Sensors and Materials, Vol. 29, No. 9 (2017) 1291–1303
MYU Tokyo

S & M 1423

A Prediction Method for Deck Motion of Aircraft Carrier Based on Particle Swarm Optimization and Kernel Extreme Learning Machine

Xixiang Liu,[1,2*] Qiming Wang,[1,2] Yongjiang Huang,[1,2] Qing Song,[1,2] and Liye Zhao[1,2]

[1]School of Instrument Science & Engineering, Southeast University, Nanjing 210096, China
[2]Key Laboratory of Micro-inertial Instrument and Advanced Navigation Technology,
Ministry of Education, Nanjing 210096, China

(Received March 1, 2017; accepted August 23, 2017)

Keywords: prediction of deck motion, extreme learning machine, support vector machine, particle swarm optimization

The prediction of deck motion is an effective and potential means of improving the landing/take-off safety of carrier-based aircraft using current and historical deck-motion measurements when deck motion in six degrees of freedom cannot be effectively controlled or restrained. The prediction models of deck motion should have excellent nonlinear fitting ability to cope with the deck-motion characteristics of randomness and nonlinearity caused by waves and wind; and should not use heavy computation to fulfill the requirement of real-time prediction for deck motion. It is generally believed that classical feed-forward neural networks, such as the back-propagation (BP) network, have excellent nonlinear fitting ability but suffer from slow training processes and reduced local optimum, thus failing to satisfy the requirements of real-time and high accuracy for deck-motion prediction. In addition, the extreme learning machine (ELM) is easy to train but it is difficult for ELM to determine the number of hidden layer nodes; an incorrect number of hidden layer nodes will introduce poor stability and generalization ability. To fulfill the requirements of deck motion prediction, a prediction method based on ELM, support vector machine, and particle swarm optimization [particle swam optimization kernel extreme learning machine (PSO-KELM)] is designed. In this method, the fundamental structure of the ELM is used and the kernel function from the support vector machine (SVM) is introduced to replace the hidden function in ELM. Further aiming at the acquisition of optimal parameters, including the penalty coefficient and kernel parameters for the kernel function, autoadaptive particle swarm optimization is adopted. Simulation results indicate that a prediction method based on PSO-KELM has the advantages of a simple structure, fast training speed, and powerful generalization ability, and thus can satisfy the requirements of real-time and high-accuracy deck-motion prediction. Compared with the prediction data from BP and the ELM, high-precision prediction data can be obtained with PSO-KELM. PSO-KELM has a significantly reduced training time compared with BP.

[*]Corresponding author: e-mail: scliuseu@163.com

http://dx.doi.org/10.18494/SAM.2017.1588

1. Introduction

As a moving platform for aircraft at sea, aircraft carriers have been widely valued for the powerful offensive/defensive ability of carrier-based aircraft. During the process of aircraft landing, aircraft carriers should be in a stable condition and the position of the ideal landing point should be known. Because of excitation by wind and waves, deck motion in six degrees of freedom is generated. Deck motion, especially heave and the coupling motion caused by pitch and the lever arm, will make the ideal landing point of aircraft variable in three-dimensional space; thus, the landing safety of aircraft will be significantly decreased.(1–4) In this paper, the lever arm refers to the geometric distance between the ideal landing point and the swinging center of an aircraft carrier.

Using the active control method to suppress the six-degree-of-freedom motion of ships may be an effective means of improving the landing safety of carrier-based aircraft.(5) Unfortunately, there is no effective method to restrain or control wave-induced motion except yaw and roll, which can be partially restrained through the combined operation of the fin and rudder.

When swaying motion, especially heave and pitch, cannot be restrained, the prediction of deck motion is a potential means of improving the safety of aircraft. The prediction of deck motion is based on extreme short-term prediction technologies for ships, in which historical deck-motion data, including current data, is used to predict the motion of approximately the next 10 s.(6) It is believed that if the motion of the next 10–13 s can be accurately predicted, the landing track can be adjusted in real time, and thus the safety of aircraft can be significantly improved.(7)

Currently, technologies for predicting extreme short-term conditions of ships have been widely studied from such aspects as ship safety and navigation performance, and many effective accomplishments, such as statistical prediction, convolution prediction, and Kalman prediction, have been achieved.(8–11) These methods are all based on an accurate mechanical model or accurate statistical parameters. However, in engineering, it is difficult to acquire an accurate input/output mechanical model for carriers because of the overcomplex relationship between wind/waves and a ship. Also it is difficult to acquire accurate statistical parameters because large ships always sail in a large expanse of sea over a long period and experience the complexity of the sea. With the aim of realizing deck-motion prediction, intelligent methods such as grey prediction and neural network prediction are introduced.(12–14) Among these methods, the grey prediction method requires the predicted object to conform to the exponential function law, but it is an ideal assumption and therefore difficult to realize. When this assumption cannot be satisfied, fitting and generalization ability will degrade. In contrast, back-propagation (BP) networks are believed to be highly capable of fitting any nonlinear function and are an effective method for solving nonlinear prediction problems. Unfortunately, classical neural networks easily suffer from large amounts of calculation, slow training, and a local minimum problem; thus, they cannot meet the requirements of real-time calculation and high accuracy required for deck-motion prediction.(15) To make matters worse, the deck motion of aircraft carriers is always measured with the Strapdown Inertial Navigation System (SINS). SINS has a high update frequency (≥100 Hz). This means that the amount of input for the prediction model is very large and the calculation is very heavy.

In 2004, the extreme learning machine (ELM) was first proposed by Huang and co-authors. It was also based on a feed-forward neural network structure but a new and single hidden layer was adopted.(16–18) Unlike the classical neural network, only the number of hidden nodes in ELM needs to be set in advance. During training, input weights and thresholds are randomly assigned, and output weights are solved using these samples for training and the theory of the generalized inverse matrix. In this paper, a prediction method based on ELM is introduced to cope with deck-motion prediction that has the characteristics of a simple structure, high training speed, powerful fitting ability, and a global optimum. However, when using ELM, there is no mature method of selecting the number of hidden nodes which is a decisive factor deciding prediction accuracy. Also this method suffers from model instability and a low generalization ability caused by random mapping in the hidden layer.(19) In order to solve these problems, a new network named kernel extreme learning machine (KELM) was introduced in Ref. 16. In KELM, the kernel function of the support vector machine (SVM) is introduced to act as the hidden layer in ELM; thus, the advantages of ELM and the SVM can be combined. In KELM, there are also difficulties in setting the penalty coefficient and kernel parameter, which govern the prediction accuracy. With the aim of acquiring optimal parameters for KELM, autoadaptive particle swarm optimization (PSO) is further introduced. Simulation results indicate that the prediction model based on PSO-KELM shows significant advantages compared with BP and ELM.

The rest of this paper is organized as follows. In Sect. 2, models describing deck motion are studied and the method for simulating deck motion is introduced; the algorithm for ELM and the improved method of KELM are studied in Sect. 3; autoadaptive PSO is further introduced for parameter optimization in Sect. 4; simulations are demonstrated in Sect. 5; finally the conclusions are given in Sect. 6.

2. Models for Deck Motion of Aircraft Carrier

Models describing ship motion or deck motion are widely studied. However, in engineering it is difficult to use a unified model describing deck motion because of the complexity of sea conditions and different ship types. When studying the ship-swaying model, it is always studied under a fixed sea condition for one ship type. In this paper, deck motion of an aircraft carrier with a tipped bottom in general sea conditions is studied.

Currently, there are two widely used models describing deck motion: the random model based on the power spectrum and the combined model based on the superimposition of sine waves.(20,21) The latter is widely accepted and has been used since the 1950s. In the combined model, waves in mature periods are regarded as a stable random process and can be reconstructed with independent random variables; under a fixed sea condition, ship motion is regarded as a process with a narrow band stochastic process and can be simplified by superimposing with some sine waves. In Ref. 21, deck-motion models for pitch, roll, yaw, and heave for a moderate-size aircraft carrier under a moderate sea condition with a velocity of 30 knots are given as

$$\begin{cases} P = 0.5\sin(0.6t) + 0.3\sin(0.63t) + 0.25, \\ R = 2.5\sin(0.5t) + 3.0\sin(0.52t) + 0.5, \\ H = 0.25\sin(0.7t) + 0.5\sin(0.1t), \\ T_z = 1.22\sin(0.6t) + 0.3\sin(0.2t), \end{cases} \tag{1}$$

where P, R, and H are pitch, roll, and yaw motion with the unit of degrees, respectively; T_z is the heave motion with the unit of meters; t is the running time in seconds.

During deck-motion prediction, acquiring historical deck-motion data is a precondition. In the sea, there is no direct reference position; thus, it is difficult to measure deck motion with indirect methods, such as light and radar. Direct measuring methods are preferred. SINS is always used to measure deck motion. When SINS is installed at or near the ideal landing point, pitch, roll, and heave relative to the earth's horizontal plane and yaw relative to north can be calculated with angle velocity and accelerations measured with an inertial measurement unit and SINS navigation algorithm. With only SINS, navigation data will drift with time, but if aided navigation systems such as GNSS are adopted, errors from SINS/GNSS can be dealt with as white noise.

In this paper, the motion described by Eq. (1) represent the ideal deck motion without any error/noise. Random noise is added to this ideal motion, and the resulting motion containing noise is used as the measured deck motion by SINS.

3. Prediction Model Based on KELM

ELM is a new type of feed-forward neural network with a single hidden layer. Compared with those in classical feed-forward neural networks, only the number of hidden nodes need be determined in advance in ELM, and input weights and thresholds are randomly assigned, then the optimal solution can be obtained. The structure of ELM is as follows.(16–18)

As illustrated in Fig. 1, the function of the hidden layer is denoted as $g(x)$ and the number of hidden nodes is denoted as L. When a set of samples ($\boldsymbol{x}_i$, $\boldsymbol{t}_i$), $1 \leq i \leq N$, is given, where $\boldsymbol{x}_i = [x_{i1}, x_{i2}, ..., x_{in}]^{\mathrm{T}} \in \mathbf{R}^n$ denotes the input, $\boldsymbol{t}_i = [t_{i1}, t_{i2}, ..., t_{im}]^{\mathrm{T}} \in \mathbf{R}^m$ denotes the target, and $\boldsymbol{\beta}$ denotes the connection weight between the hidden and output layers, and $\boldsymbol{\beta}$ can be expressed as

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1m} \\ \beta_{21} & \beta_{22} & \cdots & \beta_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ \beta_{L1} & \beta_{L2} & \cdots & \beta_{Lm} \end{bmatrix}_{L\times m}, \tag{2}$$

where $\boldsymbol{\beta}_{im}$ denotes the connection weight between the ith neuron of the hidden layer and the jth node of the output layer.

The model of ELM can be constructed as

$$\boldsymbol{T} = [\boldsymbol{t}_1, \boldsymbol{t}_{2,} \ldots, \boldsymbol{t}_N]_{m\times N},\ f(\boldsymbol{x}_j) = \begin{bmatrix} \sum_{i=1}^{L} \beta_{i1} g\left(\boldsymbol{\omega}_i, \boldsymbol{b}_i, \boldsymbol{x}_j\right) \\ \sum_{i=1}^{L} \beta_{i2} g\left(\boldsymbol{\omega}_i, \boldsymbol{b}_i, \boldsymbol{x}_j\right) \\ \cdots \\ \sum_{i=1}^{L} \beta_{im} g\left(\boldsymbol{\omega}_i, \boldsymbol{b}_i, \boldsymbol{x}_j\right) \end{bmatrix} = \boldsymbol{t}_j,\ j = 1, 2, \ldots, N, \tag{3}$$

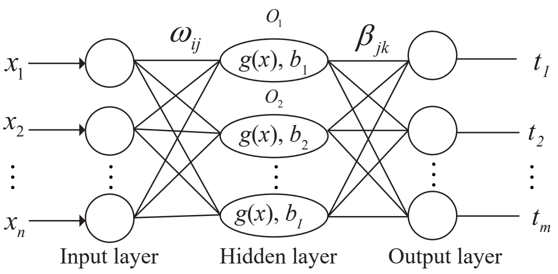

Fig. 1. The structure of ELM.

where $\boldsymbol{\omega}_i$ denotes the connection weight between the hidden and input layers, $\boldsymbol{b}_i$ denotes the threshold of neurons in the hidden layer, $g(\boldsymbol{\omega}_i, \boldsymbol{b}_i, \boldsymbol{x}_j)$ is the output of the ith neuron with the input of $\boldsymbol{x}_j$, and $\boldsymbol{T}$ is the output matrix of samples. Furthermore, Eq. (3) can be rewritten as

$$\boldsymbol{H\beta} = \boldsymbol{T}', \tag{4}$$

where $\boldsymbol{T}'$ is the inverse of $\boldsymbol{T}$, and $\boldsymbol{H}$ is the output matrix of the hidden layer expressed as

$$\begin{aligned}\boldsymbol{H} &= [\omega_1, \omega_2, \ldots, \omega_L, \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_L, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N] \\ &= \begin{bmatrix} g(\omega_1, \boldsymbol{b}_1, \boldsymbol{x}_1) & g(\omega_2, \boldsymbol{b}_2, \boldsymbol{x}_1) & \cdots & g(\omega_L, \boldsymbol{b}_L, \boldsymbol{x}_1) \\ g(\omega_1, \boldsymbol{b}_1, \boldsymbol{x}_2) & g(\omega_2, \boldsymbol{b}_2, \boldsymbol{x}_2) & \cdots & g(\omega_L, \boldsymbol{b}_L, \boldsymbol{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ g(\omega_1, \boldsymbol{b}_1, \boldsymbol{x}_N) & g(\omega_2, \boldsymbol{b}_2, \boldsymbol{x}_N) & \cdots & g(\omega_L, \boldsymbol{b}_L, \boldsymbol{x}_N) \end{bmatrix}_{N\times L} .\end{aligned} \tag{5}$$

With the aim of acquiring the lowest structural and empirical risks, the training target function of ELM is defined as

$$\begin{cases} \min : \dfrac{1}{2}\,||\beta||^2 + \dfrac{1}{2}C\displaystyle\sum_{i=1}^{N}\xi_i^2, \\ \text{s.t.} : \boldsymbol{t}_i - f(\boldsymbol{x}_i) = \xi_i, \end{cases} \tag{6}$$

where $\dfrac{1}{2}\,||\beta||^2$ and $\dfrac{1}{2}\displaystyle\sum_{i=1}^{N}\xi_i^2$ denote structural risk and empirical risk, respectively, ξ_i is the difference between the theoretical output and actual output, and is defined as the penalty coefficient that is used to weight the ratio between structural risk and empirical risk.

Applying the Karush–Kuhn–Tucker (KKT) condition, the output weight matrix of $\boldsymbol{\beta}$ can be expressed as

$$\boldsymbol{\beta} = \boldsymbol{H}^{\mathrm{T}}\left(\frac{1}{C}\boldsymbol{I}_N + \boldsymbol{HH}^{\mathrm{T}}\right)^{-1}\boldsymbol{T}, \tag{7}$$

where $\boldsymbol{I}_N$ is a unit matrix.

As illustrated above, the ELM algorithm has the advantages of a simple structure, requiring a small amount of calculation. However, there are two defects as follows. Firstly, the generalization ability of ELM is determined by the number of hidden nodes. Unfortunately, there is no established method of selecting this number, thus the generalization ability is unstable. Secondly, the input weight and threshold are assigned randomly, which introduces random fluctuation of the output and leads to the instability of ELM and unsatisfactory generalization. To overcome these shortcomings, the kernel function of the SVM is introduced in ELM, which is then renamed KELM, in which kernel mapping is used as a substitute for random mapping.

If the matrix $\boldsymbol{H}$ in the hidden layer is unknown, the kernel matrix can be defined as follows in ELM under the condition described by Mercer.[16]

$$\begin{cases} \Omega_{ELM} = HH^{\mathrm{T}} \\ \Omega_{i,j} = h(x_i)\,h\left(x_j\right) = K\left(x_i, x_j\right) \end{cases} \tag{8}$$

Then the output of KELM can be expressed as

$$f(\boldsymbol{x}) = \boldsymbol{h}(\boldsymbol{x})\boldsymbol{H}^{\mathrm{T}}\left(\frac{1}{C}\boldsymbol{I}_N + \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}}\right)^{-1}\boldsymbol{T} = \begin{bmatrix} \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_1) \\ \vdots \\ \boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_N) \end{bmatrix}^{\mathrm{T}} \left(\frac{1}{C}\boldsymbol{I}_N + \boldsymbol{\Omega}_{ELM}\right)^{-1}\boldsymbol{T}, \tag{9}$$

where $\boldsymbol{h}(\boldsymbol{x})$ denotes the output function of hidden nodes and $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i)$ the kernel function. Then the output weight of KELM can be solved as

$$\alpha = \left(\frac{1}{C}\boldsymbol{I}_N + \boldsymbol{\Omega}_{ELM}\right)^{-1}\boldsymbol{T}. \tag{10}$$

Commonly used kernel functions include linear, polynomial, RBF, and sigmoid kernels. Among these functions, the RBF kernel function has few variables, a simple structure, and global approximation capability. The detailed expression for the RBF kernel function is

$$\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}_i\|^2}{\sigma^2}\right). \tag{11}$$

From the above, with a set of N samples $(\boldsymbol{x}_i, \boldsymbol{t}_i)$ and the kernel function $\boldsymbol{K}(\boldsymbol{x}, \boldsymbol{x}_i)$, the KELM prediction model can be formulated as Eq. (9). With KELM, difficulties in ELM when determining the number of hidden nodes can be avoided because the number of hidden nodes is the number of samples. Furthermore, the merits of a fast training speed and simple structure in ELM are retained, and those of good stability and generalization in SVM can be introduced.

Further analysis of Eq. (9) will show that the stability and generalization of KELM are closely related with the parameters C and σ. If these two parameters are optimized, optimal stability and generalization can be acquired. To this end, self-adaptive PSO is introduced to optimize the above two parameters for KELM.

4. Improved Prediction Model for KELM with PSO

4.1 Algorithm of PSO

The PSO algorithm is a kind of swarm intelligent optimization algorithm generated from the foraging behavior of birds.(22–24) In PSO, one particle denotes one potential optimized solution, and three indexes, velocity, location, and fitness, denote its characteristics. During the optimizing process, each particle is evaluated by goodness of fit, and the moving direction and distance of particles are controlled by the velocity vector. During an iteration, particles can update their velocity and location referring to their memory of the optimal location and finally achieve individual optimization in the solution space after some iterations.

It is assumed that the search space is of D dimensions; the population is composed of n particles, that is, $\boldsymbol{X} = [X_1, X_2,..., X_n]$, among which the ith particle is expressed as $\boldsymbol{X}_i = [x_{i1}, x_{i2},..., x_{iD}]^{\mathrm{T}}$ and $x_{min} \leq x_i \leq x_{max}$ with x_{min} and x_{max} denoting the lower and upper bounds, respectively; the velocity of the ith particle can be expressed as $\boldsymbol{V}_i = [v_{i1}, v_{i2},..., v_{iD}]^{\mathrm{T}}$ and $v_{min} \leq v_i \leq v_{max}$ with v_{min} and v_{max} denoting the lower and upper bounds, respectively; the fitness function J is defined in accordance with the optimization function following the equation

$$J = \sqrt{pemean^2 + pemse^2}, \tag{12}$$

where *pemean* denotes the mean of the prediction and *pemse* denotes the mean square error of the prediction error.

During an iteration, particles update their velocities and locations in accordance with the ultimate values of the individual and the population following the equation

$$\begin{cases} \boldsymbol{V}_i^{k+1} = \omega \boldsymbol{V}_i^k + c_1 r_1 \left(\boldsymbol{P}_i - \boldsymbol{X}_i^k\right) + c_2 r_2 \left(\boldsymbol{P}_g - \boldsymbol{X}_i^k\right), \\ \boldsymbol{X}_i^{k+1} = \boldsymbol{X}_i^k + \boldsymbol{V}_i^{k+1}, \end{cases} \tag{13}$$

where ω denotes the inertial weight; c_1 and c_2 denote non-negative acceleration constants; r_1 and r_2 are random values between 0 and 1; $\boldsymbol{P}_i$ and $\boldsymbol{P}_g$ are the ultimate individual value and population value, respectively.

In order to overcome disadvantages such as premature convergence, slow search, and low efficiency in PSO, two measures can be taken. One is to introduce genetic and mutation algorithms to reinitialize some variables with some probability and narrow the search space. The other is to adopt a new inertial-weight-selecting scheme to strengthen the search ability in the initial solution space. In this study, the latter measure is adopted and the improved weight can be calculated as

$$\omega(i) = \omega_{start} - \frac{\omega_{start} - \omega_{end}}{T_{max}} i, \tag{14}$$

where ω_{start} is the weight at the start, ω_{end} is the weight at the end, i is the current iterative cycle, and T_{max} is the maximum number of iterations. In Eq. (14), the inertial weight decreases with increasing number of iterations thus the local search ability is strengthened.

4.2 Prediction model based on PSO-KELM

In the prediction model based on PSO-KELM, two parameters in KELM, the penalty coefficient and the kernel parameter, should first be determined with the PSO algorithm and then, with these optimized parameters, training and prediction with KELM can be carried out. The detailed steps are as follows.

Step 1: Parameter optimization with the PSO algorithm.

(1) Initialize particle population including initial velocity $\boldsymbol{V}$ and location $\boldsymbol{X}$; assign maximum number of iterations, population size, inertial weight, and acceleration constant.

(2) Calculate the fitness of each particle: these particles with low fitness have a good optimizing index.

(3) Update ultimate values of $\boldsymbol{P}_i$ and $\boldsymbol{P}_g$ in accordance with the fitness.

(4) Update the velocity and location of each particle in accordance with $\boldsymbol{P}_i$ and $\boldsymbol{P}_g$.

(5) End of Step 1: The parameters C and σ at the maximum number of iterations are selected as the optimized values.

Step 2: Training and prediction with KELM

(1) With the parameters C and σ optimized using the PSO algorithm and the sample set with N samples, the output weight matrix $\boldsymbol{\alpha}$ and other structural parameters of KELM can be determined using Eq. (10).

(2) With the trained model, the prediction of deck motion can be executed using the current deck-motion measurements and Eq. (9).

The flowchart of the system is shown in Fig. 2.

5. Simulation and Analysis

5.1 Simulation setting

During simulation, the data from Eq. (1) are used as theoretical data without any error. White noise with variances of 0.01° and 0.05 m are added to the theoretical pitch, roll, yaw, and heave and used as actual values measured using SINS. The update frequency of samples is 1 Hz and the preset prediction time is 15 s. After some simulation experiments and considering the issues of real-time and precision in the prediction of deck motion, the number of samples is set to 50. The duration of data used for simulation is 2300 s and the data in the first 500 s are used as samples and those in the latter 1800 s are used in the prediction test. The input and output data are constructed as

$$\begin{cases} \boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N] = \begin{bmatrix} x_1 & x_2 & \cdots & x_N \\ x_2 & x_3 & \cdots & x_{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_n & x_{n+1} & \cdots & x_{N+n-1} \end{bmatrix}. \\ \boldsymbol{T} = [\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_N] = \begin{bmatrix} t_1 & t_2 & \cdots & t_N \\ t_2 & t_3 & \cdots & t_{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ t_m & t_{m+1} & \cdots & t_{N+m-1} \end{bmatrix}. \end{cases} \tag{15}$$

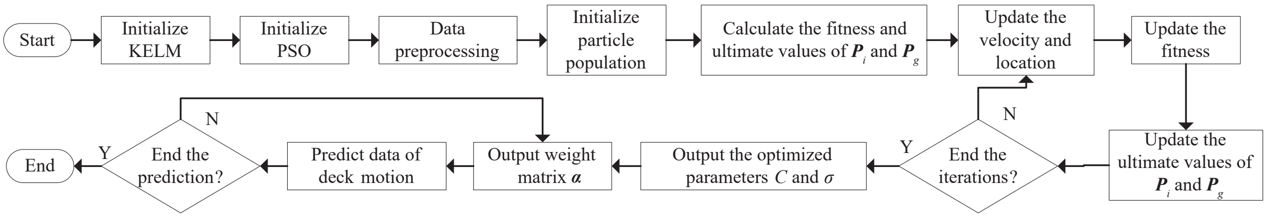
Start
Initialize KELM
Initialize PSO
Data preprocessing
Initialize particle population
Calculate the fitness and ultimate values of $\boldsymbol{P}_i$ and $\boldsymbol{P}_g$
Update the velocity and location
Update the fitness
Update the ultimate values of $\boldsymbol{P}_i$ and $\boldsymbol{P}_g$
End the iterations?
N
Y
Output the optimized parameters C and σ
Output weight matrix $\boldsymbol{\alpha}$
Predict data of deck motion
End the prediction?
N
Y
End

Fig. 2. Flowchart of the system.

After the completion of the model training, the data in the past 50 s are used to predict the data in the next 15 s when the new data are added and the oldest data are omitted. The total amount of model data is unchanged. In this way, the total data of 1800 s are predicted, and the input and output data are normalized and antinormalized to meet the needs of the model.

The initial settings for the PSO algorithm are as follows: the maximum number of iterations is 40; the initial population is 20; the initial inertial weight is 0.8; the last inertial weight is 0.2; the acceleration constants are both 1.49445; the search range for C and σ are both from 0.001 to 1000; the maximum and minimum values for position and speed are 20 and 0, respectively; 500 s of data are used for parameter optimization, and according to Eq. (15), 436 samples can be produced and the first 400 samples are used for model training and the last 36 samples are used for parameter optimization tests.

5.2 Simulation results and analysis

The optimized values using the PSO algorithm are shown in Table 1. Taking the PSO algorithm optimization process of pitch data as an example, as shown in Fig. 3, it is found that the fitness value changes from 0.3362 to 0.1023 after two evolutionary optimization processes. After 40 iterations, the fitness value is stabilized at 0.1015. The PSO algorithm can quickly find the appropriate penalty factor C and kernel parameter σ and keep the fitness value in a very low range. For roll, yaw, and heave data, similar results are also obtained.

Simulation results including pitch, roll, yaw, and heave are shown in Fig. 4 with solid lines denoting the actual measured values, dashed lines denoting the predicted values, and dot-dashed

Table 1
Optimized values using PSO algorithm.

	Pitch	Roll	Yaw	Heave
C	66.3026	239.7010	21.2722	20.0010
σ	980.6548	713.1792	840.4762	268.2339
J	0.1015	0.0151	0.0788	0.1038

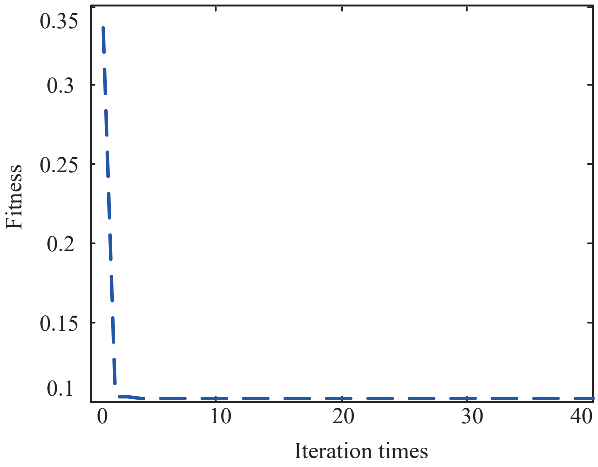

Fig. 3. (Color online) Optimal individual fitness value of pitch training sample.

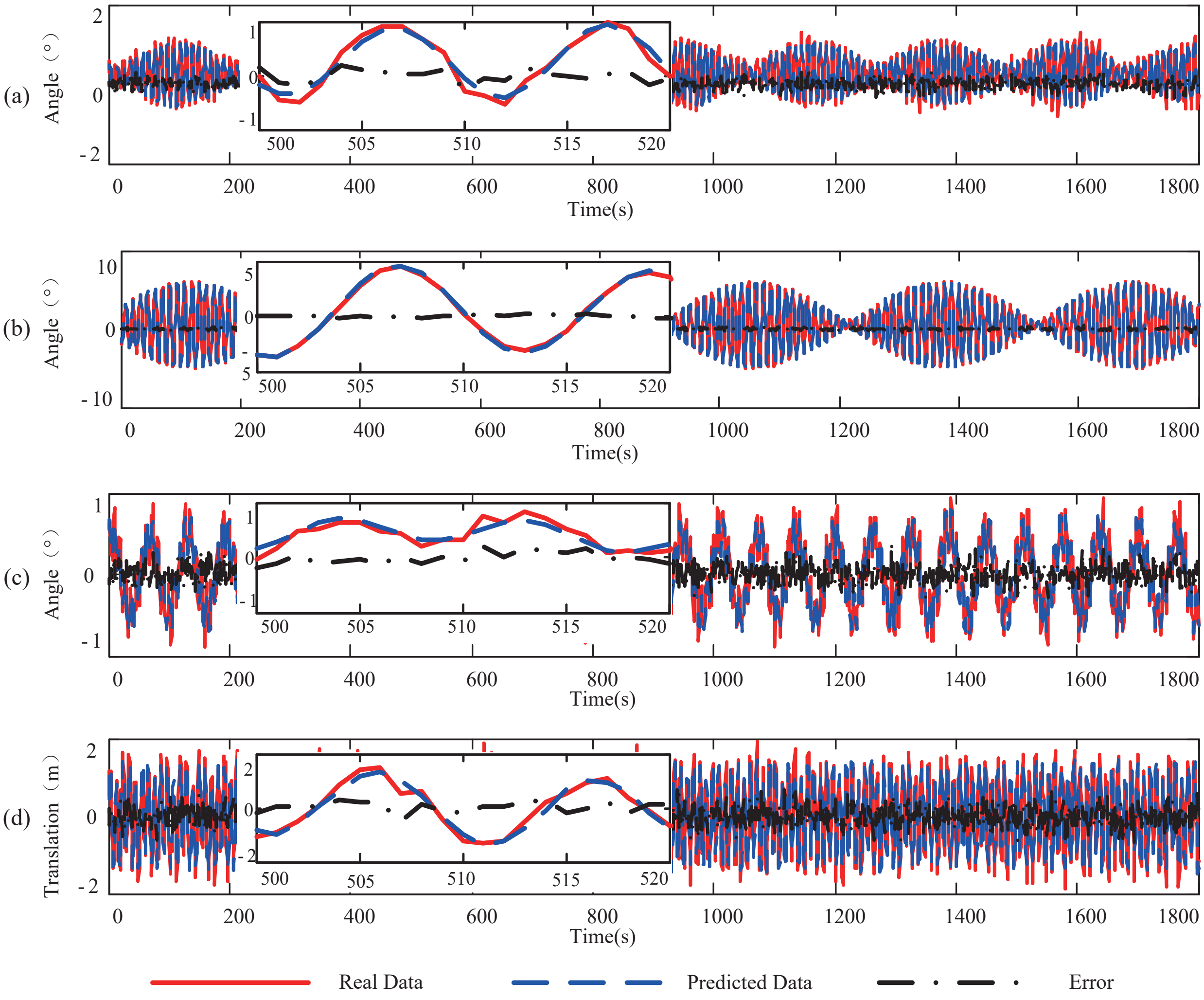

Fig. 4. (Color online) Results of prediction using PSO-KELM model. (a) Pitch, (b) roll, (c) yaw, and (d) heave.

lines denoting the prediction error. From the curves in Fig. 4, it can be seen that with PSO-KELM, the deck motion in the next 15 s can be effectively predicted with only small fluctuation deviations.

Furthermore, in order to evaluate prediction accuracy using PSO-KELM, prediction methods based on BP and ELM are selected to execute prediction with the same deck-motion data. The prediction error and training time for BP, ELM, and PSO-KELM are listed in Tables 2–4, respectively. In Tables 2–4, variance percentage means the variance of actual data divided by the variance of the prediction error.

Seen from Tables 1–3, it is obvious that PSO-KELM runs about 20 times faster than BP but about 10 times slower than ELM. The reason is that in contrast with BP, PSO-KELM has no constant error feedback correction process and in contrast with ELM, PSO-KELM has a parameter optimization process. PSO-KELM can be completed to predict data over 1800 s within 38 s, the average per-second data prediction does not exceed 0.03 s, so PSO-KELM fully meets real-time

Table 2
Results of prediction with BP model.(25)

	Error variance (degree or m)	Variance percentage (%)	Training time (s)
Pitch	0.0212	11.5220	605.5803
Roll	0.0165	0.2221	1111.1351
Yaw	0.0196	11.6673	652.2090
Heave	0.0754	9.1123	609.8079

Table 3
Results of prediction with ELM model.

	Error variance (degree or m)	Variance percentage (%)	Training time (s)
Pitch	0.0198	10.9925	4.1808
Roll	0.1165	1.4612	4.4304
Yaw	0.0167	10.0306	4.3836
Heave	0.0765	9.4784	4.4616

Table 4
Results of prediction with PSO-KELM model.

	Error variance (degree or m)	Variance percentage (%)	Training time (s)
Pitch	0.0124	6.7588	32.1206
Roll	0.0137	0.1723	33.5402
Yaw	0.0111	6.6526	34.1642
Heave	0.0560	6.7609	37.2686

requirements for deck-motion prediction of aircraft carriers. Note that in this paper, predictions are run with a personal computer (PC) and the running times are gathered from the same PC.

Also seen from Tables 1–3, the results of prediction with PSO-KELM have higher accuracy than those of BP and ELM, except that the mean square of the roll error with BP is 0.01652, while the mean square of the roll error with PSO-KELM is 0.01695; the errors of the rest of the results with BP are 1 to 2 times larger than those with PSO-KELM. The roll error with ELM is about 8 times larger than that with PSO-KELM; the errors of the rest of the results with ELM are 1 to 2 times larger than those with PSO-KELM. From the above analysis, it can be concluded that with PSO-KELM, deck motion can be accurately predicted with a short training time.

6. Conclusion

The problem of deck-motion prediction of an aircraft carrier was studied. Firstly, the characteristics of deck motion were analyzed, and a combined model based on the composition of sine waves was adopted to describe deck motion. To satisfy the requirements of real-time calculation and high-accuracy of deck-motion prediction, ELM was introduced, and the concept of kernel mapping in SVM was introduced to overcome the difficulty in selecting the number of hidden nodes as well as the instability and poor generalization caused by random mapping in ELM. Further analysis indicated that the stability and generalization ability are closely related to the penalty coefficient and kernel parameter. To acquire optimal values of these parameters for KELM, autoadaptive PSO was introduced, and then the prediction model and steps to be implemented based on PSO-KELM were designed. Simulation with prediction models based on PSO-KELM, BP, and ELM were executed and the results were compared. The comparison indicated that the prediction accuracy and training speed of PSO-KELM are better and faster than those of ELM and BP, and the model of PSO-KELM can be used for successful deck motion prediction of an aircraft carrier.

References

1 D.-S. Xu, X.-Y. Liu, and L.-X. Wang: J. Beijing Univ. Aeronaut. Astronaut. **37** (2011) 289.
2 X.-D. Yang, Z.-C. Wang, A.-G. Shi, B. Liu, and L. Li: Adv. Mater. Res. **2450** (2013) 1550.
3 L.-Q. Cai, J. Jiang, X.-H. Wang, and T.-T. Pan: Flight Dyn. **32** (2014) 105.
4 S.-M. Zhou: Aircr. Des. **32** (2012) 28.
5 C. Hong and Y.-X. Chen: Ship Eng. **34** (2012) 236.
6 J.-C. Yin, Z.-J. Zou, F. Xu, and N.-N. Wang: Neurocomputing **129** (2014) 168.
7 X.-Q. Jia, P. Lin, M.-W. Wang, and J. Shen: Aeronaut. Comput. Tech. **40** (2010) 114.
8 X.-Y. Peng, X.-R. Zhao, and Q.-F. Gao: J. Syst. Simul. **19** (2007) 267.
9 M. Triantafyllou, M. Bodson, and M. Athans: Oceanic Eng. **8** (1983) 9.
10 F. Cai, L. Wan, and A. Shi: J. Hydrodyn. **6** (2005) 780.
11 J. C. Chung, Z. Bien, and Y. S. Kim: Oceanic Eng. **15** (1990) 244.
12 E. Kayacan, B. Ulutas, and O. Kaynak: Expert Syst. Appl. **37** (2010) 1784
13 V. Nicolau, D. Aiordachioaie, and R. Popa: Proc. Int. Joint Conf. Neural Networks (IEEE, 2004) p. 2801.
14 M. Gu, C.-D. Liu, and J.-F. Zhang: J. Ship Mech. **17** (2013) 1147.
15 G.-B. Huang, D.-H. Wang, and Y. Lan: Int. J. Mach. Learn. Cybern. **10** (2010) 107
16 G.-B. Huang, H.-M. Zhou, X.-J. Ding, and R. Zhang: IEEE Trans. Syst. Man Cybern. **4** (2012) 513.
17 G.-B. Huang, Z. Bai, L. L. C. Kasun, and C.-M. Vong: IEEE Comput. Intell. Mag. **10** (2015) 18.
18 G.-B. Huang, Q.-Y. Zhu, and C. K. Siew: Inst. Electr. Electron. Eng. Inc. **2** (2004) 985.
19 K. Lang, M.-Y. Hang, and Y.-B. Yuan: J. Comput. Appl. **35** (2015) 2083.
20 M. Sidar and B. Doolin: IEEE Trans. Autom. Control **28** (1983) 350.
21 Y.-H. Zhang and X. Zhou: J. Syst. Simul. **25** (2013) 826.
22 S.-Z. Lv and Z.-R. Hou: Acta Electronica Sinica **32** (2004) 416.
23 T. K. Sharma, M. Pant, and T. Bhardwaj: Proc. Int. Conf. Computer Applications and Industrial Electronics (IEEE, 2011) p. 108.
24 H. Azim and K. Farshid: Reg. Energy Resour. Inf. Center **15** (2015) 93.
25 X.-X. Liu, Y.-J. Huang, Q.-M. Wang, Q. Song, and L.-Y. Zhao: Proc. 10th Int. Conf. Sensing Technology (IEEE, 2016) p. 1.

About the Authors

Xixiang Liu received his B.S. degree from Nanjing Forestry University, China, in 1999 and his M.S. and Ph.D. degrees from Southeast University, China, in 2004 and 2007, respectively. From 2007 to 2010 and 2010 to 2015, he was a lecturer and an associate professor, respectively, at Southeast University, China. Since 2015, he has been a professor with Southeast University. His research interests are inertial sensors and navigation.

Qiming Wang received his B.S. degree from Southeast University, China, in 2016. Since 2016, he has been a graduate student with Southeast University, China. His research interests are inertial sensors and navigation.

Yongjiang Huang received his B.S. degree from Tianjin Polytechnic University, China, in 2013 and his M.S. degree from Southeast University, China, in 2017. Now he is a doctoral student with Southeast University, China. His research interests are inertial navigation and integrated navigation.

Qing Song received her B.S. degree from Anhui Jianzhu University, China, in 2013 and her M.S. degree from Southeast University, China, in 2016. Her research interests are inertial navigation and data processing.

Liye Zhao received his B.S. degree from Nanjing Agriculture University, China, in 1999 and his M.S. and Ph.D. degrees from Southeast University, China, in 2003 and 2006, respectively. From 2006 to 2010 and 2010 to 2016, he was a lecturer and an associate professor, respectively, at Southeast University, China. Since 2016, he has been a professor with Southeast University. His research interests are inertial sensors, theories of MEMS inertial sensor systems, and the application of MCU, DSP, or ARM in human motion detection systems.