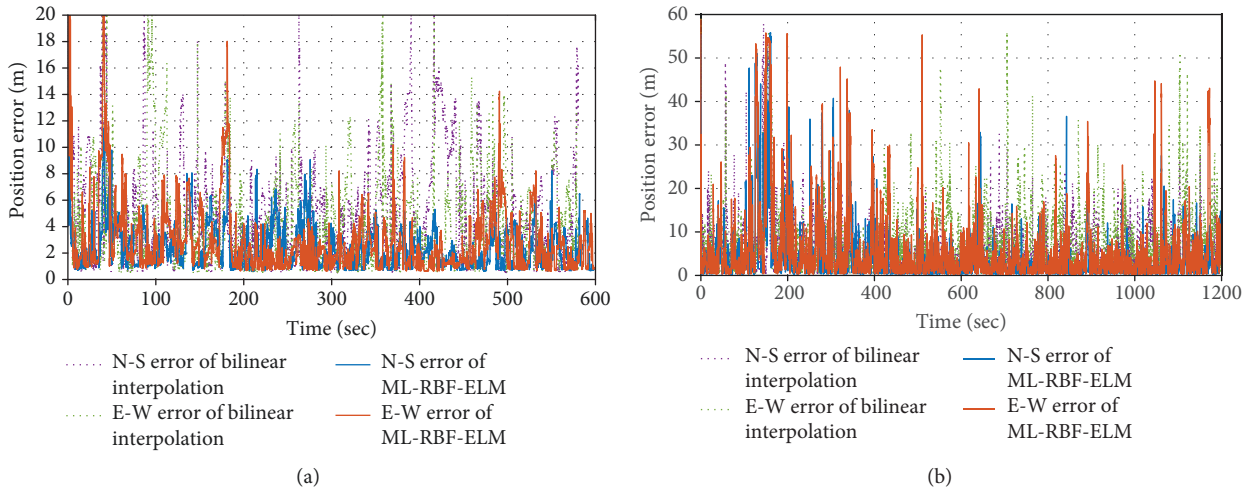

Figure 8: BKF-based TRN RMS errors. (a) DTED level 3 result. (b) DTED level 2 result.

position estimates of the ith filter in the exclusion region. The terrain validity check conditions in Figure 6 are as follows:

$$(\mathrm{SWRS}_{\mathrm{min}^*} - \mathrm{SWRS}_{\mathrm{min}})\mathrm{SWRS}_{\mathrm{min}}^{-1} > \mathrm{th}_r,$$
$$N_{\mathrm{MS}} > \mathrm{th}_u. \tag{16}$$

Here, $\mathrm{SWRS}_{\mathrm{min}^*}$ is the minimum SWRS value outside the beta area and N_{MS} represents how many times the filter retains the minimum SWRS value. th_r and th_u are their respective thresholds. Since the smallest value in the grid is $\mathrm{SWRS}_{\mathrm{min}}$, the terrain validity check conditions are always positive and the larger the difference, the rougher the terrain. The more often the above equation is satisfied, the more likely it is that the terrain is unique. The thresholds were each set as 5×10^{-4} and 2 for the rough area and 5×10^{-5} and 29 for the flat area through numerical simulations.

6. Simulation Results

Figure 8 compares the case of using the bilinear interpolation in the same simulation conditions (dotted lines) and the case of using the regression model fitted according to the proposed ML-RBF-ELM (solid lines). The position root mean square (RMS) errors in the north-south and east-west axes were smaller for the BKF-based TRN using the ML-RBF-ELM. Figure 8(a) shows such characteristics more clearly between 400 seconds and 500 seconds. Figure 8(b) also shows improved performance from 500 seconds onward. The 100 Monte Carlo simulation results of INS/TRN-integrated navigation, which uses the latitude and longitude information estimated by TRN as measurements, are shown in Figure 9. The performance difference between the bilinear interpolation and the ML-RBF-ELM in INS/TRN is smaller compared to TRN. However, it is evident that compared to using bilinear interpolation in Figures 9(a) and 9(c), using ML-RBF-ELM carries out TRN and INS/TRN more stably as shown in Figures 9(b) and 9(d). If the terrain height changes smoothly, the error of bilinear interpolation is small. However, the bilinear interpolation error may be bigger in terrains with higher nonlinearity. Meanwhile, the proposed ML-RBF-ELM is less affected by such terrain roughness or uniqueness and so the change in the overall RMS error may be smaller than bilinear interpolation. Table 3 shows the values of converting the horizontal position RMS errors of TRN and INS/TRN to circular error probability (CEP). Comparing TRN with the bilinear interpolation, TRN with the ML-RBF-ELM improved the result by about 2.335 m CEP in DTED level 3 and about 1.402 m CEP in DTED level 2. In other words, using ML-RBF-ELM, it is possible to fit a highly accurate regression model without the DEM. Moreover, TRN and INS/TRN using the proposed method achieved stable performance.

7. Conclusions

This study introduced DEM fitting methods, a key technology of TRN, that can be used as an alternative in environments where GNSS is unavailable. The DEM is a database that stores the terrain's elevation data at a constant resolution. Therefore, large amounts of memory are needed to use the high-resolution DEM. Furthermore, the various interpolation methods for obtaining continuous elevation information from the discontinuous DEM includes fitting errors. Accordingly, there are currently many ongoing studies on DEM learning through various machine learning techniques. However, these efforts have only produced results on DEM learning in low-resolution or narrow fields and they have yet to achieve stable performance for TRN using these fitting methods. In this study, we used the ML-RBF-ELM to fit a 10 m resolution DEM in an area of 680.70 km^2 and a 30 m resolution DEM in an area of 4203.86 km^2. The proposed regression model by the ML-RBF-ELM operates more quickly than the previous machine learning regression methods for the DEM and demonstrates similar fitting results with bilinear interpolation. Furthermore, we used BKF-based TRN, which showed improved TRN performance results over using the bilinear interpolation. Thus, our study verified that the proposed ML-RBF-ELM can make real-time operation of navigation

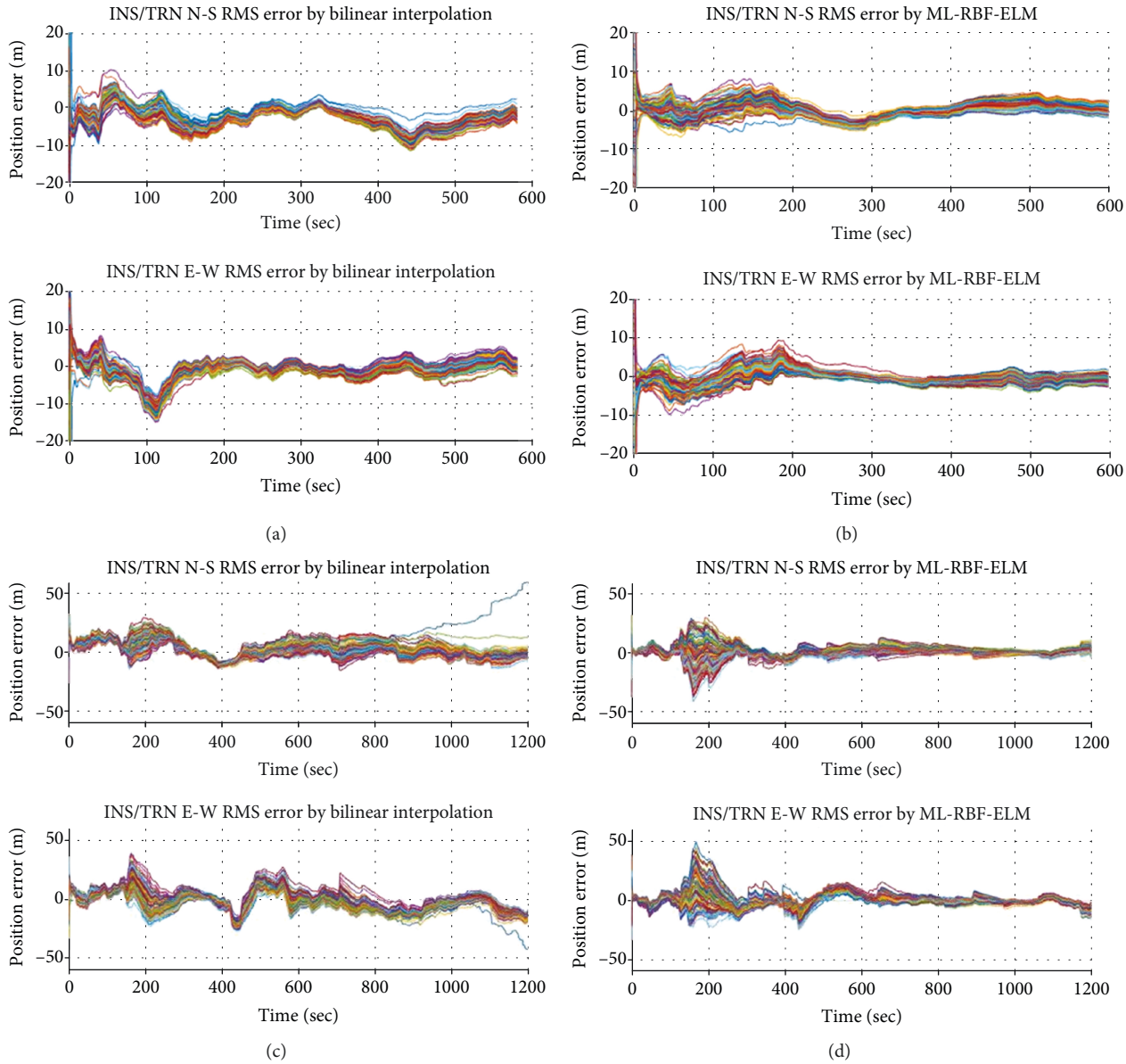

Figure 9: INS/TRN RMS errors. (a) Bilinear interpolation with DTED level 3. (b) ML-RBF-ELM with DTED level 3. (c) Bilinear interpolation with DTED level 2. (b) ML-RBF-ELM with DTED level 2.

Table 3: Comparison of TRN and INS/TRN RMS errors.

DTED level	RMS errors	Bilinear interpolation	ML-RBF-ELM
3	TRN	7.465 m CEP	5.130 m CEP
3	INS/TRN	4.126 m CEP	3.793 m CEP
2	TRN	12.540 m CEP	11.138 m CEP
2	INS/TRN	9.891 m CEP	6.079 m CEP

and is suitable to TRN. It is clear that the proposed technology can be used by low-priced small unmanned aerial vehicles to execute TRN in environments where GNSS is not possible. Moreover, the proposed ML-RBF-ELM can be used in the field of simultaneous localization and map building (SLAM) because of the small training time.

Data Availability

The DTED level 3 full sample data and terrain reference navigation S/W used to support the findings of this study are not freely available, because of our institutional security policies. But the edited sample data, MRBF-ELM fitting S/W, and the simulation result data used to support the findings of this study are included within the supplementary information files (available here). The edited sample data size is smaller than original sample data size.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Supplementary Materials

The list of the supplementary information files is as follows. The edited sample data: "TestDBData3.dat," "TrainDBData3.dat" in "TrainDB_ML_RBF_ELM_Using AE2_180919" Folder MRBF-ELM fitting S/W: "TrainDB_ML_RBF_ELM_Using AE2_180919" Folder the simulation result data: "TRNSimulationOutput_180912" folder, and "CompareMatchingResults_180909" folder. *(Supplementary Materials)*

References

[1] M. S. Grewal, L. R. Weill, and A. P. Andrews, "GNSS/INS integration," in *Global Positioning Systems, Inertial Navigation, and Integration*, pp. 382–424, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2nd edition, 2006.

[2] H. C. Jeon, W. J. Park, and C. G. Park, "Grid design for efficient and accurate point mass filter-based terrain referenced navigation," *IEEE Sensors Journal*, vol. 18, no. 4, pp. 1731–1738, 2018.

[3] F. C. Teixeira, J. Quintas, P. Maurya, and A. Pascoal, "Robust particle filter formulations with application to terrain-aided navigation," *International Journal of Adaptive Control and Signal Processing*, vol. 31, no. 4, pp. 608–651, 2017.

[4] A. Murangira, C. Musso, and K. Dahia, "A mixture regularized rao-blackwellized particle filter for terrain positioning," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 4, pp. 1967–1985, 2016.

[5] J. Melo and A. Matos, "Survey on advances on terrain based navigation for autonomous underwater vehicles," *Ocean Engineering*, vol. 139, pp. 250–264, 2017.

[6] C. Liu, H. Wang, and P. Yao, "On terrain-aided navigation for unmanned aerial vehicle using B-spline neural network and extended Kalman filter," in *Proceedings of the 2014 IEEE Chinese Guidance, Navigation and Control Conference*, pp. 2258–2263, Yantai, China, 2014.

[7] O. Okwuashi and C. Ndehedehe, "Digital terrain model height estimation using support vector machine regression," *South African Journal of Science*, vol. 111, no. 9/10, pp. 148–152, 2015.

[8] C.-W. T. Yeu, M. H. Lim, G.-B. Huang, A. Agarwal, and Y. S. Ong, "A new machine learning paradigm for terrain reconstruction," *IEEE Geoscience and Remote Sensing Letters*, vol. 3, no. 3, pp. 382–386, 2006.

[9] E. M. Kan, M. H. Lim, Y.-S. Ong, A. H. Tan, and S. P. Yeo, "Extreme learning machine terrain-based navigation for unmanned aerial vehicles," *Neural Computing and Applications*, vol. 22, no. 3-4, pp. 469–477, 2013.

[10] X. Hu and Y. Yuan, "Deep-learning-based classification for DTM extraction from ALS point cloud," *Remote Sensing*, vol. 8, no. 9, p. 730, 2016.

[11] Y. Jian, D. Huang, J. Yan et al., "A novel extreme learning machine classification model for e-nose application based on the multiple kernel approach," *Sensors*, vol. 17, no. 6, article 1434, 2017.

[12] V. M. Janakiraman, X. L. Nguyen, and D. Assanis, "Stochastic gradient based extreme learning machines for stable online learning of advanced combustion engines," *Neurocomputing*, vol. 177, pp. 304–316, 2016.

[13] Z. Wang, S. Yoon, and D. S. Park, "A novel visual tracking system with adaptive incremental extreme learning machine," *KSII Transactions on Internet and Information Systems*, vol. 11, no. 1, pp. 451–465, 2017.

[14] P. Yuan, D. Chen, T. Wang et al., "A compensation method based on extreme learning machine to enhance absolute position accuracy for aviation drilling robot," *Advances in Mechanical Engineering*, vol. 10, no. 3, 11 pages, 2018.

[15] K. Sun, J. Zhang, C. Zhang, and J. Hu, "Generalized extreme learning machine autoencoder and a new deep neural network," *Neurocomputing*, vol. 230, pp. 374–381, 2017.

[16] N. Zhang, S. Ding, and J. Zhang, "Multi layer ELM-RBF for multi-label learning," *Applied Soft Computing*, vol. 43, pp. 535–545, 2018.

[17] Y. Zhou, J. Peng, and C. L. P. Chen, "Extreme learning machine with composite kernels for hyperspectral image classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2351–2360, 2015.

[18] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 809–821, 2016.

[19] G.-B. Huang, Z. Bai, L. L. C. Kasun, and C. M. Vong, "Local receptive fields based extreme learning machine," *IEEE Computational Intelligence Magazine*, vol. 10, no. 2, pp. 18–29, 2015.

[20] J. Tapson, P. D. Chazal, and A. V. Schaik, "Explicit computation of input weights in extreme learning machines," in *Proceedings of ELM-2014 Volume 1. Proceedings in Adaptation, Learning and Optimization, vol 3*, J. Cao, K. Mao, E. Cambria, Z. Man, and K. A. Toh, Eds., pp. 41–49, Springer, Cham, 2014.

[21] I. Chaturvedi, E. Ragusa, P. Gastaldo, R. Zunino, and E. Cambria, "Bayesian network based extreme learning machine for subjectivity detection," *Journal of the Franklin Institute*, vol. 355, no. 4, pp. 1780–1797, 2018.

[22] K. Pasupa and W. Kudisthalert, "Virtual screening by a new clustering-based weighted similarity extreme learning machine approach," *PLoS One*, vol. 13, no. 4, article e0195478, 2018.

[23] F. Parrella, "Online support vector regression," Master's Thesis, Department of Information Science, University of Genoa, Italy, 2007.

[24] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[25] Z. Zhang, J. Zhang, and H. Xue, "Improved K-means clustering algorithm," in *2008 Congress on Image and Signal Processing*, pp. 169–172, Sanya, Hainnan, China, May 2008.

[26] J. Hollow, "HELI/SITAN: a terrain referenced navigation algorithm for helicopters," in *IEEE Symposium on Position Location and Navigation. A Decade of Excellence in the Navigation Sciences*, pp. 616–625, Las Vegas, NV, USA, March 1990.

[27] J. Lee and H. Bang, "Radial basis function network-based available measurement classification of interferometric radar altimeter for terrain-aided navigation," *IET Radar, Sonar & Navigation*, vol. 12, no. 9, pp. 920–930, 2018.