Journal of Engineering and Applied Science

RESEARCH Open Access

A high accuracy modeling scheme for dynamic systems: spacecraft reaction wheel model

Abd-Elsalam R. Abd-Elhay[1*], Wael A. Murtada[1] and Mohamed I. Yosof[2]

* Correspondence: Abdelsalam.ramadan@narss.sci.eg
[1]National Authority for Remote Sensing and Space Sciences (NARSS), 23 Jozeph Tito St., Cairo, Egypt
Full list of author information is available at the end of the article

Abstract

Reaction wheels are crucial actuators in spacecraft attitude control subsystem (ACS). The precise modeling of reaction wheels is of fundamental need in spacecraft ACS for design, analysis, simulation, and fault diagnosis applications. The complex nature of the reaction wheel leads to modeling difficulties utilizing the conventional modeling schemes. Additionally, the absence of reaction wheel providers' parameters is crucial for triggering a new modeling scheme. The Radial Basis Function Neural Network (RBFNN) has an efficient architecture, alluring generalization properties, invulnerability against noise, and amazing training capabilities. This research proposes a promising modeling scheme for the spacecraft reaction wheel utilizing RBFNN and an improved variant of the Quantum Behaved Particle Swarm Optimization (QPSO). The problem of enhancing the network parameters of the RBFNN at the training phase is formed as a nonlinear constrained optimization problem. Thus, it is proposed to efficiently resolve utilizing an enhanced version of QPSO with mutation strategy (EQPSO-2M). The proposed technique is compared with the conventional QPSO algorithm and different variants of PSO algorithms. Evaluation criteria rely upon convergence speed, mean best fitness value, stability, and the number of successful runs that has been utilized to assess the proposed approach. A non-parametric test is utilized to decide the critical contrast between the results of the proposed algorithm compared with different algorithms. The simulation results demonstrated that the training of the proposed RBFNN-based reaction wheel model with enhanced parameters by EQPSO-2M algorithm furnishes a superior prediction accuracy went with effective network architecture.

Keywords: Attitude control subsystem, Reaction wheel model, Quantum Particle Swarm Optimization, PSO, RBFNN

Introduction

In spacecraft missions that need a high pointing accuracy, Attitude Control Subsystem (ACS) with specific actuators shall be used. The reaction wheel (RW) is a vital actuator for the spacecraft ACS [1]. The accurate modeling of the spacecraft reaction wheel is recommended for the design, simulation, analysis, and fault identification applications. Meanwhile, increasing the accuracy of the reaction wheel modeling will improve the overall accuracy of the ACS modeling process. There are three common approaches for modeling dynamic systems. The noteworthy models are white box, black box, and gray box models. The white-box modeling is characterized by a good understanding of model parameters compared to black-box modeling that needs some measurements for the model inputs and outputs. Furthermore, the high accuracy modeling process can be achieved by black-box rather than white-box. Despite the black-box modeling accuracy, the generalization characteristics are proven to be superior in the case of white-box modeling rather than black-box modeling. Due to the complexity of the reaction wheel modeling, it is recommended to be in a white-box modeling manner. This is to satisfy the appropriate accuracy and generalization characteristics [2]. Unfortunately, many manufacturers provide insufficient information in their datasheets, which is needed to accurately model the dynamics of the reaction wheel. Therefore, building the white-box mathematical model is very difficult. Thus, the researchers have proposed many artificial intelligence (AI) schemes for modeling the reaction wheels [3–8]. For instance, Al-Zyoud and Khorasani [3] proposed a dynamic multilayer perceptron scheme for modeling the spacecraft reaction wheel. The dynamic properties were introduced into the multilayer perceptron network by adding delays between layers. Furthermore, the optimal results were obtained using six neurons at the hidden layers. Thus, a tiny training error was noticed in order of 0.04. However, the simulation results have shown that the dynamic multilayer perceptron had an improved performance compared to the linear reaction wheel model. There are some limitations for dynamic multilayer perceptron like the model complexity and noticeably low modeling accuracy. In [4], the three-layer Elman neural network is introduced to model the dynamics of the spacecraft reaction wheel. Therefore, the proposed Elman neural network had two inputs, 25 hidden neurons, and 1 output. Moreover, the network was trained through 5000 epochs to get a small mean square error of about 10^{-3}. Furthermore, simulation results have demonstrated the superiority of the Elman neural network-based observer compared to the linear observer for fault detection and identification. It was noticed that the former model has a computational complexity due to a large number of hidden neurons. Thus, this imposes a long computation time. Later on, Mousavi and Khorasani [5] proposed a reaction wheel model that represents four spacecraft formation flight missions. Thus, reaction wheel dynamics have been introduced by using an infinite impulse response filter with dynamic hidden layer neurons. Therefore, hopeful results were achieved from the four spacecraft constellations. The first one has a training error of 0.05 using a neural network architecture with ten hidden neurons. Furthermore, the second, third, and fourth spacecraft have a training error near of 0.018, 0.015, and 0.03 with eight, eight, and six neurons at their hidden layers, respectively. The drawback of the aforementioned proposed model is the use of the infinite impulse response filter that consumes tremendous computational resources.

Radial Basis Function Neural Networks (RBFNNs) are considered to be promising for modeling nonlinear dynamic systems like spacecraft reaction wheels. Moreover, RBFNN facilitates the modeling process due to its simple architecture, good generalization performance, low sensitivity against noise, and training capability [9]. Therefore, to address the drawbacks in related researches, this research proposes an efficient high accuracy modeling scheme for spacecraft reaction wheel using RBFNN. Many researchers have proposed RBFNN as a modeling paradigm in different research areas [9–15]. For instance, in [9], RBFNN had been used for online modeling and adaptive control of nonlinear systems. Furthermore, it is proved that RBFNN has a noticeable performance with the effect of noise and parameters' variations. Nevertheless, the results also proved that RBFNN has a better performance than the feedforward neural network. Ali N. et al. [11] investigated the superiority of RBFNN over multilayer perceptron for predicting the welding features. The results proved the effectiveness of the high accuracy modeling capability for RBFNN over multilayer perceptron in modeling dynamic systems. Recently, Yunguang et al. [13–15] suggested an optimization module based on radial basis function and particle swarm optimization to develop a wheel profile fine-tuning system. Simulation results have proven that the proposed optimization algorithm can recommend an optimal wheel profile according to train operators' needs.

Training the RBFNN includes calculating the number of hidden neurons, centers of the Radial Basis Function (RBF), widths of the hidden layers, and the connection weights. Therefore, determining the optimal values for these parameters is a crucial factor for the RBFNN network performance. To address this concern, an optimization algorithm shall be used to enhance the training performance and then the modeling accuracy. Recently, different optimization algorithms have revealed promising performance. When compared to other optimization approaches, Particle Swarm Optimization (PSO) has a robust search ability, fast computation, and is inexpensive in terms of speed and memory [16]. However, it was proven that PSO is certifiably not a global optimization algorithm [17]. Therefore, numerous variants of PSO have been proposed to work on the performance of PSO [18–23]. Quantum Behaved Particle Swarm Optimization (QPSO) algorithm is another adaptation of the conventional PSO that was presented by Sun [24]. It had been started by quantum mechanics and the analysis of PSO dynamic behavior. Besides, QPSO is a sort of stochastic algorithm that has iterative equations, which differ from that of the conventional PSO. Moreover, there are limited QPSO parameters that should be adapted compared with conventional PSO. Hence, experimental results showed that QPSO has a superior performance compared with the standard PSO on various benchmark functions [25]. In any case, QPSO is a proper algorithm for global optimization issues, yet it suffers from premature convergence. Consequently, this premature convergence enables performance degradation and inefficiency for solving optimization problems. This convergence is caused because of catching in local optimal. Nevertheless, premature convergence happens because of the consistent declination of particles' diversity [26].

This research proposes a high accuracy modeling scheme for spacecraft reaction wheel utilizing RBFNN and a further enhanced version of the QPSO algorithm. As an improvement, firstly, two progressive mutations were applied to further improve the exploitation process. Besides, a diversity control strategy is applied to enhance the particles' diversity and overcome the premature convergence. Subsequently, expanding the chance of the swarm to leap out the local minima and discovering new encouraging solutions further improve the algorithm performance. Accordingly, an improved QPSO algorithm, signified by Enhanced Quantum Particle Swarm Optimization – 2 Mutation (EQPSO-2M), is proposed for the training of the RBFNNbased reaction wheel model. The enhancement aims to improve the search abilities of QPSO and trying not to stick at local optimal. Moreover, the proposed reaction wheel mathematical model that was proposed in [27], has been implemented to create the dataset that is needed for the testing of the RBFNN-based reaction wheel model. The effectiveness of the proposed EQPSO-2M algorithm is investigated using convergence speed, mean best fitness value, stability, and the number of successful runs. The obtained results indicate the superior performance of the EQPSO-2M method. Once the optimal parameters of RBFNN are obtained, the performance of the proposed reaction wheel model has been tested using the simulation results.

Methods

Spacecraft dynamic model

Attitude Control Subsystem (ACS) is one of the vital systems in the spacecraft that provides the in-orbit attitude control and determination functions. ACS is conceptually composed of three main parts: attitude sensors, feedback control system, and actuators [28]. Figure 1 illustrates the simplified block diagram of the ACS subsystem. Spacecraft can be represented as a rigid body where the dynamics can be obtained using Euler's dynamical formulas.

Euler's equation is equivalent to Newton's second law for rotation about the center of mass. Thus, the body motion equations about its center of mass using reaction wheels as actuators are described by Euler equations as in [28] as follows:

$$\dot{\omega}_x = \frac{1}{I_{xx}} \left(\tau_x + \tau_{dx} - \omega_z \omega_y \left(I_{zz} - I_{yy} \right) \right) \tag{1}$$

$$\dot{\omega}_y = \frac{1}{I_{yy}} \left(\tau_y + \tau_{dy} - \omega_x \omega_z \left(I_{xx} - I_{zz} \right) \right) \tag{2}$$

$$\dot{\omega}_z = \frac{1}{I_{zz}} \left(\tau_z + \tau_{dz} - \omega_y \omega_x \left(I_{yy} - I_{xx} \right) \right) \tag{3}$$

where I_{xx} , I_{yy} and I_{zz} represent the spacecraft moment of inertia. ω_x, ω_y, and ω_z are the spacecraft's angular velocities in body-fixed axes toward inertial coordinate system along x, y, and z axes, respectively. τ_{dx}, τ_{dy}, and τ_{dz} represent the disturbances torques, which act on the spacecraft about roll, pitch, and yaw axis respectively. τ_x, τ_y, and τ_z represent the torque due to the motion of the wheel on each axis. To get the spacecraft's actual attitude, which are the Euler angles roll, pitch, and yaw, the Eqs. 1, 2, and 3 shall be integrated twice.

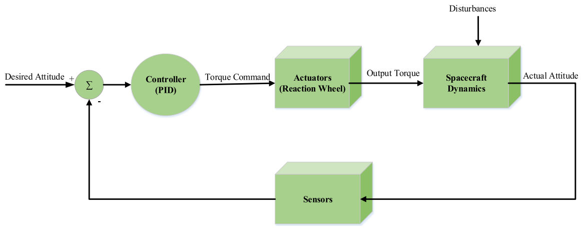

Fig. 1 Attitude control subsystem block diagram

Reaction wheel mathematical model

Reaction wheels are the common actuators for three axes stabilized spacecraft ACS, specifically for unmanned spacecraft. They are simply flywheels mounted to an electric direct current (DC) motor that can rotate in the desired direction to establish one axis control for each RW [29]. Furthermore, the reaction wheel is a nonlinear ACS component, which consists of several internal loops. Thus, these loops should be considered to ensure accurate mathematical modeling. Figure 2 illustrates the RW internal loops, which are described in [27]. The block diagram in Fig. 2 can be described mathematically as in [30] by Eqs. 4 and 5 as follows :

$$\begin{bmatrix} \dot{I}_m \\ \dot{w}_m \end{bmatrix} = \begin{bmatrix} G_d w_d [\Psi_1(I_m, w_m) - \Psi_3(w_m) - w_d I_m] \\ \frac{1}{J_\omega} [k_t I_{m-\tau_c} \Psi_2(w_m) - \tau_v w_m] \end{bmatrix} + \begin{bmatrix} G_d w_d \\ 0 \end{bmatrix} V_{com} \tag{4}$$

$$\tau = k_t I_m \tag{5}$$

In Eq. 4, I_m represents the motor current, k_t is the motor torque constant, w_m is the motor angular velocity, G_d is the driver gain, w_d is the driver bandwidth, and Ψ_1, Ψ_2, and Ψ_3 represent the nonlinearities for back-EMF limiting torque, Coulomb friction, and speed limiter circuit. This research proposes the use of the ITHACO type-A

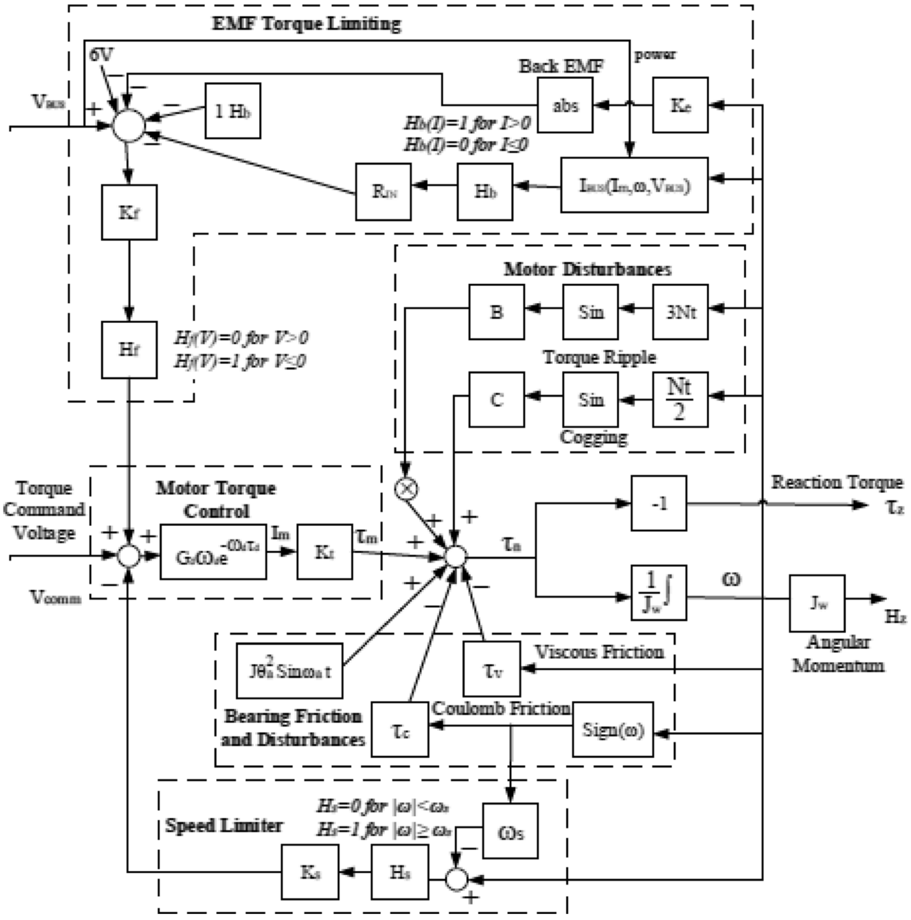

Fig. 2 Reaction wheel mathematical model

reaction wheel, which is produced by Goodrich Corporation. Table 1 shows the parameters of the ITHACO type-A reaction wheel.

Radial Basis Function Neural Network Architecture

The RBFNNs were firstly proposed in 1988 [31] based on the principle that the biological neuron has a local response. Moreover, RBFNN has a simple architecture, fast training time, and efficient approximation capabilities rather than other neural networks [9]. A typical architecture of RBFNN includes three layers: input layer, hidden layer, and output layer as depicted in Fig. 3. The input layer consists of input nodes that are connecting the inputs to the neural network. The hidden neurons use radial basis function such as Gaussian function $\varphi_i(x)$ as the activation function as follows:

$$\varphi_i(x) = \exp\left(-\frac{\|x - c_i\|^2}{{\sigma_i}^2}\right) \tag{6}$$

where x represents the network input, σ_i and c_i are width and center of the ith neuron, respectively, and ‖•‖ is the Euclidean distance between two different vectors. The output layer has a linear activation function that produces the network output corresponding to the network input [10]. Thus, the output of the network y_j can be addressed as follows:

$$y_j = \sum_{i=1}^{n} w_i \varphi_i(x) + b_i \tag{7}$$

In Eq. 7, b_i and w_i , are the bias and the weight of the ith neuron respectively. Therefore, to define the proposed RBFNN-based reaction wheel model, it is mandatory to determine some critical parameters. These parameters include the number of input neurons, number of hidden neurons, output layer's neurons, and the weights of all neurons. In addition, other important parameters shall be tuned like the centers and the widths of the hidden neurons. Generally, the number of the problem inputs will determine the number of input layer neurons [32]. Thus, the input layer of the proposed RBFNN-based reaction wheel model comprises a single neuron that represents the torque command voltage. Furthermore, the number of the output layer neurons is

Table 1 ITHACO type-A RW main parameters

Parameter	Description	Value
G_d	Driver gain	0.19 A/V
K_t	Motor torque constant	0.029 N.m/A
K_e	Motor back-EMF	0.029 V/rad/s
K_s	Over-speed circuit gain	95 V/ rad/s
w_s	Maximum wheel speed	690 rad/s
w_d	Driver bandwidth	2000 rad/s
R_{in}	Input resistance	2 Ω
K_f	Voltage feedback gain	0.5 V/V
N	Number of motor poles	36
T_c	Torque command range	− 5 to + 5 V
τ_c	Coulomb fiction	0.002 N.m
J	Flywheel inertia	0.0077 N.m.s^2

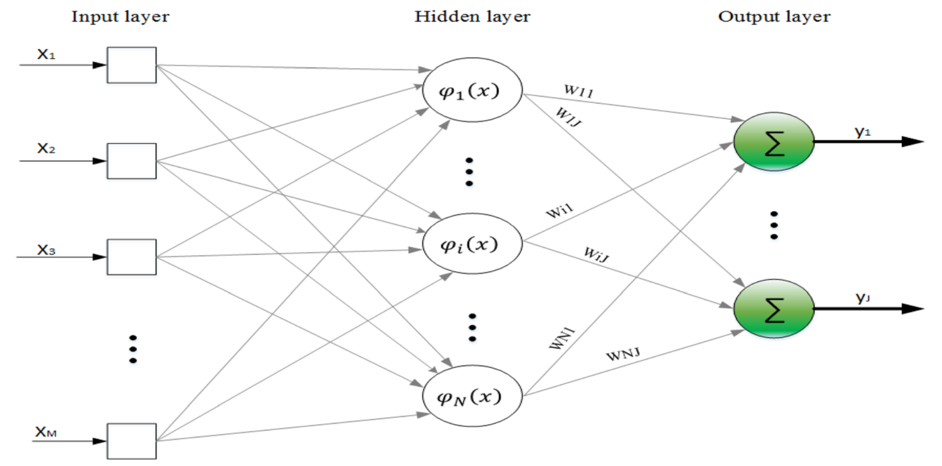

Fig. 3 Typical structure of the RBFNN

determined corresponding to the number of model outputs. Because the reaction wheel has only one output, which is the generated torque, thus the output layer has a single neuron. The number of hidden layer neurons has a paramount impact on the RBFNN performance. Generally speaking, the more the neurons in the hidden layer, the better the network accuracy [33]. However, the addition of hidden neurons after the right number is reached will not improve the network accuracy, but increase the computational power and architectural complexity. Therefore, the optimal number of neurons in the hidden layer needs to be justified experimentally and it is based on the network designer experience [34] as will be introduced in the experimental results and discussions section. Furthermore, the centers, widths, and weights between the hidden neurons and the output layer shall be estimated. Thus, this research proposes an enhanced version of QPSO, which is EQPSO-2M to estimate the optimal values of the centers, widths, and weights.

Standard Particle Swarm Optimization

PSO was proposed by Eberhart and Kennedy [35]. In the PSO algorithm, each particle is assumed as a point in an N-dimensional Euclidian space. Moreover, at each iteration, there are three vectors, which are used to describe the behavior of the particle i that are: the current position vector: $\boldsymbol{X_{i,n}} = (X^1_{i,n}, X^2_{i,n}, \ldots, X^j_{i,n}, \ldots, X^N_{i,n})$; the velocity vector: $\boldsymbol{V_{i,n}} = (V^1_{i,n}, V^2_{i,n}, \ldots, V^j_{i,n}, \ldots, V^N_{i,n})$, and the personal best position vector; $\boldsymbol{P_{i,n}} = (P^1_{i,n}, P^2_{i,n}, \ldots, P^j_{i,n}, \ldots, P^N_{i,n})$; where $(1 \leq j \leq N)$. Therefore, at the $(n + 1)$ iteration, the particles' velocity and position vectors are updated as the following [36]:

$$V^j_{i,n+1} = V^j_{i,n} + c_1 r^j_{i,n}\left(P^j_{i,n} - X^j_{i,n}\right) + c_2 R^j_{i,n}\left(G^j_{i,n} - X^j_{i,n}\right), \quad (8)$$

$$X^j_{i,n+1} = X^j_i + V^j_{i,n+1} \quad (9)$$

where c_1 and c_2 are the acceleration coefficients, $r^j_{i,n}$, $R^j_{i,n}$ are two different random numbers that are distributed uniformly over (0, 1); therefore, $\{r^j_{i,n} : j = 1, 2, \ldots, N\} \in U(0, 1)$; {

$R_{i,n}^{j} : j = 1, 2, \ldots, N\} \in U(0, 1)$.$\boldsymbol{G_{i,n}^{j}}$, is the global best position vector. It is noticed that when c_1 is greater than c_2, the swarm has a higher local search ability. On the other hand, the swarm explores the search space more globally when c_2 is greater than c_1 [37]. To improve the performance of the standard PSO and minimize the probability of trapping in local optimal, many PSO variants have been proposed. For instance, Ziyu and Dingxue [21] introduced the Time-varying Adaptive PSO (TAPSO) version without using the velocity of the previous iteration. Thus, the particle's velocity update can be formulated as follows:

$$V_{i,n+1}^{j} = c_1 r_{i,n}^{j}\left(P_{i,n}^{j} - X_{i,n}^{j}\right) + c_2 R_{i,n}^{j}\left(G_{i,n}^{j} - X_{i,n}^{j}\right), \quad (10)$$

In TAPSO, the reinitialization criterion is based on assuming random velocity to avoid premature searching for the velocity of a particle at zero. Moreover, the authors introduced an exponential time-varying acceleration coefficient to enhance the exploration and exploitation capabilities. Therefore, the acceleration coefficients are updated according to the following equations:

$$c_1 = c_{\min} + (c_{\max} - c_{\min}).e^{-\left(\frac{4k}{G}\right)^2} \quad (11)$$

$$c_2 = c_{\min} - (c_{\max} - c_{\min}).e^{-\left(\frac{4k}{G}\right)^2} \quad (12)$$

where $\boldsymbol{k}$ represents the current iteration number, and $\boldsymbol{G}$ represents the maximum number of iterations. PSO is simple to implement, has a fast convergence, and its convergence can be controlled using a few coefficients. For this reason, it has been used for solving a wide range of optimization problems. However, standard PSO can't converge to the global optimal when it is used with complex optimization problems [38].

Quantum Behaved Particle Swarm Optimization (QPSO)

QPSO algorithm was introduced by Sun in 2004 based on quantum mechanics and computing [39]. In QPSO, the particle's state is represented by a wave function. Therefore, the probability of the particles that appear in position $\overrightarrow{x}$ can be estimated from the probability density function of its position [40]. Regarding PSO convergence analysis, PSO converges when each particle converges to the local attractor $p_{i,n}^{j}$ that can be represented by:

$$p_{i,n}^{j} = \phi.P_{i,n}^{j} - (1-\phi).G_{n}^{j}, \phi \sim U(0,1) \quad (13)$$

where $p_{i,n}^{j}$ represents the *j*th dimension of the particle local attractor, $P_{i,n}^{j}$ is the particle best position, and G_{n}^{j} is the global best position. It is assumed that the particle *i* moves in *N*-Dimensional space with a δ potential well at $p_{i,n}^{j}$ to guarantee the algorithm convergence at *n* iterations. Using the Monte Carlo method, the position for the *j*th dimension of the *i*th particle at $n + 1$ iteration is formulated according to Eq. 14 as follows [40]:

$$X_{i,n+1}^{j} = \begin{cases} p_{i,n}^{j} + \alpha \left|X_{i,n}^{j} - C_{n}^{j}\right| * \ln\left(1 \Big/ u_{i,n+1}^{j}\right), \text{if } m \geq 0.5 \\ p_{i,n}^{j} - \alpha \left|X_{i,n}^{j} - C_{n}^{j}\right| * \ln\left(1 \Big/ u_{i,n+1}^{j}\right), \quad \text{if } m < 0.5 \end{cases} \quad (14)$$

where u and m are two random numbers that are uniformly distributed in [0,1] and $C_n^j = (C_n^1, C_n^2, \ldots, C_n^N)$ is the average of the best positions for all particles. Thus, it can be calculated by:

$$C_{n+1}^j = \left(\frac{1}{M}\right)\sum\nolimits_{i=1}^{M} P_{i,n}^j (1 \le j \le N) \quad (15)$$

In Eq. 14, α represents the contraction-expansion (CE) coefficient that enhances the performance of QPSO when it is properly selected [39]. Many proposed methods were introduced to control the contraction-expansion coefficient such as in [41, 42].

The proposed Enhanced Quantum Behaved Particle Swarm Optimization Algorithm

Although the QPSO algorithm has revealed a good performance to find the optimal solution for many optimization problems [40]. However, it still introduces a deteriorative performance in searching for the global optimal solution in complex optimization problems. This performance degradation in QPSO occurs due to the premature convergence. To resolve this problem for QPSO and other PSO variants, this research proposes an EQPSO-2M algorithm that has two significant improvements. First, the diversity of particles is enhanced to guarantee a healthy diversity of the particles during the search process. Therefore, to avoid the premature convergence of the algorithm. The particle diversity is calculated using the following formula:

$$D = \frac{1}{M.|A|} \sum_{i=1}^{m} \sqrt{\sum\nolimits_{j=1}^{N} \left(X_{i,n}^j - \overline{C}_n^j\right)^2} \quad (16)$$

where M represents the swarm size, N represents the dimensions of the problem, A denotes the length of the longest diagonal in the search space, $X_{i,n}^j$ is the jth component of the ith particle's position for the nth iteration, and $\overline{C}_n^j$ represents the particles' mean best position [42]. Meanwhile, the particles' diversity is monitored during the search process; when it is decreased below the threshold value d_{low}; the particles' mean best position will be reinitialized with values that maximize the diversity again as follows:

$$C_{n+1}^j = \begin{cases} X_{i,n+1}^j + X_{\max}, & if\ X_{i,n+1}^j > 0 \\ X_{i,n+1}^j + X_{\min}, & if\ X_{i,n+1}^j \le 0 \end{cases} \quad (17)$$

where $X_{\max}$ and $X_{\min}$ represent the maximum and minimum boundaries of the search interval respectively. The main idea behind the reassignment of the mean best position vector using Eq 17 is to increase the distance between the particle's position and the mean best position as we can. Thus, the population diversity will increase monotonically and this would make the particle escapes the local optima. The other improvement of the EQPSO-2M is to overcome the premature convergence by adding two consecutive single dimension Gaussian mutations on the particle's personal best position as follows:

$$P_{i,n}^j = P_{i,n}^j r_1 + 0.01 P_{i,n}^j r_2, r_1, r_2 \sim U(0,1) \quad (18)$$

where r_1 and r_2 represent two different arrays of uniform distribution random numbers. The two consecutive mutations will help the particles to explore extensively different regions of the search space to find the best positions. Thus, this is to enhance the convergence speed of the QPSO and to avoid premature convergence. Applying the diversity control and the two successive single dimension Gaussian mutations will avoid the premature convergence that may occur in the conventional QPSO. Moreover, these two processes can enhance the convergence speed of QPSO and prevent the algorithm from trapping in local minima. The pseudocode for the proposed EQPSO-2M is shown in Algorithm 1 as below:

Algorithm 1: Pseudocode for the proposed EQPSO-2M algorithm

```
Begin
Initialize the population of size M and the dimensions of the problem N;
Set the maximum and the minimum limits of search interval  $[X_{max}, X_{min}]$;
Set initial values for the particles' current position  $X_{i,0}^{j}$  and the best position $P_{i,0}^{j}$;
Compute the fitness of the current iteration and the personal best position using equation 19. Then calculate the global best position $G_0$;
Set the value of CE coefficient α to 0.75;
Set the value of diversity threshold value $d_{low}$;
for (n = 1 to Maximum Iterations)
        for (i = 1 to population size M)
        for (j = 1 to Dimensions N)
          $\varphi_{i,n}^{j}$ = rand (0,1);
          $p_{i,n}^{j} = \varphi_{i,n}^{j} P_{i,n}^{j} + (1-\varphi_{i,n}^{j}) G_n^{j}$;
          $u_{i,n}^{j}$ = rand (0,1);
          if (rand (0,1) < 0.5)
            $X_{i,n+1}^{j} = p_{i,n}^{j} + \alpha \left|X_{i,n}^{j} - C_n^{j}\right| \ln\left(1/u_{i,n+1}^{j}\right)$;
          else
            $X_{i,n+1}^{j} = p_{i,n}^{j} - \alpha \left|X_{i,n}^{j} - C_n^{j}\right| \ln\left(1/u_{i,n+1}^{j}\right)$;
          end if
        end for
        Compute the fitness of the particle at the position  $X_{i,n+1}^{j}$
        if (f ( $X_{i,n+1}^{j}$) < f ( $X_{i,n}^{j}$))
         Update the personal best position for the current particle $P_{i,n}$
        end if
        Apply the first mutation on the personal best $P_{i,n}$ using equation 18 and recalculate $P_{i,n}$ ;
        If the fitness of the new $P_{i,n}$ is better, then update the value of $P_{i,n}$ and the global best Gn
        End if
        Apply the second mutation on personal best $P_{i,n}$ using equation  18 and recalculate $P_{i,n}$ ;
        If the fitness of the new $P_{i,n}$ is better, then update the value of $P_{i,n}$ and the global best Gn.
        end for
        Compute the mean best position using equation 15.
        Compute the diversity of nth iteration $D_n(s)$ using equation 16 ;
        if (the diversity of the nth iteration $D_n(s)$ < Diversity threshold $d_{low}$)
          for (j = 1 to Dimension N)
             if ( $X_{i,n+1}^{j} < 0$)
                 $C_{n+1}^{j} = X_{i,n+1}^{j} + X_{max}$
             else
                 $C_{n+1}^{j} = X_{i,n+1}^{j} + X_{min}$
               end if
             end for
          end if
          Display and write the fitness of the global best position of n^th iteration at this run.
 end for
 end
```

In the above algorithm, QPSO is firstly initialized with the swarm size M and dimensions N. Therefore, the number of particles is set to 20. The swarm size selection will be discussed in the next section. The particles' positions and the personal best positions are randomly initialized. Furthermore, the initial global best and the mean best positions should be estimated. Thus, the value of α shall be set to 0.75 according to [39]. Moreover, the iterative process for updating the particle's current position $X_{i,n+1}^{j}$ should be started according to Eq. 14. Therefore, the fitness value is evaluated according to Eq. 19. When the fitness of the current particle's position is better than the previous one, the particle's best position $P_{i,\ n}$ should be updated. Hence, two consecutive mutations are applied on the particle's personal best position according to equation 18. After each mutation, the fitness of a new personal best position should be estimated to update the personal best position and the global best position of the particles $P_{i,\ n}$ and $\mathrm{G_n}$, respectively. Further, the particle's mean best position C_{n+1}^{j} should be calculated using Eq. 15. Thus, the diversity of the particle should be evaluated using Eq. 16, and then compared with the threshold value d_{low}. Meanwhile, when the current diversity is below the threshold; the mean best position should be estimated according to Eq. 17. The searching process will be continued until the maximum iterations are met.

Results and discussion

In order to evaluate the performance of the proposed modeling scheme, simulation experiments should be done to benchmark the proposed RW model. Furthermore, a 3-axis ACS nonlinear model was implemented using MATLAB/SIMULINK. It includes the spacecraft dynamic model, the RW mathematical model, and the Proportional Integral Derivative (PID) controller. Therefore, the input to the ACS model is the desired attitude and the output is the actual attitude.

The RW input will be the torque command voltage, and the output is the generated torque a large number of experiments, the training dataset is suggested for the whole simulation to run with perspective angles within the range of [−5°, 5°]. Moreover, the simulation time in every iteration is three hundred seconds. Figures 4 and 5 show RW input torque command signal and output torque, respectively.

RBFNN hidden layer size analysis

The determination of the suitable number of hidden neurons significantly affects the RBFNN performance. In this research, the number of hidden neurons is chosen on the basis that to get the best performance from RBFNN and keep the design of RBFNN as simple as could be expected. To choose the number of the hidden neurons, we began according to [43] with one hidden neuron and increment the number of neurons progressively by one neuron. Table 2 shows the results of this study.

It can be observed from Table 2 that the performance of the RNFNN model is improved when the number of hidden layer neurons increased. The RBFNN with only one hidden layer neuron has MSE ≈ 1.87E-05. As seen from Table 2, we can notice that RBFNN with two hidden neurons decreases the mean square error (MSE) to be 6.54E −07. However, increasing the number of hidden neurons to more than two neurons has no significant improvement in the model performance. Therefore, it is recommended for the number of RBFNN hidden layer neurons for the spacecraft reaction wheel model to be two neurons.

Fitness function

The problem of the RBFNN model training has been defined as a nonlinearly constrained optimization problem, which is settled utilizing the proposed EQPSO-2M. This optimization problem aims to find the optimal values of the RBFNN parameters that minimize the error between the RBFNN model output and the target output. Accordingly, to utilize the proposed EQPSO-2M algorithm for the training of the RBFNN-based reaction wheel model, a fitness function ought to be carried out. In this research, the well-known MSE has been chosen as the objective function. This function takes the difference between the RBFNN output and the actual reaction wheel output to compute the mean of the square errors as follows:

$$\mathrm{MSE}=\frac{1}{N}\sum_{i=1}^{N}\left\|t_i-\left(w_1\ \exp\left(-\frac{\|x_i-c_1\|^2}{{\sigma_1}^2}\right)+w_2\ \exp\left(-\frac{\|x_i-c_2\|^2}{{\sigma_2}^2}\right)+b\right)\right\|^2 \qquad (19)$$

In Eq. 19, N is the number of the training patterns, t_i is the target reaction wheel output torque, w_1 is the weight among the first hidden neuron and the output neuron, w_2 represents weight among the second hidden neuron and the output neuron c_1 and c_2 are the centers of the first and the second hidden neuron RBF respectively, σ_1 and σ_2 are the widths of first and second hidden neurons, respectively, and b is the bias. These parameters can be obtained when the fitness function in Eq. 19 is minimized.

Swarm size selection assessment

Picking the fitting population size of the QPSO algorithm is a principal factor that influences its performance. As a rule, the optimal swarm size relies upon the complexity of the optimization problem to be addressed. However, increasing the population size might increase the algorithm's performance, but it will increase the computational time. Then again, decreasing the number of particles to a specific limit might cause the

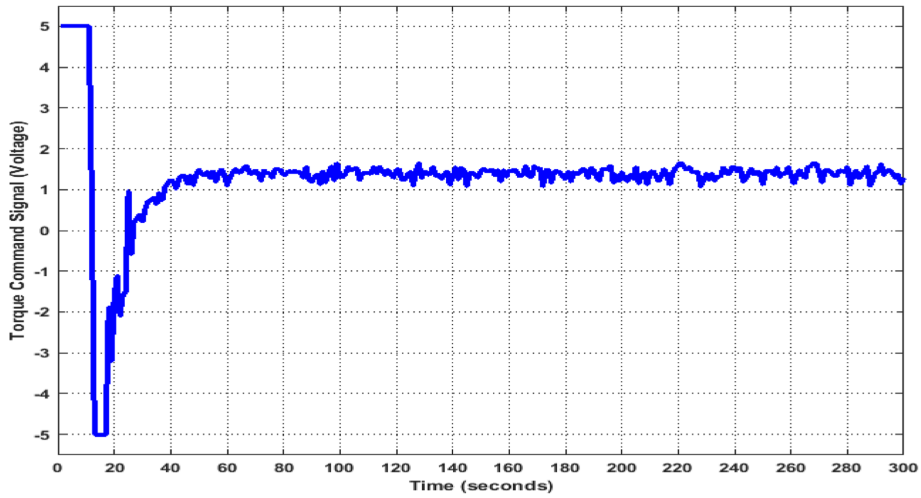

Fig. 4 Reaction wheel torque command

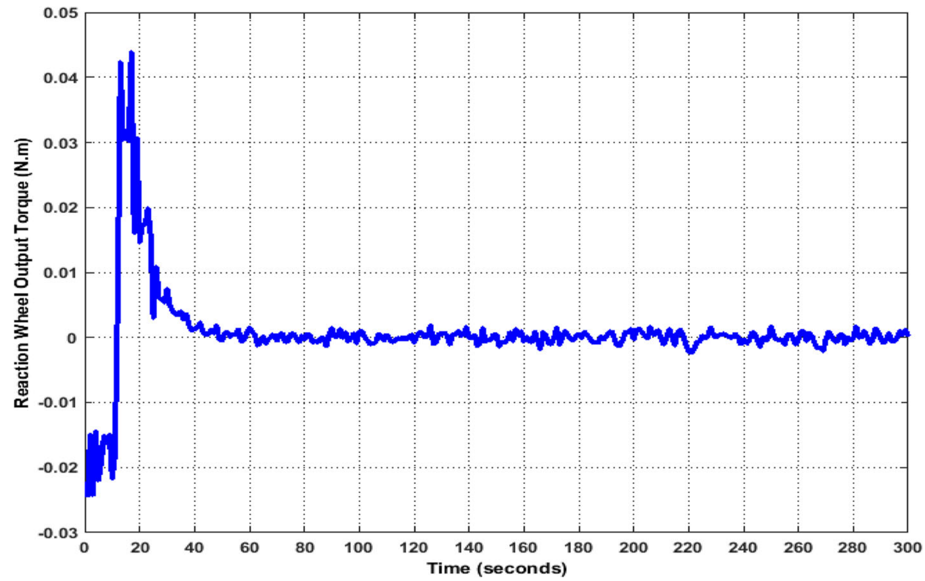

Fig. 5 Reaction wheel output torque

optimization process to fail. Danial J et al. [44] suggested that the population size is regularly changed from 20 to 50 particles. In addition, to choose the proper population size, five experiments have been done. Each experiment was executed 20 times with a most extreme number of cycles up to 3000. Further, the five experiments were analyzed in terms of standard deviation (STD) of the fitness values, mean of best fitness values, and success rate (SR)

The SR is computed as follows:

$$SR = \frac{NSR}{TNR} * 100\% \tag{20}$$

where NSR addresses the number of successful runs and TNR is the total number of runs, which are 20 runs in runs in all the experiments. Besides, it is considered for the single run to be effective at the end of 3000 iterations in a manner that MSE ≈6.5E−7. The results acquired from these trials are given in Table 3 and Fig. 6. As per the results in Table 3, it can be observed that the QPSO algorithm with 20 particles has a 50% success rate. The STD of the best fitness value for the four cases is around something similar. It can be observed from Fig. 6 that the four cases have approximately the same average fitness values. Although, QPSO with 50 particles has a slightly fast convergence speed, but expanding the population size will increment the computational time and the calculation intricacy. Consequently, we chose the population size to be 20 particles.

Table 2 QPSO-trained RBFNN performance at different hidden layer neurons

Hidden layer neurons	Performance, mean square error (MSE)
1	1.87E−05
2	**6.54E−07**
3	6.44E−07
4	6.32E−07
5	6.44E−07

Experiments and analysis

Evaluation criteria

In this subsection, the benchmark results of various algorithms including EQPSO-2M were compared in terms of convergence speed, mean best fitness, STD, SR, Minimum best fitness, and Maximum best fitness. Moreover, every one of the outcomes is tested with a nonparametric statistical investigation utilizing Wilcoxon rank-sum test. To investigate the efficiency of the proposed approach for optimizing the RBFNN parameters, it is compared with other optimization algorithms. These algorithms incorporate the TAPSO [21], Modified PSO (MPSO) [20], Autonomous Groups PSO (AGPSO) [22], enhanced leader PSO (ELPSO) ELPSO [23], modified PSO with inertia weight coefficient (PSO-In) [45], and the traditional QPSO algorithm [39]. Moreover, the proposed scheme is compared with an enhanced QPSO algorithm (EQPSO-1M) that was developed during this research dependent on diversity control and just a one mutation strategy. To ensure the fairness of the comparison, every one of the outcomes is gotten dependent on the results of 30 free experiments through 2000 cycles. Meanwhile, all tests are done utilizing a similar PC and with similar conditions. MATLAB 2019B software is utilized for creating and testing during every one of the investigations.

Parameter settings

The problem dimension is set to seven variables that represent the proposed RBFNN architecture. All the control parameters of the algorithms are chosen by the suggestions from the original literature. Concerning the TAPSO algorithm [21], the acceleration coefficients c_1 and c_2 are refreshed by Eqs. 11 and 12. The values of $c_{\max}$ and $c_{\min}$ are set to 2.5 and 0.5, respectively. In the MPSO algorithm [20], c_1 and c_2 are updated during the search process utilizing Eqs. 14 and 15. $c_{1\max}$ is set to 2.25, and $c_{1\max}$ = 1.25, $c_{2\max}$ is set to 2.55, and $c_{2\min}$ is set to 0.5. The inertia weight coefficient w is diminished linearly from 1 to 0.4. As to the AGPSO algorithm [22], the particles are divided into groups, where c_1 and c_2 for each group are refreshed by Table 4: In Table 4, T represents the greatest number of iterations, and t shows the current iteration.

Besides, the inertia weight parameter w is diminished step by step from 0.9 to 0.4. In the ELPSO calculation $c_1= c_2 = 2$, w is decreased linearly from 0.9 to 0.4. the STD of the gaussian mutation is set to 1, and the scale parameter of Cauchy mutation is 2. In PSO with inertia weight coefficient, the inertia weight coefficient is set to 0.5, $c_1= c_2=$ 1.5. Therefore, the values of the coefficients are updated according to the following equation [45]:

Table 3 Effect of swarm size change on QPSO-RBFNN performance

Swarm size	Iterations	Mean best fitness value	STD	Success rate, %
20	**3000**	**4.1130E−06**	**4.8996E−06**	**50%**
30	3000	3.6148E−06	4.1460E−06	30%
40	3000	3.4110E−06	4.5874E−06	25 %
50	3000	3.0094E−06	4.6556E−06	30%

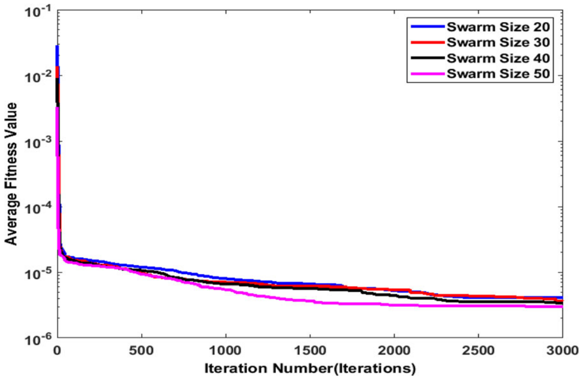

Fig. 6 Convergence speed of QPSO algorithm at different population size

$$c_1 + c_2 < \frac{24\left(1 - \omega^2\right)}{7 - 5\omega} \tag{21}$$

For the QPSO algorithm, the upper and lower limits of the search interval are [-5, 5], and the threshold of the diversity is set to 1E-05 as experimentally seen from the simulation.

Discussion

The statistical results acquired by the proposed EQPSO-2M and different algorithms are introduced in Table 5. As can be acquired from the results in Table 5, the proposed EQPSO-2M outperforms all other peers in terms of Max best fitness value, mean best fitness value, STD, and SR. In the meantime, the SR of 96.7% at the proposed algorithm shows that the algorithm converges to the global minima at 29 of the 30 experiments. Besides, the STD results demonstrate the higher stability of the proposed algorithm in optimizing the RBFNN parameters. It very well may be seen from Table 5 that the proposed algorithm outperforms any remaining peers by 1.6E-07 of STD. The second-best outcomes in terms of STD, SR, mean best fitness, and the Max best fitness are acquired by the EQPSO-1M algorithm. It is obvious that applying the diversity control and a single mutation has further improved the SR of the EQPSO-1M by 43% compared with conventional QPSO. However, using the diversity control and two progressive mutations have enhanced the SR of the EQPSO-2M by 53%. The third best outcomes in terms of SR and mean best fitness are accomplished by PSO-In, TAPSO, and QPSO algorithms, respectively. Therefore, in light of STD, EQPSO-2M is positioned one followed by EQPSO-1M, AGPSO, MPSO, ELSPO, TAPSO, PSO-In, and QPSO, respectively. In terms of the mean best fitness, the best four outcomes are accomplished by EQPSO-2M, EQPSO-1M, PSO-In, and TAPSO, individually. The ELPSO and MPSO algorithms have a similar SR of ≈13 %. The improvement in the results of the PSO-In is because of the legitimate determination of the PSO control parameters ω, c_1, and c_2. Besides, the TAPSO algorithm profits by the dramatic time-fluctuating acceleration coefficients that enhanced the exploration in the beginning stage and the exploitation in the later period of the search cycle. The fundamental justification of the bad outcomes got by the ELPSO algorithm is the utilization of constant values for c_1 = c_2. Hence, the algorithm fails to make a balance between global and local searching stages, and it is

Table 4 AGPSO coefficients updating strategies

Group	c_1	c_2
Group 1	$1.95-2t^{1/3}/T^{1/3}$	$2t^{1/3}/T^{1/3}+0.05$
Group 2	$(-2t^3/T^3)+2.5$	$(2t^3/T^3)+0.5$
Group 3	$1.95-2t^{1/3}/T^{1/3}$	$(2t^3/T^3)+0.5$
Group 4	$(-2t^3/T^3)+2.5$	$2t^{1/3}/T^{1/3}+0.05$

caught in local minima at 26 experiments. Moreover, the observed outcomes reveal that the AGPSO and MPSO algorithms failed to accomplish a decent harmony among exploration and exploitation. In this way, the algorithms get trapped in most of the experiments and have the most exceedingly awful mean best fitness.

The convergence speed is a significant factor that can be utilized to assess optimization problems. Therefore, to additionally assess the performance of the proposed EQPSO-2M strategy, its convergence speed is compared with different algorithms. Figure 7 shows the average convergence curves that are plotted in a logarithmic scale for the proposed EQPSO-2M and different algorithms. As can be seen from Fig. 7, that the average fitness of the EQPSO-2 M algorithm is the minimum. In addition, the EQPSO-2M algorithm shows the best convergence speed compared with different peers. It converges to an average fitness value of ≈ 6.5E-07 at around 300 iterations. Besides, EQPSO-1M has the convergence speed after the proposed algorithm. Thanks for applying the diversity control and the single mutation. Based on the convergence speed, the second-best result is obtained by the PSO-In algorithm. This improvement in the results of the PSO-In ensures strong relation between the PSO convergence behavior and the control parameters selection. The acquired results demonstrate that the other algorithms (ELSPO, AGPSO, QPSO, MPSO, and TAPSO) show less performance and they have a slow convergence to the optimal minimum value. These algorithms have a delay in converging to the minimum best values due to stagnation conditions. The main reason for the high convergence rate of the proposed EQPSO-2M is that the utilization of the two successive mutations mechanism helps the swarm in each iteration to explore the search space extensively near the personal best position to find more best positions. In addition, the utilization of diversity control improves the diversity of the particles. Consequently, it reveals a better performance and more efficacy in jumping out from local minima in case of stagnation, and hence obtaining more high-

Table 5 Comparison results between EQPSO-2M and other algorithms

Algorithm	Minimum best fitness	Maximum best fitness	Mean best fitness	STD	Success rate, %
TAPSO	6.7E−07	0.3229	4.4E−06	6.1E−06	63.3
PSO-In	6.5E−07	0.4624	3.9E−06	6.4E−06	73.3
MPSO	6.6E−07	0.6003	1.0E−05	4.8E−06	13.3
AGPSO	6.7E−07	0.4677	8.4E−06	3.9E−06	20
ELPSO	7.0E−07	0.1998	1.1E−05	5.8E−06	13.1
QPSO	6.6E−07	0.3977	6.2E−06	6.5E−06	43.3
EQPSO-1M	6.5E−07	0.0393	3.2E−06	1.8E−06	86.7
EQPSO-2M	**6.5E−07**	**0.0088**	**6.9E−07**	**1.6E−07**	**96.7**

quality positions. Consequently, the proposed EQPSO-2M achieves a better performance in terms of exploration and exploitation.

From Table 5 and Fig. 7, the proposed EQPSO-2M algorithm has the best results compared to all other algorithms in terms of convergence accuracy and speed. The EQPSO-2M benefits from the diversity control and mutation strategies that allow the QPSO algorithm to generate a better global search ability and converge faster than the other algorithms. Moreover, the proposed algorithm achieves a good balance between exploration and exploitation. Thus, it has an efficient performance that allows escaping from the local minimum for finding the optimal parameters of the RBFNN-based reaction wheel model.

Wilcoxon rank-sum test analysis

Wilcoxon rank-sum test is a non-parametric test strategy of the t-test for two independent samples. It is utilized primarily to test that there are differences between two groups of samples. Moreover, it is utilized to test the invalid speculation that two samples are procured from a continuous distribution with equivalent means [46]. Additionally, utilizing the mean and STD values for assessing the performance of the proposed algorithm compared with different algorithms might be questionable. To determine this issue, the Wilcoxon rank-sum test as a nonparametric test method is utilized as evidence that the results of the EQPSO-2M mechanism are not the same as those of different mechanisms. The significance level α is set to 0.05. In the interim, if the p-value is greater than 0.05, this implies that there is no huge distinction between the results of the two algorithms [47]. Something else, if the p value is lower than the significance level α, it implies that there is a huge contrast between the results of the compared algorithms. Table 6 shows the p values got by Wilcoxon rank total test at 0.05

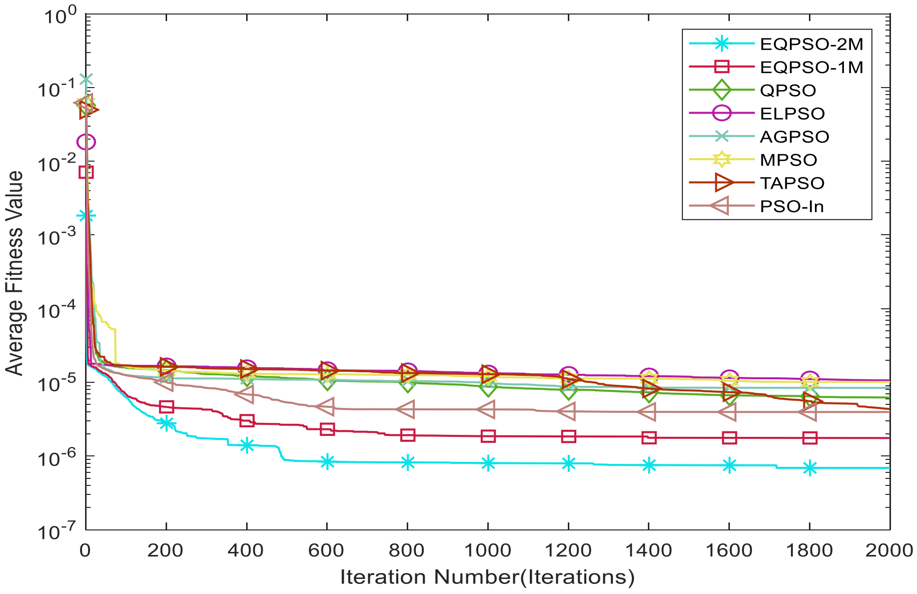

Fig. 7 Average fitness curves for different algorithms

significance level of EQPSO-2M outcomes against the consequences of QPSO-1M, QPSO, ELSPO, AGPSO, MPSO, TAPSO, and PSO-In.

As displayed in Table 6, the *p* values that are lower than 0.05 show the predominance of the proposed algorithm. Notwithstanding, there is no huge contrast between the proposed algorithm and the PSO-In with coefficients controlled by [45]. However, the proposed EQPSO-2M algorithm reveals better performance in terms of convergence speed, SR, stability, and the mean best fitness.

Modeling scheme performance evaluation

Based on the results of the proposed EQPSO-2M algorithm, three RBFNN-based Reaction Wheel models have been created for the spacecraft roll, pitch, and yaw axes. In the meantime, the global best positions of EQPSO-2M represent the optimal values of the RBFNN coefficients. To evaluate the performance of the proposed models, they were tested for various tilting angles. Figure 8 shows the outputs of the developed RBFNN-based RW models compared with the actual reaction wheel outputs for 10° roll, pitch, and yaw tilting angles. Furthermore, from Fig. 8, it very well may be seen that there is a good agreement between the models' outputs and the actual RW outputs. Moreover, the MSE error is about 3.9E−07 for roll, 3.5E−07 for pitch, and 5.9E−07 for yaw angles. Moreover, Fig. 8 reveals the superior matching between the models' outputs and the actual RW outputs at various working conditions.

Modeling scheme generalization evaluation

To explore the generalization of the proposed modeling scheme, the performance of the developed models was tested for tilting angles in the scope of [− 90°, 90°] for roll, pitch, and yaw. Figure 9 shows how the mean square errors between the models' outputs and the actual outputs change with the tilting angles. The results show that the three models can foresee the RW output torque with high precision. The MSE during the interval of [− 20°,20°] is about 6E−7 for roll, pitch, and yaw. It increases to arrive at 4E−6 at the interval limits. These tiny MSE values show the predominant presentation of the proposed RW models. Furthermore, Fig. 9 shows the generalization ability of the proposed modeling scheme that has demonstrated the ability of the models to work in a wide working scope of tilting angles with a high pointing accuracy.

Table 6 *p*-values of Wilcoxon rank sum test comparison between the results of EQPSO-2M and other algorithms

EGPSO-2M Vs.	***p* value**
EGPSO-1M	0.0138
QPSO	6.11E−10
ELSPO	4.50E−11
AGPSO	7.39E−11
MPSO	8.99E−11
TAPSO	6.71E−10
PSO-In	0.0574

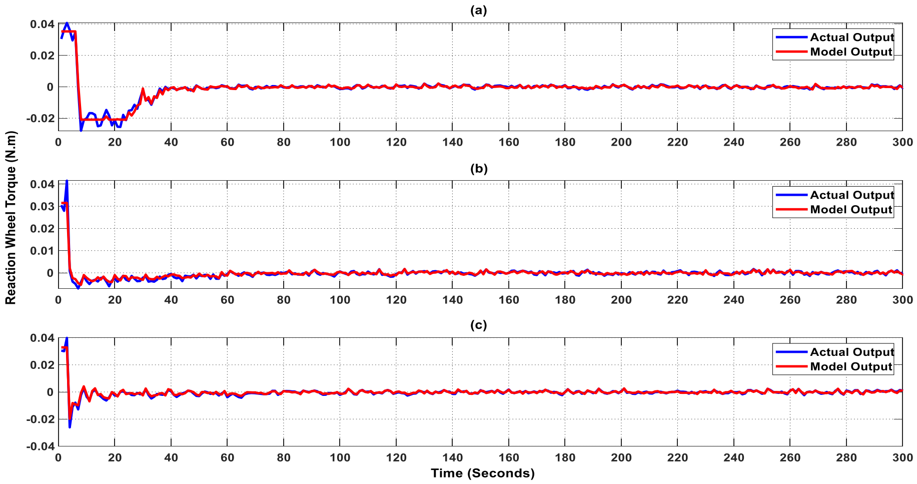

Fig. 8 Output of reaction wheel model compared to the actual output at 10° (**a**) Roll output, (**b**) Pitch output, (**c**) Yaw output

Conclusions

This research proposes another modeling scheme for the spacecraft reaction wheel utilizing RBFNN with an enhanced version of QPSO. In light of the principles of the diversity control and mutation strategy, EQPSO-2M is proposed to ameliorate the RBFNN parameters. In this way, the estimation of the RBFNN parameters is demonstrated as an optimization problem that was settled in terms of the EQPSO-2M algorithm. Additionally, the performance of the proposed algorithm was compared with other strategies like ELPSO, AGPSO, PSO-In, MPSO, TAPSO, and the conventional QPSO algorithm. Statistical benchmark rules dependent on the SR, convergence speed, and

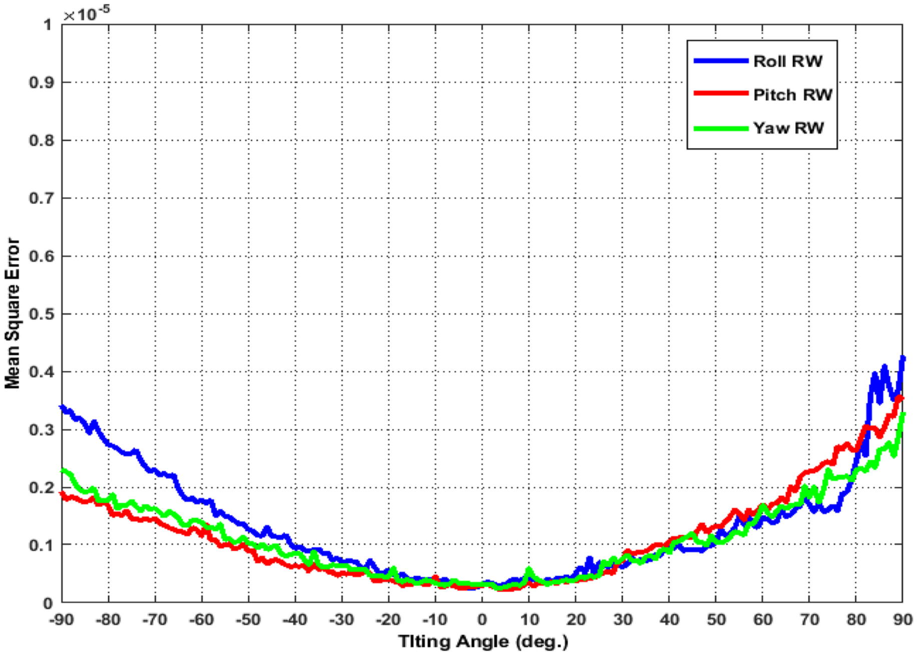

Fig. 9 Performance of the developed RW models measured at tilting angle from − 90° to 90° roll, pitch, and yaw

stability have shown the superiority and effectiveness of the proposed EQPSO-2M. Thanks to the EQPSO-2M algorithm for efficient performance and to accurately find the best particles' positions. Consequently, further improving the global search ability, helps the particles from stagnation in nearby optima and overcomes the premature convergence of the conventional QPSO.

In addition, the simulation results revealed that the proposed EQPSO-2M has a superior performance in terms of stability, mean best fitness value, SR, and convergence speed. Moreover, three RBFNN-based reaction wheel models that are roll, pitch, and yaw were developed and then validated with MATLAB mathematical model. Extensive simulation has been done to evaluate the models' performance. Therefore, the very small value of MSE, which is close to 6.5E−7 indicates a distinct performance and stability of the proposed modeling scheme.

To further investigate the generalization of the proposed reaction wheels' models, they were tested for roll, pitch, and yaw angles in the range of [− 90°, 90°]. The superiority of the proposed approach additionally emanates from the MSE value, which is approximately proximate to 4E−6. Thus, the efficiency of testing results proves the capability of the proposed RBFNN modeling scheme. In fact, the EQPSO-2M algorithm is an efficient mechanism for optimizing the RBFNN parameters. Furthermore, the proposed modeling scheme is considered to be superior for modeling dynamic systems like spacecraft reaction wheels. It is recommended for the future work to utilize the developed model for the detection and identification of reaction wheel faults.

Abbreviations

ACS: Attitude Control Subsystem; RBFNN: Radial Basis Function Neural Network; QPSO: Quantum Behaved Particle Swarm Optimization; EQPSO-2M: Enhanced QPSO Two Mutation; EQPSO-1M: Enhanced QPSO One Mutation; RW: Reaction Wheel; AI: Artificial Intelligence; FDI: Fault Detection and Identification; PSO: Particle Swarm Optimization; AGPSO: Autonomous Groups Particle Swarm Optimization; PID: Proportional Integral Derivative; MSE: Mean square Error; MPSO: Modified PSO; ELPSO: Enhanced leader PSO; STD: Standard deviation; SR: Success Rate

Acknowledgements

Not applicable.

Authors' contributions

The modeling scheme is proposed by W.A. M and A.R.A. A.R.A. has implemented the scheme, verified the results, and written the manuscript. M.I.Y. was the overall supervisor of the research and was a major contributor in writing the manuscript. All authors have read and approved the final manuscript.

Funding

Not applicable

Availability of data and materials

Not applicable

Declarations

Competing interests

The authors declare that they have no competing interests.

Author details

[1]National Authority for Remote Sensing and Space Sciences (NARSS), 23 Jozeph Tito St., Cairo, Egypt. [2]Department of Electrical Engineering, Faculty of Engineering, Al-Azher University, Cairo, Egypt.

Received: 16 September 2021 Accepted: 30 November 2021
Published online: 10 January 2022

References

1. Rahimi A, Saadat A (2020) Fault isolation of reaction wheels onboard three-axis controlled in-orbit satellite using ensemble machine learning. Aerosp Syst 3(2):119–126. https://doi.org/10.1007/s42401-020-00046-x
2. Afram A, Farrokh J (2015) Black-box modeling of residential HVAC system and comparison of gray-box and black-box modeling methods. Energ Buildings 94:121–149. https://doi.org/10.1016/j.enbuild.2015.02.045

3. Al-Zyoud I, Khorasani K (2006) Neural network-based actuator fault diagnosis for attitude control subsystem of an unmanned space vehicle. In: Proceeding Of 2006 IEEE International Joint Conference on Neural Network Proceedings, Vancouver, BC Canada, pp 3686–3693. https://doi.org/10.1109/IJCNN.2006.247383
4. Li Z, Ma L, Khorasani K (2006) dynamic neural network-based reaction wheel fault diagnosis for satellites. In: Proceeding Of 2006 International Joint Conference on Neural Networks Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada, pp 3714–3721. https://doi.org/10.1109/IJCNN.2006.247387
5. Mousavi S, Khorasani K (2014) Fault detection of reaction wheels in attitude control subsystem of formation flying satellites: A dynamic neural network-based approach. Int J Intell Unmanned Syst 2(1):2–26. https://doi.org/10.1108/IJIUS-02-2013-0011
6. Vahid I et al (2017) Supervisory algorithm based on reaction wheel modelling and spectrum analysis for detection and classification of electromechanical faults. IET Sci Meas Technol 11(8):1085–1093
7. Mba CU et al (2017) Fault Diagnosis in Flywheels: Case Study of a Reaction Wheel Dynamic System with Bearing Imperfections. Int J Performability Eng 13(4):362–373
8. Franceso S, Daniel A et al (2018) A novel Dynamic Model of a Reaction Wheel Assembly for High Accuracy Pointing Space Missions. In: Proceedings of ASME Dynamic Systems and Control Conference, Atlanta, Georgia, USA. https://doi.org/10.1115/DSCC2018-8918
9. Rajesh K, Smriti S, Gupta1 (2016) Modeling and adaptive control of nonlinear dynamical systems using radial basis function network. Soft Comput 21(15):4447–4463. https://doi.org/10.1007/s00500-016-2447-9
10. Xie Y, Yu J, Xie S, Huang T, Gui W (2019) On-line prediction of ferrous ion concentration in goethite process based on self-adjusting structure RBF neural network. Neural Netw 116:1–10. https://doi.org/10.1016/j.neunet.2019.03.007
11. Ali N, Noor M, Mohammed F, Ahmed E (2018) RBF-NN-based model for prediction of weld bead geometry in Shielded Metal Arc Welding (SMAW). Neural Comput Applic 29:889–899. https://doi.org/10.1007/s00521-016-2496-0
12. Linag et al (2020) Radial Basis Function Neural Network for prediction of medium frequency sound absorption coefficient of composite structure open-cell aluminum foam. Appl Acoust 170:107505. https://doi.org/10.1016/j.apacoust.2020.107505
13. Ye Y, Qi Y, Shi D, Sun Y, Zhou Y, Hecht M (2020) Rotary-scaling fine-tuning (RSFT) method for Optimizing RAILWAY WHEEL profiles and its application to a locomotive. Railw Eng Sci 28(2):160–183. https://doi.org/10.1007/s40534-020-00212-z
14. Ye Y, Vuitton J, Sun Y, Hecht M (2021) Railway wheel profile fine-tuning system for profile recommendation. Railw Eng Sci 29(1):74–93. https://doi.org/10.1007/s40534-021-00234-1
15. Qi Y, Dai H, Wu P, Gan F, Ye Y (2021) RSFT-RBF-PSO: A RAILWAY WHEEL Profile optimisation procedure and its application to a metro vehicle. Veh Syst Dyn:1–21. https://doi.org/10.1080/00423114.2021.1955135
16. Shanshan TU et al (2020) A novel quantum inspired particle swarm optimization algorithm for electromagnetic applications. IEEE Access 8:21909–21916. https://doi.org/10.1109/ACCESS.2020.2968980
17. Xin-gang Z, Liang J et al (2020) An improved quantum particle swarm optimization algorithm for environmental economic dispatch. Expert Syst Appl 152:113370. https://doi.org/10.1016/j.eswa.2020.113370
18. Shi Y, Eberhart R (1998) A modified Particle Swarm Optimizer. In: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, 69-73
19. Z. Cui, J. Zeng and Y. Yin (2008) An Improved PSO with Time-Varying Accelerator Coefficients. 2008 Eighth International Conference on Intelligent Systems Design and Applications, Kaohsuing, Taiwan,638-643 https://doi.org/10.1109/ISDA.2008.86.
20. G. Q. Bao and K. F. Mao (2009) Particle swarm optimization algorithm with asymmetric time varying acceleration coefficients. 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO), Guilin, China, 2134-2139 https://doi.org/10.1109/ROBIO.2009.5420504.
21. T. Ziyu and Z. Dingxue, et al (2009) A Modified Particle Swarm Optimization with an Adaptive Acceleration Coefficients. 2009 Asia-Pacific Conference on Information Processing, Shenzhen, China, 330-332 https://doi.org/10.1109/APCIP.2009.217.
22. Mirjalili S et al (2014) Autonomous Particles Groups for Particle Swarm Optimization. Arab J Sci Eng 39(6):4683–4697. https://doi.org/10.1007/s13369-014-1156-x
23. Jordehi AR (2014) Enhanced leader PSO (ELPSO): A new PSO variant for solving global optimization problems. Appl Soft Comput 26:401–417. https://doi.org/10.1016/j.asoc.2014.10.026
24. Sun J, Feng B, Xu W (2004) Particle Swam Optimization with Particles Having Quantum Behavior. Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753), Portland, OR, USA, USA, 325-331.
25. Kun Y (2018) Quantum-Behaved particle swarm optimization for far-distance rapid cooperative rendezvous between spacecraft. Adv Space Res 62(11):2998–3011
26. Tianyu L (2016) Cultural quantum-behaved particle swarm optimization for environmental/economic dispatch. Appl Soft Comput 48:597–611. https://doi.org/10.1016/j.asoc.2016.04.021
27. Bialke B (1998) High fidelity mathematical modeling of reaction wheel performance, In: Annual Rocky Mountain guidance and control conference; 21st, Guidance and control, Breckenridge.
28. Yaguang Y (2019) Spacecraft modeling, Attitude determination, and control quaternion-based approach. CRC Press, Taylor & Francis Group, U.S.Ahttps://doi.org/10.1201/9780429446580
29. Omran EA, Murtada W (2019) An Efficient anomaly classification for spacecraft reaction wheels. Neural Comput Applic 31(7):2741–2747. https://doi.org/10.1007/s00521-017-3226-y
30. Afshin R, Krishna D et al (2020) Fault Isolation of Reaction Wheels for Satellite Attitude Control. IEEE Trans Aerosp Electron Syst 56(1):610–629. https://doi.org/10.1109/TAES.2019.2946665
31. Broomhead D S, David L (1988) Radial Basis Functions Multi-Variable Functional Interpolation and Adaptive Networks. Complex Systems. 2:321-355
32. Ortombina L, Tinazzi F, Zigliotto M (2017) Magnetic Modelling of Synchronous Reluctance and Internal Permanent Magnet Motors Using Radial Basis Function Networks. IEEE Trans Ind Electron 65(2):1140–1148. https://doi.org/10.1109/TIE.2017.2733502

33. Yadav AK (2017) Daily array yield prediction of grid-interactive photovoltaic plant using relief attribute evaluator based Radial Basis Function Neural Network. Renew Sustain Energy Rev 18:2115–2127. https://doi.org/10.1016/j.rser.2017.06.023
34. Alexandridis A, Chondrodima E, Sarimveis H (2013) Radial Basis Function Network Training Using a Nonsymmetric Partition of the Input Space and Particle Swarm Optimization. IEEE Trans Neural Netw Learn Syst 24:219–230. https://doi.org/10.1109/TNNLS.2012.2227794
35. Kennedy J, Eberhart R (1995) Particle Swarm Optimization. Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, 1942-1948
36. Najjarzadeh M, Sadjedi H (2020) Implementation of particle swarm optimization algorithm for estimating the innovative parameters of a spike sequence from noisy samples via maximum likelihood method 106: 102799
37. Chen K, Fengyu Z et al (2017) A hyprid particle swam optimization with sine cosine acceleration coefficients. Inform Sci 422:218–241. https://doi.org/10.1016/j.ins.2017.09.015
38. Khan S, Yang S, Ur Rehman, Obaid (2017) A Global Particle Swarm Optimization Algorithm Applied to Electromagnetic Design Problem. Int J Appl Electromagn Mech 53(3):451–467. https://doi.org/10.3233/JAE-160063
39. Sun J, Wu X, Palade V, Fang W, Lai C-H, Xu W (2012) Convergence analysis and improvements of quantum-behaved particle swarm optimization. Inform Sci 139:81–103. https://doi.org/10.1016/j.ins.2012.01.005
40. Amandeep S, Mandeep K et al (2020) QPSO-CD: quantum-behaved particle swarm optimization algorithm with Cauchy distribution. Quantum Inf Process 19(10):345. https://doi.org/10.1007/s11128-020-02842-y
41. Liu W, He J, Hongbo S (2017) A cooperative quantum particle swarm optimization based on multiple groups. In: Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Canada
42. Xie Y, Peng L (2021) Differential privacy distributed learning under chaotic quantum particle swarm optimization. Computing 103(3):449–472. https://doi.org/10.1007/s00607-020-00853-2
43. Zurada JM (1992) Introduction to artificial neural systems. St. Paul: West Publishing Company Los Angeles USA.
44. Danial J, Raja S, Koohyar F, Ahmad S (2017) Developing a hybrid PSO–ANN model for estimating the ultimate bearing capacity of rock-socketed piles. Neural Comput Applic 28(2):391–405. https://doi.org/10.1007/s00521-015-2072-z
45. C. W. Cleghorn and A. P. Engelbrecht (2014) Particle swarm convergence: An empirical investigation," 2014 IEEE Congress on Evolutionary Computation (CEC), 2014, 2524-2530, https://doi.org/10.1109/CEC.2014.6900439.
46. Mohammadi D, Abd Elaziz M, Moghdani R, Demir E, Mirjalili S (2021) Quantum Henry gas solubility optimization algorithm for global optimization. Eng Comput. https://doi.org/10.1007/s00366-021-01347-1
47. Joaquin D, Garcia S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18. https://doi.org/10.1016/j.swevo.2011.02.002

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.