Modified Neural Network Method for Stabilizing Multi-Rotor Unmanned Aerial Vehicles

Serhii Vladov[1], Yurii Shmelov[1], Ruslan Yakovliev[1], Alona Khebda[1] and Oksana Brusakova[2]

[1] *Kremenchuk Flight College of Kharkiv National University of Internal Affairs, vul. Peremohy, 17/6, Kremenchuk, Poltavska Oblast, Ukraine, 39605*
[2] *Kharkiv National University of Internal Affairs, L. Landau Avenue, 27, Kharkiv, Ukraine, 61080*

Abstract

The work is devoted to the development of a neural network method for stabilizing multirotor unmanned aerial vehicles in three stabilization angles (roll, yaw, pitch), which is based on a hybrid neural control scheme with an emulator and a controller. A distinctive feature of the developed method from the existing one is the use of a recurrent multilayer perceptron RMLP, which makes it possible to solve the problem under the conditions of an unmanned aerial vehicle flight. To train the neuroemulator, which is based on the recurrent multilayer perceptron RMLP, a gradient training algorithm is applied. The results of training the neural network showed that the use of the recurrent multilayer perceptron RMLP made it possible to reduce the learning process of the neuroemulator, as well as to reduce the error to the level of $10^{-2}…10^{-3}$, which is sufficient to solve the problem of stabilizing multirotor unmanned aerial vehicles. The results of the studies showed a significant reduction in the transition process time, which is less than 1 s, and overshoot of the stabilization angles (roll, yaw, pitch) less than 3...5º, which are acceptable parameters for the flight of multirotor unmanned aerial vehicles. Prospects for further research is a more in-depth study of the influence of random perturbations on the behavior of the RMLP neural network in the stabilization task for multirotor unmanned aerial vehicles.

Keywords

Unmanned aerial vehicles, multirotor, neuroemulator, recurrent multilayer perceptron RMLP, training, gradient method, error, stabilization angles (roll, yaw, pitch)

1. Introduction

Over the past years, the topic of studying unmanned aerial vehicles (UAVs) is becoming increasingly widespread [1, 2]. At the moment, there are many types of UAVs that differ in their functional features, as well as in their application. Of particular interest is the multiengine type UAVs – UAVs with remote control, driven by several air screws located in the same plane [3, 4]. As a rule, in the center of the multi-core UAV, there is avionics, batteries, sensors, etc., and $N > 2$ "rays" is located in the same plane in the corners of the correct N-angle from the center.

Nevertheless, there are a number of technological barriers that restrain the pace of development of the UAV industry [5]. Some of them are associated with material and structural issues: power and energy plants with high specific power and capacity, respectively, powerful processors, a high – precision system of sensors and sensors, are required.

Another part of the barriers is associated with a software-algorithmic unit: tasks of optimal management; possibility of adaptation to unprofitable external influences; machine vision systems for recognizing target objects and detecting obstacles; big data processing and calculation optimization, etc.

COLINS-2023: 7th International Conference on Computational Linguistics and Intelligent Systems, April 20–21, 2023, Kharkiv, Ukraine
EMAIL: ser26101968@gmail.com (S. Vladov); nviddil.klk@gmail.com (Yu. Shmelov); ateu.nv.klk@gmail.com (R. Yakovliev); alenahebda@gmail.com (A. Khebda); advokatbrusakova@gmail.com (O. Brusakova)
ORCID: 0000-0001-8009-5254 (S. Vladov); 0000-0002-3942-2003 (Yu. Shmelov); 0000-0002-3788-2583 (R. Yakovliev); 0000-0003-1917-9509 (A. Khebda); 0000-0001-8616-0424 (O. Brusakova)

Among scientific works, there are a number of articles describing the dynamics and control of a quadrocopter with a modernized design [6, 7]. For example, there are aircraft with a variable traction vector having two working modes: take-off and horizontal flight. This configuration allows you to increase the effectiveness of flights over long distances by reducing the resistance and action of the lifting force.

Among scientific works, there are a number of articles describing the dynamics and management of a quadrocopter with a modernized design. For example, there are aircraft with a variable traction vector having two working modes: take-off and horizontal flight. This configuration allows you to increase the effectiveness of flights over long distances by reducing the resistance and action of the lifting force.

In this work, the problem of optimal control of multi-core UAVs is solved, in the solution of which this or that version of the proportionally-integral-radio regulator (PID-regulator) is almost always applied [8]. Unfortunately, the usual PID-regulator is not able to adapt to the changing conditions of the nonlinear and non-stationary system, and therefore guaranteed to ensure the stability of this system [9].

A promising approach to eliminating the shortcomings of classic regulators is the use of artificial neural networks (ANS) [10, 11]. Using a pre-trained ANS to configure the coefficients of PID-regulator in real time, it is possible to eliminate its shortcomings when used in an environment with external disturbances and ensure the fulfillment of stabilization requirements in any conditions.

2. Related Works

It should be noted that the use of standard positioning sensors (accelerometer, gyroscope) requires special studies due to the fact that the accumulated measurement error can significantly distort the output information. Therefore, most of all existing publications are devoted to the appropriate algorithms for building departments and comparing their effectiveness, including using these sensors. Among them, several groups can be distinguished:

1. The method of building management, based on the theory of Lyapunov, which allows in a certain formulation to achieve asymptotic stability of the aircraft [12, 13].
2. The management algorithm, which is based on the proportionally-integral-radio-radio regulator, the most common method; Its main advantage lies in simplified implementation [14].
3. The third group of methods – energy methods applicable to passive systems with a lack of control influences [15].
4. The fourth is based on visual control based on the processing of video camera images (video cameras), often used on take-off and landing [16, 17].
5. The fifth is based on management using a neural network used in stabilization tasks when searching for optimal regulator parameters [18].
6. The sixth algorithm is based on dynamic feedback, which allows you to divide the studied system into linear and controlled subsystems.

In a separate class of tasks, it is worth highlighting research related to the management of unmanned vehicles. In particular, the problems of preventing clashes for several robotic systems in the group are considered in [19]. Among the works devoted to the topic of management of the group of quadrocopter, one can note the work [20], which presents the solution of the drill problem in which quadrocopters must maintain a given topology.

An unquenchable interest in the research and development of UAV leads to the emergence of new tools for the study of the dynamics of aircraft. A special place is occupied by methods based on the use of neural networks. The tasks in which neural network controllers are used can be divided into two classes: building control for certain flight modes and individual trajectories, and the tasks of stabilization in all or in terms of variables [21].

Significant results were achieved in [22], which developed an algorithm for managing the quadrocopter group. This algorithm contains two-layer neural network controller. The first is used to synthesize the control influences of the leading copter.

The second, in turn, is used to stabilize group flight and works on the basis of data obtained from wireless on-board sensors.

The last controller as input parameters receives the state of the system, and at the output it issues optimal control for movement with a minimum deviation from the trajectory of the leading copter. The

controllers described in the work allow you to take into account aerodynamic effects and external disturbances. Also, this work presents a method for optimizing communication channels between quadrocopters, which uses the graphs theory.

There are also a number of scientific works on the use of the neural network method in the research of the helicopter dynamics. For example, in [23], a hybrid controller consisting of two recurrent neural networks is presented. The work shows that the optimal control of Copter is studied during a separate consideration of various stages of flight, while the general optimal control for the entire flight is not built in the work.

The use of a neural network controller to control the height of the flight is described in [24], which shows a description of the joint work of proportionally-integral-different and neurotic regulators. It should be noted that one of the main features of the algorithm is a quick adaptation to external influences, which is important to achieve the optimal flight of UAVs in real conditions.

The goal of this work is to develop a neural network method for stabilizing unmanned aerial vehicles in order to optimize their flight at the angles of the roll, tanga and training.

3. Methods and materials

The UAV control system with the properties necessary for use in a more complex order control systems (a special control system for special purpose airmobile systems) can be intellectual control systems built on a hybrid control neurocontroller [25, 26].

In [26] the UAV control system functional diagram based on intellectual control is given. The second level control system (distant) in accordance with the given program and on the basis of the information sensors from the navigation system, measuring devices forms a control vector for the first level of autopilot control [27, 28].

The autopilot (AP) solves the problems of controlling the mechanical systems of the UAV, and also ensures the movement of the UAV from one point of space to another in the coordinates issued by the second level of control. AP, in fact, is a neural network regulator, made according to the pattern of a fuzzy neuro-emulator and hybrid neurontroller with feedback (fig. 1) [26, 27].

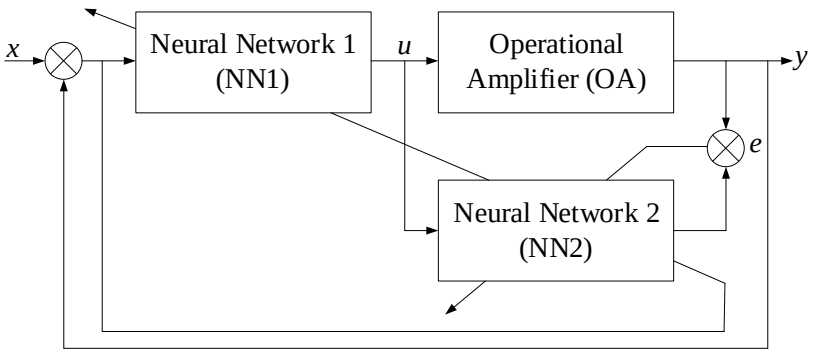

Figure 1: Unmanned aerial vehicles hybrid neural network control diagram [26]

The circuit uses a feedback controller made as a hybrid regulator of the NN1, which is studying through the NN2 identifier. Training through the identifier is necessary not to interfere with the normal functioning of the object by trial actions used for training.

In addition, such a diagram allows you to implement predicate control and increases the safety of the UAV. We assume that as a neural network (NN), in the general case, a dynamic (recurrent) neural network based on the perceptron (RMLP is a recurrent multilayer perceptron), the structure of which is shown in fig. 2 [29].

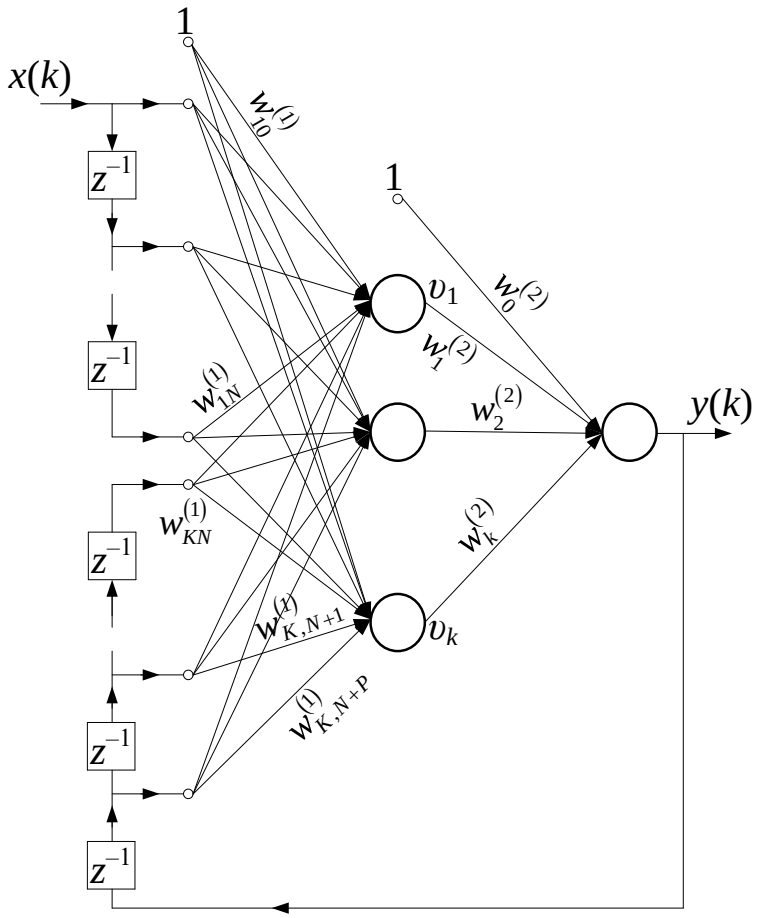

Figure 2: Unmanned aerial vehicles dynamic neural network regulator structure [26, 29] (according to Stanislaw Osowski illustration)

RMLP is a dynamic network characterized by the delay in the input and output signals combined into the input vector of the network is described by the expression:

$$y(k+1) = f\left(x(k), x(k-1), ..., x\left(k-(N-1)\right), y(k-1), y(k-2), ..., y(k-P)\right); \quad (1)$$

where $N-1$ – number of delays in the input signal, P – number of outgoing signal delays. Having accepted K – number of neurons in a hidden layer, the neural network RMLP is characterized by three numbers (N, P, K). The vector **x** submitted to the entrance of the network has the form:

$$x(k) = \left[1,\ x(k), x(k-1), ..., x\left(k-(N-1)\right), y(k-P), y(k-P+1), ..., y(k-1)\right]^T.$$

Suppose that all neurons have a sigmoidal activation function [30]. We denote the u_i suspended sum of the signals of the i-th neuron of the hidden layer, and g – output neuron signals suspended sum. With introduced designations, the output signals of specific neurons are described by dependencies:

$$u_i = \sum_{j=0}^{N+P} w_{ij}^{(1)} x_j; \quad (2)$$

$$v_i = f\left(u_i\right); \quad (3)$$

$$g = \sum_{i=0}^{K} w_i^{(2)} f\left(u_i\right); \quad (4)$$

$$y = f\left(g\right). \quad (5)$$

The neural control diagram with an emulator and controller is developed in [31, 32] and is presented in fig. 3, where a neurocontroller trained on the inverse model of the control object, and the neuroemulator on the real model of the control object (UAV).

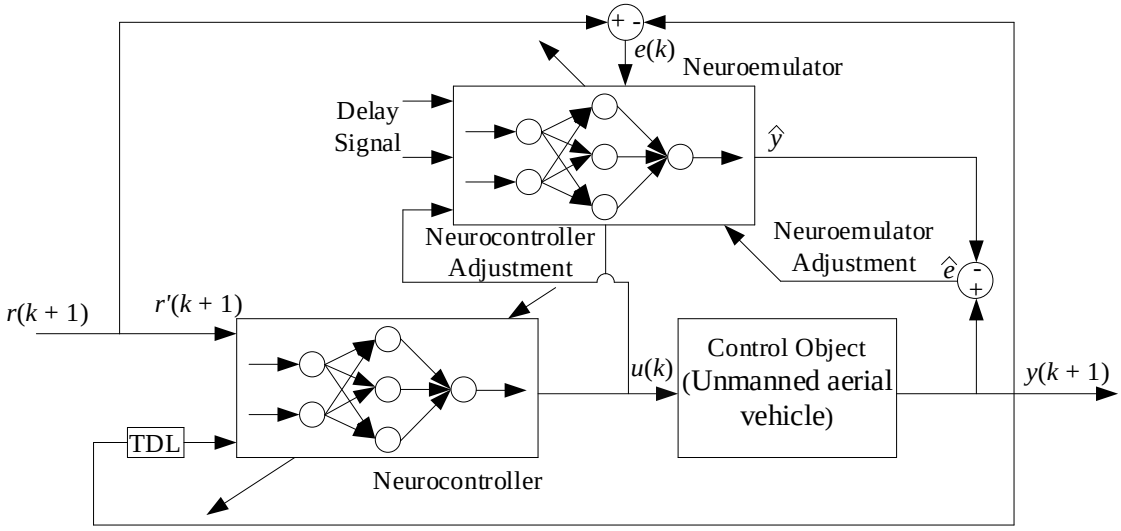

Figure 3: Neural control diagram with an emulator and controller [31, 32]

The neurontroller training method based on a neuroemulator using the method of reverse distribution of the error is described in detail in [31, 32]. For training a neuroemulator that is based on the RMLP neural network, a gradient training algorithm is used [33]. It is worth noting that the gradient training algorithm for a recurrent multilayer perceptron RMLP is systematized by Stanislaw Osowski in the book "Neural networks for information processing". As in the situation with a unidirectional network [31, 32], a gradient of the target function relative to each weight is calculated. For a neural network with one output neuron, the target function at the time t can be determined in the form:

$$E(k)=\frac{1}{2}\big(y(k)-d(k)\big)^2. \tag{6}$$

Differentiating this function with respect to an arbitrary weight $w_\alpha^{(2)}$ ($\alpha = 0, 1, \ldots, K$) of the output layer of the network, we obtain:

$$\frac{\partial E(k)}{\partial w_\alpha^{(2)}}=\big(y(k)-d(k)\big)\frac{\partial y(k)}{\partial w_\alpha^{(2)}}=\big(y(k)-d(k)\big)\frac{\partial f\big(g(k)\big)}{\partial g(k)}\frac{\partial g(k)}{\partial w_\alpha^{(2)}}. \tag{7}$$

Taking into account dependencies (2)–(5), we obtain

$$\frac{\partial E(k)}{\partial w_\alpha^{(2)}}=\big(y(k)-d(k)\big)\frac{\partial f\big(g(k)\big)}{\partial g(k)}\sum_{i=0}^{K}\frac{\partial\big(w_i^{(2)}v_i(k)\big)}{\partial w_\alpha^{(2)}}. \tag{8}$$

where $v_i = f(u_i)$. Derivative $\frac{\partial w_i^{(2)}}{\partial w_\alpha^{(2)}}=1$ only for $i = \alpha$ and $\frac{\partial w_i^{(2)}}{\partial w_\alpha^{(2)}}=0$ in all other cases. Given this fact

$$\frac{\partial E(k)}{\partial w_\alpha^{(2)}}=\big(y(k)-d(k)\big)\frac{\partial f\big(g(k)\big)}{\partial g(k)}\left(v_\alpha(k)+\sum_{i=0}^{K}w_i^{(2)}\frac{\partial v_i(k)}{\partial w_\alpha^{(2)}}\right); \tag{9}$$

and

$$\frac{\partial v_i(k)}{\partial w_\alpha^{(2)}}=\frac{\partial f\big(u_i(k)\big)}{\partial u_i(k)}\sum_{j=0}^{N+P}\frac{\partial x_j}{\partial w_\alpha^{(2)}}=\frac{\partial f\big(u_i(k)\big)}{\partial u_i(k)}\sum_{j=0}^{N+P}w_{ij}^{(1)}\frac{\partial y\big((K-P-1)+(j-N)\big)}{\partial w_\alpha^{(2)}}=\frac{\partial f\big(u_i(k)\big)}{\partial u_i(k)}\sum_{j=0}^{N+P}w_{i,j+N}^{(1)}\frac{\partial y(K-P-1+j)}{\partial w_\alpha^{(2)}}. \tag{10}$$

Taking into account dependencies (6)–(10), we obtain

$$\frac{\partial E(k)}{\partial w_\alpha^{(2)}}=\frac{\partial f\big(g(k)\big)}{\partial g(k)}\left(v_\alpha(k)+\sum_{i=0}^{K}w_i^{(2)}\frac{\partial f\big(u_i(k)\big)}{\partial u_i(k)}\sum_{j=1}^{P}w_{i,j+N}^{(1)}\frac{\partial y(K-P-1+j)}{\partial w_\alpha^{(2)}}\right). \tag{11}$$

The recursive formula (11) makes it possible to calculate the value of the derivative $\frac{\partial y_k}{\partial w_\alpha^{(2)}}$ at an arbitrary moment of time from its values at previous moments. It connects the values of the derivatives

at the moment t with the values of the same functions at the moments $t-1$, $t-2$, $t-P$. It can be assumed that the initial values of the derivatives of the signals before the start of training are equal, that is

$$\frac{\partial y(0)}{\partial w_{\alpha}^{(2)}} = \frac{\partial y(-1)}{\partial w_{\alpha}^{(2)}} = ... = \frac{\partial y(-P)}{\partial w_{\alpha}^{(2)}} = 0.$$

When using the steepest descent method in the training process, the adaptation of the weights of the output layer is determined by the formula

$$\Delta w_{\alpha}^{(2)} = -\eta\left(y(k) - d(k)\right)\frac{\partial y(k)}{\partial w_{\alpha}^{(2)}}. \tag{12}$$

The weights of the hidden layer are updated in a similar way. After calculating the derivative of the signal $y(k)$ with respect to hidden layer weight $w_{\alpha,\beta\beta}^{(1)}$, we obtain

$$\frac{\partial y(k)}{\partial w_{\alpha,\beta}^{(1)}} = \frac{\partial f\left(g(k)\right)}{\partial g(k)}\sum_{i=1}^{K} w_i^{(2)}\frac{\partial f\left(u_i(k)\right)}{\partial u_i(k)}\left(\sum_{j=1}^{P} w_{i,j+N}^{(1)}\frac{\partial y(K-P-1+j)}{\partial w_{\alpha}^{(2)}} + \delta_{i\alpha}x_{\beta}\right). \tag{13}$$

where $\delta_{i\alpha}$ – Kronecker delta.

Therefore, the expression that determines the adaptation of hidden layer weight $w_{\alpha,\beta\beta}^{(1)}$, when using the steepest descent method, takes the form

$$\Delta w_{\alpha\beta}^{(1)} = -\eta\left(y(k) - d(k)\right)\frac{\partial y(k)}{\partial w_{\alpha\beta}^{(1)}}. \tag{14}$$

In its final form, the RMLP network training algorithm is formulated as follows.

1. Perform a random initialization of the weights of neurons in the hidden and output layers.
2. For each moment t with a given excitation in the form of a vector **x**, calculate the state of all neurons in the network in accordance with expressions (2)–(5).
3. Using dependencies (11) and (13), determine the values of derivatives $\frac{\partial y_k}{\partial w_{\alpha}^{(2)}}$ and $\frac{\partial y_k}{\partial w_{\alpha\beta}^{(1)}}$ for all values of α and β corresponding to the weights of the network with the initially chosen structure.
4. Update the weights in accordance with expressions (12) and (14), and then return to step 2 of this algorithm.

The presented algorithm operates in the "online" mode, accepting the incoming input data and the corresponding values of the expected vector d and promptly correcting the values of the weights.

When training an RMLP neural network using the backpropagation method, the training rate η has a decisive influence on the training rate and on the final results obtained. The value of this coefficient in the training process can remain constant or be selected in an adaptive way. Keeping the training rate constant is considered the simplest form of determining η. This method has many disadvantages, including slow convergence, a high probability of process divergence when the value of η is too large, and the ease of hitting local minimum. However, to date, it remains the simplest and most effective method used in online training. Adaptive selection of the coefficient η makes it possible to control training errors, resulting in an increase or decrease in its value. To speed up the training process, a continuous increase in the coefficient η is provided if the level of the actual error compared to the error of the previous iteration is within acceptable limits. If ε_i and ε_{i-1} – adaptation errors at the i-th and (i–1)-th step, and η_i and η_{i-1} – corresponding training coefficients, then in the case (k_w – coefficient of the allowable increase in error), the value is reduced η according to expression [34]:

$$\eta_{i+1} = \eta_i\alpha_d; \tag{15}$$

where α_d – reduction factor for the value of η. Otherwise, when $\varepsilon_i \le k_w\varepsilon_{i-1}$, the value of this coefficient increases according to the expression

$$\eta_{i+1} = \eta_i\alpha_i; \tag{16}$$

where α_i – coefficient of increase in the value of η.

The use of output neurons with a sigmoidal activation function [30] makes it possible to minimize the structure of a recurrent neural network. In the RMLP network of the standard structure described in most literature sources [35, 36], as a rule, output neurons with a linear activation function are used, which makes it easier to bring the signal to any numerical range. Based on [37, 38], studies in the field

of replacing linear neurons with sigmoidal neurons are relevant, which can significantly reduce the dimension of the neural network. Thus, for the network proposed by Narendra [39] and containing linear output neurons, a large number of hidden neurons is needed, for example, $K = 10$. The same effect can be achieved in a network with a sigmoid output neuron and only two hidden neurons. However, it should be taken into account that the signal values of the sigmoid neuron are limited by the interval from –1 to +1. To provide any required range of values, a linear block is added at the output of the network, amplifying the signal by M times ($0 < M < \infty$). With proper selection of the gain M, such a network demonstrates the same good adaptation capabilities with a significantly smaller number of hidden neurons.

4. Experiment

4.1. Description of input data

As the initial data for the training and test samples, the diagram of quadrocopter roll angle fluctuations [40] (fig. 4) is taken. Table 1 contains a fragment of the training sample of the values of the quadcopter roll angle fluctuations. The complete training sample contains 265 rows corresponding to different test modes.

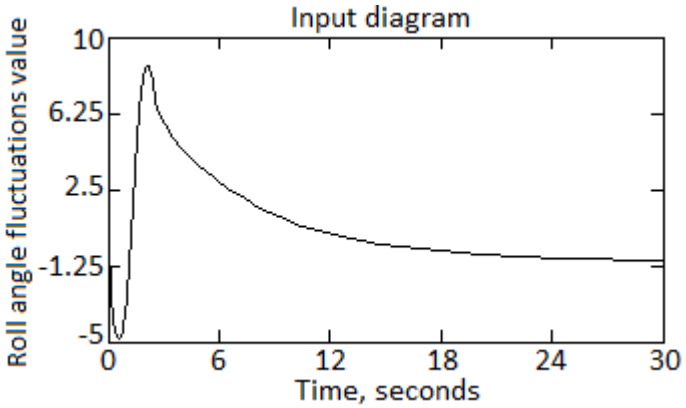

Figure 4: Diagram of quadrocopter roll angle fluctuations [31, 32]

Table 1
Fragment of the training sample

Number	Value	Number	Value
1	–2.455	11	5.126
2	–4.025	12	7.455
3	–4.675	13	8.538
4	–4.836	14	8.646
5	–4.729	15	8.483
6	–4.459	16	7.942
7	–4.133	…	
8	–3.755	…	
9	–2.617	…	
10	1.823	1024	–0.939

One of the main issues addressed at the stage of data analysis is the assessment of the representativeness of the sample, i.e., the completeness of its presentation. The solution of this problem is carried out using the methods of cluster or discriminant analysis [26]. During the clustering process, 10 classes were identified using the Statistica 12.6 package (fig. 5)

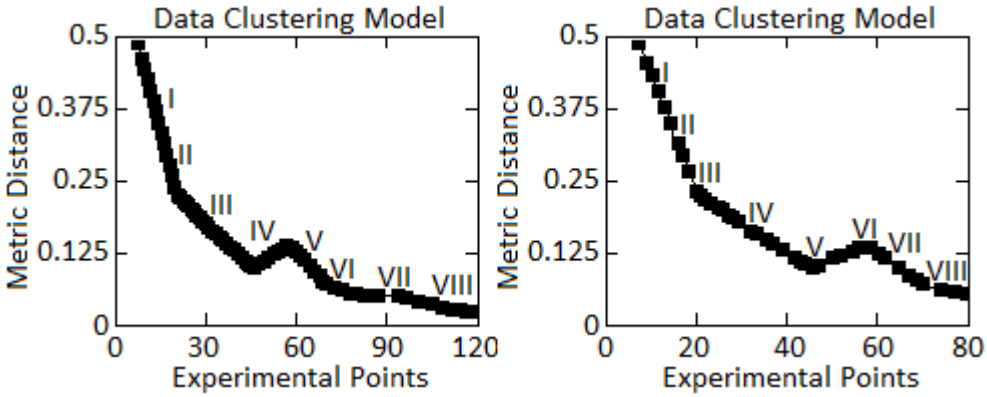

Figure 5: Clustering results: a – initial experimental sample (I…X – classes); b – training sample

After the randomization procedure [26], the actual training (control) and test samples were selected (in a ratio of 2:1, i.e., 67 % and 33 %). The process of clustering the training (fig. 5) and test samples shows that they, like the original sample, contain e_0 classes each. The distances between the clusters practically coincide in each of the considered samples, therefore, the training and test samples are representative. To form the training and test subsets, cross-validation [25] was used to estimate the values of quadrocopter roll angle fluctuations, the results of which are shown in fig. 6.

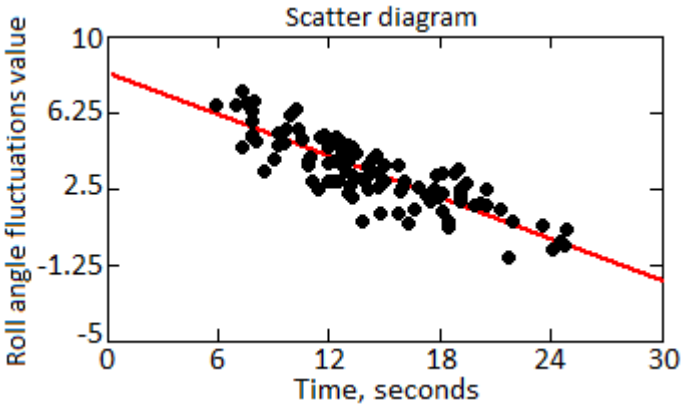

Figure 6: Scatter diagram of input parameters

4.2. Neural network training results

The RMLP recurrent multilayer perceptron was trained using the RMLP program adapted for online training. The training was based on the adaptive identification of non-linear dynamic objects, which include UAVs of multi-rotor types. An object described by a known non-linear function generated a sequence of given signals $d(n)$ as a response to excitation in the form of randomly generated vectors $\mathbf{x}$. An RMLP network with the structure shown in fig. 2 was used as a model for a UAV parameter such as bank angle. As a result of comparing the output signal of this model $y(n)$ with a given signal $d(n)$, the error value $\varepsilon(n)$ was calculated:

$$\varepsilon(n) = y(n) - d(n); \tag{17}$$

controlling the process of refining the parameters of the neural network. Fig. 7 shows the way to turn on the network during experiments, where the symbol M denotes the constant gain of the module that scales the output signal of the network so that its dynamic level lies in the same range as the level of the given signal $d(n)$.

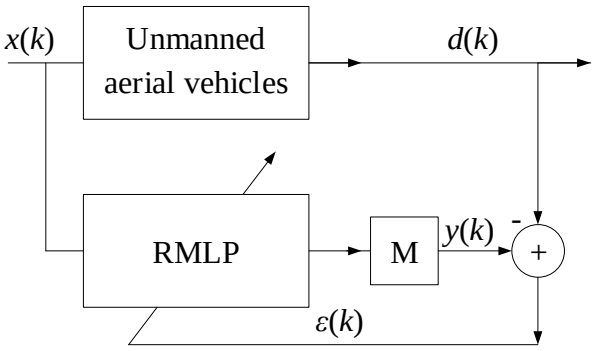

Figure 7: Diagram of switching on the RMLP neural network when solving the identification task

In all numerical experiments, a network with a 3–3–1 structure was used. The system input consisted of one input node $x(n)$, which determines the control error, two nodes, which determine the accumulated control error and the rate of change of the control error. The hidden layer also consisted of three neurons, and the output layer consisted of one neuron. When implementing the training process, the adaptive selection of the training coefficient η described above was performed. The weights were refined in two modes:

1. Single adaptation mode, in which the presentation of each new training sample is accompanied by a single refinement of the values of all network weights and the transition to the next sample.

2. The multiple adaptation mode, in which each training sample caused multiple refinement of the network weights (the presentation of the training sample to the network input was accompanied by a change in the output signal, after which the weight values were refined; change in the output signal with the corresponding refinement of the weights, etc.). Each network training process started with random values of weights uniformly distributed in a given interval. In our experiments, this was the interval (–0.1, 0.1).

The first numerical experiment was a saint with a mathematical model of the quadrocopter roll angle (fig. 4) described by the expression:

$$\varphi(t) = 0.87 \cdot t^8 - 1.38 \cdot t^7 + 1.20 \cdot t^6 - 4.37 \cdot t^5 - 3.99 \cdot t^4 + 8.57 \cdot t^3 + 1.10 \cdot t^2 - 5.97 \cdot t + 11.24. \quad (18)$$

The discrete input signal was given by the function $y(t) = Ae^{-\lambda t}\left(\cos(\omega t + \varphi) + \sin(\omega t + \varphi)\right)$, where $y(t)$ – instantaneous amplitude at time t; A – initial amplitude of the envelope; λ – damping constant, inverse to the units of time along the x axis; φ – phase angle at some arbitrary point; ω – angular frequency.

Fig. 8 shows the form of given signals generated by the dynamic system defined by expression (18). It follows from this equation that the output signal of the system will be limited, provided that the input signal is also constrained. In the experiments, both techniques for refining the weights were used, both single [41, 42] and multiple adaptation [43, 44].

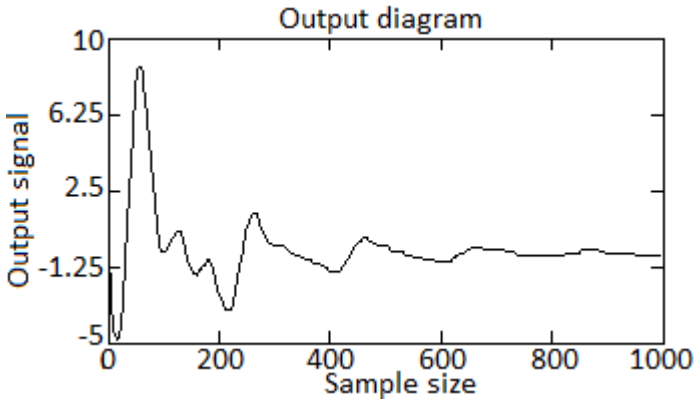

Figure 8: Given signals of a dynamic object (unmanned aerial vehicles) defined by expression (18)

When using the first technique, the values of the weights were refined after the presentation of each training sample according to the steepest descent algorithm with a constant training coefficient

η = 0.075 (adaptive selection of the training coefficient for a single adaptation does not make sense). The results of the training process in the form of changes in error (17) are shown in fig. 9, which indicate that the training error (already after 20 cycles) quickly decreased to an insignificant value, which was perceived only due to the high accuracy of system identification.

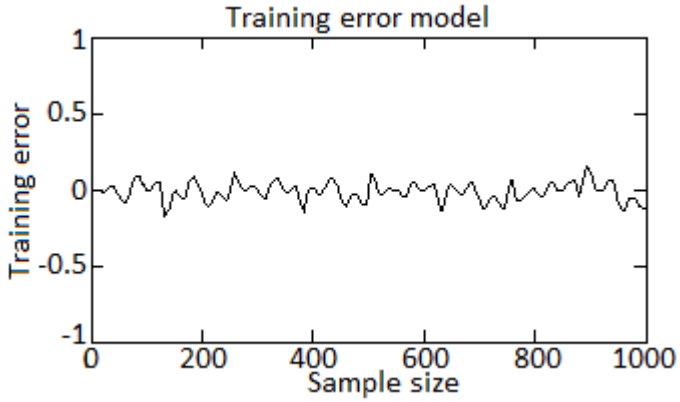

Figure 9: RMLP network training diagram with a single adaptation of the weights in each cycle for a dynamic object (unmanned aerial vehicles) from the first experiment

According to the second method, the weight values were refined three times during each cycle using the adaptive training coefficient η and coefficient values k_d = 0.685 and k_w = 1.029. The diagram of the training error for this case is shown in fig. 10, which shows that the error, especially in the first phase of training, turned out to be smaller, and the process of adapting the model to the reactions of the object proceeded faster, especially at the beginning of training. It should be emphasized that in both the first and second cases, the residual training error has stabilized at a certain, fairly low level, being the driving force behind the mechanism for adapting model parameters.

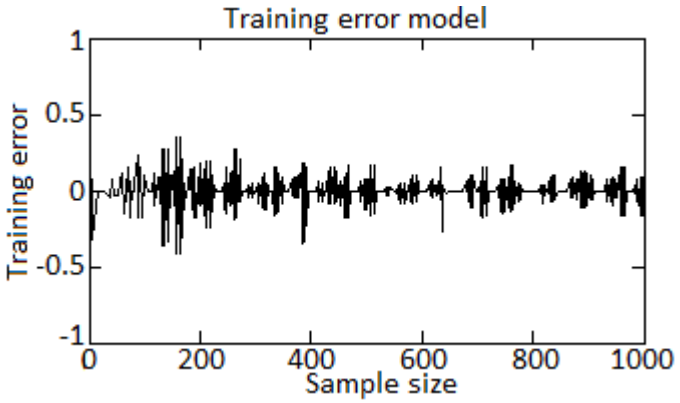

Figure 10: RMLP network training diagram with three times adaptation of weights in each cycle for a dynamic object (unmanned aerial vehicles) from the first experiment

In the second experiment, a non-linear dynamic system (UAV) was studied, while the oscillations of the roll angle are described by the following dependence:

$$y_{k+1} = \frac{y_k y_{k-1}\left(y_k + 2.5\right)}{1 + y_k^2 + y_{k-1}^2} + u_k \tag{19}$$

with input signal $y(t) = Ae^{-\lambda t}\left(\cos\left(\omega t + \varphi\right) + \sin\left(\omega t + \varphi\right)\right)$.

Fig. 11 shows a diagram of the change in the output signal of the object (set values), described by expression (19). The training results in the form of an error diagram for a single adaptation of the weights are shown in fig. 12. The training error, which at the beginning of the process took values of the 4th–5th order, very quickly (in about five cycles) decreased to a residual value that decreases in the training course.

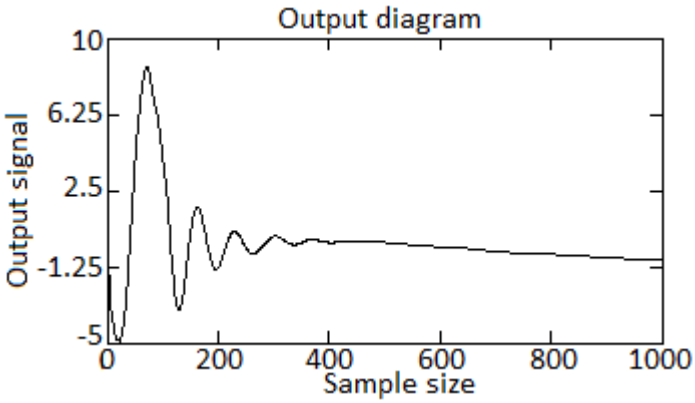

Figure 11: Specified signals of a dynamic object (unmanned aerial vehicles) defined by the expression (19) taking into account (18)

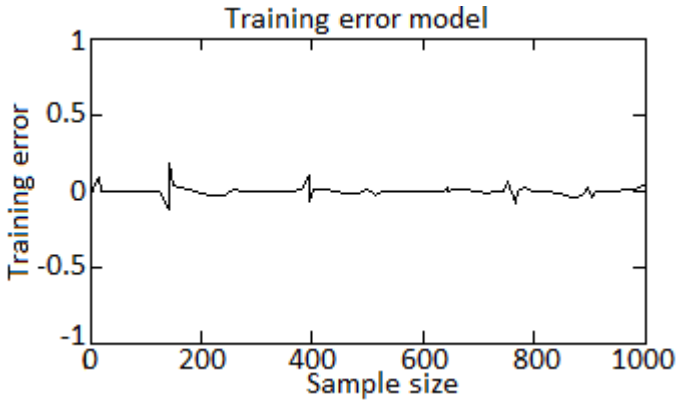

Figure 12: RMLP network training diagram with a single adaptation of the weights in each cycle

The diagram of the training error corresponding to the three-fold adaptation of the weights is shown in fig. 13. The training error with triple adaptation is much less than with a single one, and the training process is shorter and leads to a reduction in the error to the level of $10^{-2}…10^{-3}$, which is sufficient to solve the problem of stabilizing a multirotor UAV.

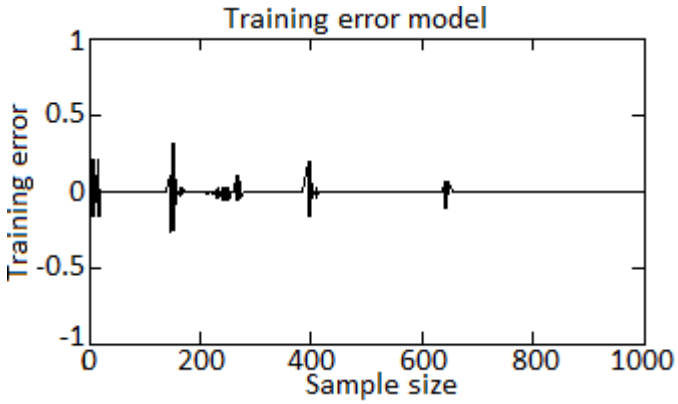

Figure 13: RMLP network training diagram with three times adaptation of weights in each cycle for a dynamic object (unmanned aerial vehicles) from the second experiment

5. Results

In this work, the problem of stabilization in angle is considered and stabilization in height is not taken into account. The PID controller uses error data in the vehicle's pitch angles (roll, pitch, and yaw),

the dynamics of these angles, and the accumulated error. According to [45], each component of the PID controller has its own coefficient and has a different effect on the output signal of the controller:

$$f(t) = K_p \varepsilon(t) + K_i \int_0^t \varepsilon(t)dt + K_d \frac{d\varepsilon(t)}{dt}; \quad (20)$$

where K_P, K_i, K_d – coefficients of the proportional, integral and differential components of the controller, ε – control error (the difference between the desired and actual UAV inclination angle).

To search for the optimal values of the PID controller parameters, a hybrid neurocontrol diagram is used using recurrent multilayer perceptron RMLP and training by the method of error back propagation. This method was chosen based on training on the effectiveness of various diagrams in the tasks of controlling dynamic objects, including UAVs [46].

The network input receives data on UAV operational status: control error ε (the difference between the set point and the actual angle), the accumulated error $\sum\varepsilon$, and the rate of change of the control error $\partial\varepsilon$.

The second, "hidden" layer is used to make the network non-linear. The choice of the number of hidden layers and neurons in these layers strongly depends on the conditions of the task [44], however, as a rule, one hidden layer is sufficient. The number of neurons in this layer is usually the average between the input and output. At the output of the neural network, the coefficients of the PID controller are formed.

The neural network controller consists of three artificial neural networks – one for each stabilization angle (roll, yaw, pitch). The presence of three networks at once is explained by the fact that the dynamics along each of the axes can be different, therefore, it is necessary to apply different coefficients for each direction.

The coefficients for each of the neurons are corrected using the error backpropagation method according to (14). For each weight, the relationship between its change and the change in the final result is calculated. The objective function E is selected, the error of which must be reduced after k iterations:

$$E(y) = \frac{1}{t_k - t_0} \int_{t_0}^{t_k} \varepsilon(t, y)^2 dt; \quad (21)$$

where $\varepsilon(t, y)$ – error between the actual and desired angle; t – number of iterations. The weight is then adjusted for the training factor:

$$w_{i\_corrected} = w_i + \gamma \frac{\delta E}{\delta w_i}; \quad (22)$$

where γ – training coefficient.

Fig. 14 shows the stages of calibration of the coefficients by the network after a different number of iterations for the roll angle of the quadcopter. A stand with one degree of freedom was used as a stabilization object, on which a quadrocopter is installed with an initial roll deviation of 10º. The neural network training procedure and the stabilization algorithm for the UAV's three angles are identical, so only one of these angles is shown on the diagrams.

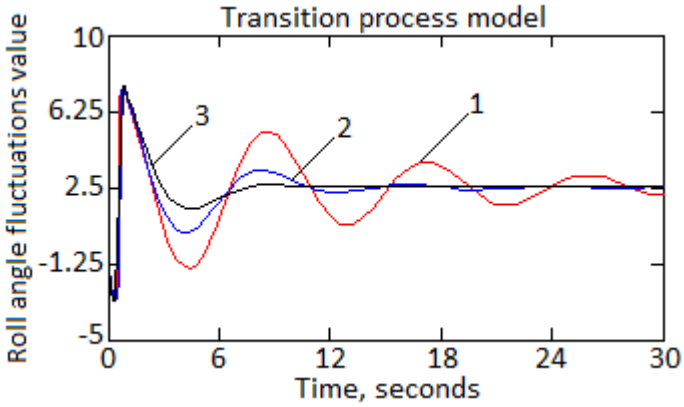

Figure 14: Diagrams of transient processes for the angle of roll during stabilization at various stages of neural network training (1 – configuration 1; 2 – configuration 2; 3 – configuration 3)

To train the neural network, a certain proportional coefficient is initially set, and the values of the integral and differential components are set sufficiently small (they differ from the proportional factor by tens of times). During the training of the neural network, RMLP adjusts the values of the coefficients, changing the stabilization dynamics. The diagrams show the dependence of the angle on the stabilization step. Three configurations of coefficients produced by the RMLP neural network at the training stage are given. As seen in fig. 14, coefficient configuration 1 (neural network start) has a very large oscillation amplitude and overshoot of more than 5…7º; the transition process time is more than 2 s. Configuration 3 (closer to the final result) does a much better job of stabilizing – less than 1 s transient times and less than 3…5º overshoot are acceptable flight parameters.

Fig. 15 shows the behavior of the UAV when adding random perturbations, which are small random changes in the angle at each stabilization step. It can be seen that adding even small additional impacts on the angle increases the time required to stabilize the model. It takes at least a thousand training iterations to bring the model to a stable state.

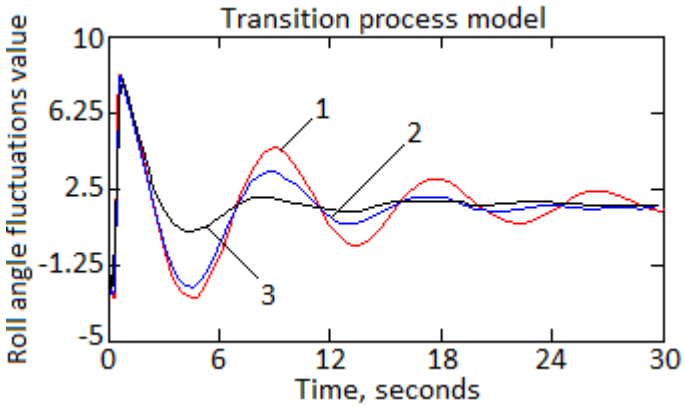

Figure 15: Diagrams of transient processes for the angle of roll during stabilization, taking into account external disturbing influences (1 – configuration 1; 2 – configuration 2; 3 – configuration 3)

Thus, the RMLP neural network is able to optimize the PID controller gains after a certain training period. The final goal of the work is to create a system capable of adapting to changing flight conditions in a short time, while not requiring large computing capabilities. The results obtained can be improved, for example, to reduce the time required to achieve the optimal values of the PID controller. It is also necessary to research more fully the influence of random disturbances on the behavior of the RMLP neural network.

6. Discussions

A comparative analysis of classical and neural network methods for stabilizing unmanned aerial vehicles at three angles, the stabilization angle (roll, yaw, pitch) is given in table 2. At the same time, the perceptron provides an error not exceeding 1.204 %; RMLP neural network – 0.685 %; classical regulator – 2.038 %.

Table 2
Results of comparative analysis

Calculation method	Parameter		
	Roll angle	Yaw angle	Pitch angle
Classical methods (regulators)	1.575	2.038	1.916
Neural network methods (regulators):			
perceptron [46]	1.002	1.204	1.118
RMLP neural network	0.646	0.685	0.667

In order to analyze the stability of neural networks to changes in input data (table 2), additive noise was added to them in relation to the current value of each of the parameters in the form of white noise with zero mathematical expectation and $\sigma_i = \pm 0.01$ (table 3).

Table 3

Results of comparative analysis under conditions of additive noise ($M = 0$, $\sigma_i = \pm 0.01$)

Calculation method	Parameter		
	Roll angle	Yaw angle	Pitch angle
Classical methods (regulators)	4.328	4.619	4.497
Neural network methods (regulators):			
perceptron [46]	2.187	2.312	2.265
RMLP neural network	0.834	0.851	0.849

The results of the analysis of accuracy by three parameters under noise conditions showed the following results: neural network perceptron – 2.312 %; RMLP neural network – 0.851 %; classical regulator – 4.619 %.

In table 4. the probabilities of errors of the 1st and 2nd kind are displayed when determining stabilization angle (roll, yaw, pitch).

Table 4

Results of calculating errors of the 1st and 2nd kind

Method of determination	Probability of error					
	Determination of the roll angle		Determination of the yaw angle		Determination of the pitch angle	
	Type 1st error	Type 2nd error	Type 1st error	Type 2nd error	Type 1st error	Type 2nd error
Classic regulators	1.77	1.24	1.96	1.33	1.99	1.41
Neural network regulators						
perceptron [46]	0.96	0.92	1.03	0.95	1.01	0.93
RMLP neural network	0.63	0.34	0.75	0.41	0.68	0.39

Thus, the use of the RMLP neural network in the control of multirotor unmanned aerial vehicles is a promising direction. The approach proposed in this paper using a hybrid neuroregulator makes it possible to improve the stabilization system of such aircraft.

7. Conclusions

The stabilization method for multi-rotor unmanned aerial vehicles was further developed, which, through the use of recurrent multilayer perceptron RMLP, made it possible to optimize their flight in terms of roll, pitch and yaw angles, and also made it possible to reduce errors of the first and second kind in determining the permissible dynamic errors in the parameters of roll angles, pitch and yaw.

The method of training artificial neural networks in a neural control system with an emulator and a controller was further developed, which, by using the error backpropagation method for training recurrent multilayer perceptron RMLP, made it possible to reduce the neural network training error to 30…35 %, which is acceptable for solving the problem stabilization of unmanned aerial vehicles of multirotor type.

The neural network stabilization controller for multi-rotor unmanned aerial vehicles has been improved, which differs from the existing ones in that due to the use of recurrent multilayer perceptron RMLP with further weight adjustment, taking into account the neural network training factor, it has

made it possible to optimize the transient processes in the roll angle during stabilization, taking into account external disturbing impact on an unmanned aerial vehicle.

It was found that the error using the perceptron neural network did not exceed 1.204 %; RMLP neural network – 0.685 %; classical regulator – 4.619 %.

It has been experimentally confirmed that neural network methods are more robust to external disturbances: for the noise level $\sigma_i = \pm 0.01$, the error increased from 1.204 to 2.312 % using the perceptron neural network; RMLP neural network – 0.685 to 0.851 %; classical regulator – 2.038 to 4.619 %.

Prospects for further research is a more in-depth study of the influence of random perturbations on the behavior of the RMLP neural network in the stabilization task for multirotor unmanned aerial vehicles.

8. References

[1] A. A. Laghari, J. Deussen, A. K. Jumani, R. A. Laghari, H. Nawaz, Unmanned aerial vehicles: A review, Cognitive Robotics, vol. 3 (2023) 8–22. doi: 10.1016/j.cogr.2022.12.004

[2] D. Lee, D. J. Hess, M. A. Heldeweg, Safety and privacy regulations for unmanned aerial vehicles: A multiple comparative analysis, Technology in Society, vol. 71 (2022) 102079. doi: 10.1016/j.techsoc.2022.102079

[3] D. A. Santos, J. A. Bezerra, On the control allocation of fully actuated multirotor aerial vehicles, Aerospace Science and Technology, vol. 122 (2022) 107424. doi: 10.1016/j.ast.2022.107424

[4] S. Panza, D. Invernizzi, M. Giurato, M. Lovera, Design and characterization of the 2DoF Drone: a multirotor platform for education and research, IFAC-PapersOnLine, vol. 54, issue 12 (2021) 32–37. doi: 10.1016/j.ifacol.2021.11.006

[5] N. Michel, P. Wei, Z. Kong, A. K. Sinha, X. Lin, Modeling and validation of electric multirotor unmanned aerial vehicle system energy dynamics, eTransportation, vol. 12 (2022) 100173. doi: 10.1016/j.etran.2022.100173

[6] A. Taame, I. Lachkar, A. Abouloifa, Modeling of an unmanned aerial vehicle and trajectory tracking control using backstepping approach, IFAC-PapersOnLine, vol. 55, issue 12 (2022) 276–281. doi: 10.1016/j.ifacol.2022.07.324

[7] Z. Yu, Y. Zhang, B. Jiang, J. Fu, Y. Jin, A review on fault-tolerant cooperative control of multiple unmanned aerial vehicles, Chinese Journal of Aeronautics, vol. 35, issue 1 (2022) 1–18. doi: 10.1016/j.cja.2021.04.022

[8] M. S. Chehadeh, I. Boiko, Design of rules for in-flight non-parametric tuning of PID controllers for unmanned aerial vehicles, Journal of the Franklin Institute, vol. 356, issue 1 (2019) 474–491. doi: 10.1016/j.jfranklin.2018.10.015

[9] V. R. Sree Ezhil, B. S. Rangesh Sriram, R. Christopher Vijay, S. Yeshwant, R. K. Sabareesh, G. Dakkshesh, R. Raffik, Investigation on PID controller usage on Unmanned Aerial Vehicle for stability control, Materialstoday: Proceedings, vol. 66, part 3 (2022) 1313–1318. doi: 10.1016/j.matpr.2022.05.134

[10] M. M. Alam, M. Y. Arafat, S. Moh, J. Shen, Journal of Network and Computer Applications, vol. 207 (2022) 103.495. doi: 10.1016/j.jnca.2022.103495

[11] S. Vladov, Y. Shmelov, M. Petchenko, A Neuro-Fuzzy Expert System for the Control and Diagnostics of Helicopters Aircraft Engines Technical State. ICTERI 2021: ICT in Education, Research, and Industrial Applications, 28 September – 02 October 2021, Kherson, Ukraine. CEUR Workshop Proceedings, vol. 3013 (2021) 40–52.

[12] V. Slynko, O. Tunc, I. Atamas, Construction of a Lyapunov function for a linear large-scale periodic system with possibly unstable subsystems, Journal of the Franklin Institute, vol. 359, issue 14 (2022) 7510–7539. doi: 10.1016/j.jfranklin.2022.07.052

[13] B. Zhou, Y. Tian, J. Lam, On construction of Lyapunov functions for scalar linear time-varying systems, Systems & Control Letters, vol. 135 (2020) 104591. doi: 10.1016/j.sysconle.2019.104591

[14] H. Wang, Q.-L. Han, J. Liu, D. He, Discrete-time filter proportional–integral–derivative controller design for linear time-invariant systems, Automatica, vol. 116 (2020) 108918. doi: 10.1016/j.automatica.2020.108918

[15] M. M. Doshi, M. S. Bhabra, P. F. J. Lermusiaux, Computer Methods in Applied Mechanics and Engineering, vol. 405 (2023) 115865. doi: 10.1016/j.cma.2022.115865
[16] O. Dronova, D. Parinov, B. Soloviev, D. Kasumova, E. Kochetkov, O. Medvedeva, I. Sergeeva, Transportation Research Procedia, vol. 63 (2022) 2308–2314. doi: 10.1016/j.trpro.2022.06.263
[17] N. Sun, J. Zhao, G. Wang, C. Liu, P. Liu, X. Tang, J. Han, Transformer-based moving target tracking method for Unmanned Aerial Vehicle, Engineering Applications of Artificial Intelligence, vol. 116 (2022) 105483. doi: 10.1016/j.engappai.2022.105483
[18] H. Nemati, A. Montazeri, Analysis and Design of a Multi-Channel Time-Varying Sliding Mode Controller and its Application in Unmanned Aerial Vehicles, IFAC-PapersOnLine, vol. 51, issue 22 (2018) 244–249. doi: 10.1016/j.ifacol.2018.11.549
[19] A. Tahir, J. Boling, M.-H. Haghbayan, H. T. Toivonen, J. Plosila, Swarms of Unmanned Aerial Vehicles – A Survey, Journal of Industrial Information Integration, vol. 16 (2019) 100106. doi: 10.1016/j.jii.2019.100106
[20] A. M. Parrany, A. Alasty, Decentralized aggregation and leader-following control of a swarm of quadcopters with nonlinear under-actuated dynamics, Aerospace Science and Technology, vol. 107 (2020) 106317. doi: 10.1016/j.ast.2020.106317
[21] M. Pouzesh, S. Mobayen, Event-triggered fractional-order sliding mode control technique for stabilization of disturbed quadrotor unmanned aerial vehicles, vol. 121 (2022) 107337. doi: 10.1016/j.ast.2022.107337
[22] B. Andrievsky, S. Tomashevich, A. Fradkov, K. Amelin, Quadrocopters Formation Control Over the Limited-band Communication Network, IFAC-PapersOnLine, vol. 48, issue 9 (2015) 85–90. doi: 10.1016/j.ifacol.2015.08.064
[23] M. M. Alam, M. Y. Arafat, S. Moh, J. Shen, Topology control algorithms in multi-unmanned aerial vehicle networks: An extensive survey, Journal of Network and Computer Applications, vol. 207 (2022) 103495. doi: 10.1016/j.jnca.2022.103495
[24] Z. Yu, Y. Zhang, B. Jiang, C.-Y. Su, J. Fu, Y. Jin, T. Chai, Fractional order PID-based adaptive fault-tolerant cooperative control of networked unmanned aerial vehicles against actuator faults and wind effects with hardware-in-the-loop experimental validation, Control Engineering Practice, vol. 114 (2021) 104861. doi: 10.1016/j.conengprac.2021.104861
[25] D. A. M. Ravell, M. M. Maia, F. J. Diez, Modeling and control of unmanned aerial/underwater vehicles using hybrid control, Control Engineering Practice, vol. 76 (2018) 112–122. doi: 10.1016/j.conengprac.2018.04.006
[26] Yu. Shmelov, A. Daved, N. Abramov, Hybrid neural network control of an unmanned aerial vehicle, Proceedings I International scientific and practical conference "Aviation, Industry, Society", dedicated to the 60th anniversary of KLK KhNUVS, 14 May 2020, Kremenchuk, Ukraine, 52–54.
[27] S. Krishnan, G. A. Rajagopalan, S. Kandhasamy, M. Shanmugavel, Continuous-Time Trajectory Optimization for Decentralized Multi-Robot Navigation, IFAC-PapersOnLine, vol. 53, issue 1 (2020) 494–499. doi: 10.1016/j.ifacol.2020.06.083
[28] F. d'Apolito, C. Sulzbachner, System Architecture of a Demonstrator for Indoor Aerial Navigation, IFAC-PapersOnLine, vol. 52, issue 25 (2019) 316–320. doi: 10.1016/j.ifacol.2019.12.542
[29] A. Nikolakopoulou, M. S. Hong, R. D. Braatz, Dynamic state feedback controller and observer design for dynamic artificial neural network models, Automatica, vol. 146 (2022) 110622. doi: 10.1016/j.automatica.2022.110622
[30] S. Vladov, Y. Shmelov, R. Yakovliev, Methodology for Control of Helicopters Aircraft Engines Technical State in Flight Modes Using Neural Networks, The Fifth International Workshop on Computer Modeling and Intelligent Systems (CMIS-2022), May, 12, 2022, Zaporizhzhia, Ukraine, CEUR Workshop Proceedings (ISSN 1613-0073) vol. 3137 (2022) 108–125. doi: 10.32782/cmis/3137-10
[31] S. Vladov, Y. Shmelov, R. Yakovliev, Modified Helicopters Turboshaft Engines Neural Network On-board Automatic Control System Using the Adaptive Control Method. ITTAP'2022: 2nd International Workshop on Information Technologies: Theoretical and Applied Problems, November 22–24, 2022, Ternopil, Ukraine. CEUR Workshop Proceedings, vol. 3309 (2022) 205–229.
[32] Y. Shmelov, S. Vladov, Y. Klimova, M. Kirukhina, Expert system for identification of the technical state of the aircraft engine TV3-117 in flight modes, in: Proceedings of the System

Analysis & Intelligent Computing: IEEE First International Conference on System Analysis & Intelligent Computing (SAIC), 08–12 October 2018. 77–82. doi: 10.1109/SAIC.2018.8516864
[33] B. Perez-Sanchez, O. Fontenla-Romero, B. Guijarro-Berdinas, A review of adaptive online learning for artificial neural networks, Artificial Intelligence Review, vol. 49 (2018) 281–299. doi: 10.1007/s10462-016-9526-2
[34] F. M. Salem, Recurrent Neural Networks: From Simple to Gated Architectures, Switzerland, Springer Nature Switzerland AG, 2022, 200.
[35] R. Vang-Mata, Multilayer Perceptrons: Theory and Applications, New York, Nova Science Publishers (2020) 143.
[36] D. Plonis, A. Katkevicius, V. Urbanavicius, D. Miniotas, A. Serackis, A. Gurska, Delay systems synthesis using multi-layer perceptron network, Acta Physica Polonica A, vol. 133, no 5 (2018) 1281–1286. doi: 10.12693/APhysPolA.133.1281
[37] C. Qin, J. Wang, H. Zhu, J. Zhang, S. Hu, D. Zhang, Neural network-based safe optimal robust control for affine nonlinear systems with unmatched disturbances, Neurocomputing, vol. 506 (2022) 228–239. doi: doi.org/10.1016/j.neucom.2022.07.072
[38] W. Mlynarski, M. Hledik, T. R. Sokolowski, G. Tkacik, Statistical analysis and optimality of neural systems, Neuron, vol. 109, issue 7 (2021) 1227–1241.e5 doi: doi.org/10.1016/j.neuron.2021.01.020
[39] J. Sarangapani, Neural Network Control of Nonlinear Discrete-Time Systems, Boca Raton, CRC Press, 622.
[40] S. Akhramovich, A. Barinov, V. Malyshev, A. Starkov, Backstepping synthesis of the height control system of an unmanned aerial vehicle, Aerospace and Mechanical Engineering, vol. 17, issue 2 (2018) 7–22. doi: 10.18287/2541-7533-2018-17-2-7-22
[41] Y. Fu, J. Li, X. Li, S. Wu, Dynamic event-triggered adaptive control for uncertain stochastic nonlinear systems, Applied Mathematics and Computation, vol. 444 (2023) 127800. doi: 10.1016/j.amc.2022.127800
[42] K. Xie, X. Yu, W. Lan, Optimal output regulation for unknown continuous-time linear systems by internal model and adaptive dynamic programming, Automatica, vol. 146 (2022) 110564. doi: 10.1016/j.automatica.2022.110564
[43] Q. Wu, B. Zhao, D. Liu, M. M. Polycarpou, Event-triggered adaptive dynamic programming for decentralized tracking control of input constrained unknown nonlinear interconnected systems, Neural Networks, vol. 157 (2023) 336–349. doi: 10.1016/j.neunet.2022.10.025
[44] S. Vladov, Y. Shmelov, R. Yakovliev, Modified Searchless Method for Identification of Helicopters Turboshaft Engines at Flight Modes Using Neural Networks, in: Proceedings of the 2022 IEEE 3rd KhPI Week on Advanced Technology, Kharkiv, Ukraine, October 03–07, 2022, 257–262.
[45] S. Andropov, A. Guirik, M. Budko, M. Budko, Unmanned air vehicle stabilization based on neural network regulator, Scientific and technical bulletin of information technologies, mechanics and optics, vol. 16, issue 5 (2016) 796–800. doi: 10.17586/2226-1494-2016-16-5-796-800
[46] Q. Wei, Z. Yang, H. Su, L. Wang, Monte Carlo-based reinforcement learning control for unmanned aerial vehicle systems, Neurocomputing, vol. 507 (2022) 282–291. doi: 10.1016/j.neucom.2022.08.011