https://doi.org/10.15407/knit2024.02.003
UDC 629.78

S. V. KHOROSHYLOV[1], Leading Researcher, Dr. Sci. in Tech., Professor
ORCID.org/0000-0001-7648-4791
E-mail: skh@ukr.net
C. WANG[2], Professor, PhD
ORCID.org/0000-0002-1358-7731
E-mail: wangcq@nwpu.edu.cn

[1]Institute of Technical Mechanics of the National Academy of Science of Ukraine and the State Space Agency of Ukraine
15 Leshko-Popel Str., Dnipro, 49005 Ukraine
[2]Northwestern Polytechnical University
127 West Youyi Road, Xi'an Shaanxi, 710072, P.R. China

SPACECRAFT RELATIVE ON-OFF CONTROL VIA REINFORCEMENT LEARNING

The article investigates the task of spacecraft relative control using reactive actuators, the output of which has two states, "on" or "off". For cases where the resolution of the thrusters does not provide an accurate approximation of linear control laws using a pulse-width thrust modulator, the possibility of applying reinforcement learning methods for direct finding of control laws that map the state vector and the on-off thruster commands has been investigated. To implement such an approach, a model of controlled relative motion of two satellites in the form of a Markov decision process was obtained. The intelligent agent is presented in the form of "actor" and "critic" neural networks, and the architecture of these modules is defined. It is proposed to use a cost function with variable weights of control actions, which allows for optimizing the number of thruster firings explicitly. To improve the control performance, it is proposed to use an extended input vector for the "actor" and "critic" neural networks of the intelligent agent, which, in addition to the state vector, also includes information about the control action on the previous control step and the control step number. To reduce the training time, the agent was pre-trained on the data obtained using conventional control algorithms. Numerical results demonstrate that the reinforcement learning methodology allows the agent to outperform the results provided by the linear controller with the pulse-width modulator in terms of control accuracy, response time, and number of thruster firings.

Keywords: *on-off control, reinforcement learning, spacecraft relative control, actor, critic, neural network, thruster firing.*

1. INTRODUCTION

Recently, on-orbit servicing missions [19] have attracted significant attention in the space community. For example, such missions can be used to replace or repair faulty spacecraft components, refuel in orbit, and remove space debris [1, 9]. To implement such operations, the service spacecraft (SSC) needs to perform maneuvers in close proximity to a servicing object (SO), solving the tasks of relative guidance and control [13]. Thrusters (TH) are usually used to control the SSC relative motion. Unlike other actuators, such as reaction wheels, the output of a TH has two values: on or off. This mode of operation is explained by the fact that precise adjustment of thrust is difficult to implement, mainly because of dirt particles, which prevent the small valve from being completely closed. This, in turn, leads to leakage of the propellant

Цитування: Khoroshylov S. V., Wang C. Spacecraft relative on-off control via reinforcement learning. *Space Science and Technology*. 2024. **30**, № 2 (147). P. 3—14. https://doi.org/10.15407/knit2024.02.003

and the engagements of the THs, pointed in opposite directions. A TH operating in this mode is a significantly nonlinear actuator, which complicates the direct synthesis of control laws [3, 16].

Some of the first control algorithms using on-off actuators [25] were based on the Lyapunov stability theory, where the TH firing is selected by minimizing the derivative of the Lyapunov function. However, such control algorithms do not minimize a practically meaningful performance criterion, such as propellant consumption and control error.

To overcome the issue, it is often necessary to synthesize a linear control law that minimizes a selected performance criterion. After that, modulators are used to approximate the linear control by generating a sequence of thrust pulses with the required width, as mentioned in references [2, 17]. For this task, pulse-width (PWM) and pulse-width pulse-frequency (PWFM) modulators are used [28]. The control system design is easier with PWM than with PWFM since the first one only introduces additional damping, and the second one changes the bandwidth and phase characteristics of the system closed-loop.

Control performance within the PWM approach largely depends on the approximation accuracy of the linear control by the sequence of pulses after the modulator. Ref. [11] investigates the optimal time delay of the pulse, expressed as the error between the output states without and with PWM. The results of this work suggest to center the pulse within the sample period. In addition to pulse centering, the authors of the article [5] suggest dividing the pulse into several smaller pulses, which are uniformly distributed over the sampling period. However, this gives only a marginal improvement but requires THs with a much longer operational lifetime. Such insignificant improvements do not justify the qualification of the THs for a significantly greater number of work cycles.

To provide precise control, it is recommended that the PWM must have a resolution that is 50—100 times greater than the sampling period. If the modulator has an insufficient resolution, then control performance degrades. In addition to the issue, the above approach does not allow designers to explicitly optimize the number of TH firings.

The impressive results obtained using deep learning (DL) techniques [4] recently boosted interest in artificial intelligence methods [6] among researchers and practitioners in the world. DL is rapidly developing and demonstrating promising capabilities in solving complex tasks and finding non-trivial solutions to existing problems [27].

Machine learning is a subset of artificial intelligence methods that are used to develop algorithms capable of solving a problem based on the search for regularities in various input data [20]. Machine learning methods based on artificial neural networks (NNs) are called deep learning. Recent advances in DL are largely achieved due to the development of new NN architectures.

Not so long ago, these methods were begun to be used to solve space-related tasks [12, 15]. In Ref. [22], the policy for performing docking maneuvers with six degrees of freedom was developed based on reinforcement learning (RL) and implemented in the form of the feedback control law. The simulation results of the approach and docking maneuvers for the Apollo mission demonstrate that the capabilities of the resulting policy can be compared with the algorithms obtained by conventional optimal control methods.

The article [14] presents an approximation of the optimal relative control for the underactuated spacecraft using the RL and the study of the influence of various factors on the performance of such a solution. This approach allows finding close to optimal control algorithms as a result of the interaction of the control system with the plant using the reinforcement signal to estimate the performance of the control actions.

A new approach called deep guidance is investigated in Ref. [10]. The authors use deep RL to learn guidance policies instead of handcrafting them. The results show that such a system can be fully simulated and transferred into real-world conditions with an acceptable loss of performance without any additional tuning. Ref. [7] proposes a new adaptive guidance system developed using meta-RL. The recurrent NN allows the obtained algorithms to adapt in real time to environmental disturbances acting on the SC. In Ref. [8], an adaptive integrated guidance, navigation, and control system was developed for maneuvering in the proximity of asteroids with unknown environmental dynamics, with initial conditions covering large launch areas, and without knowing the model

of the asteroid shape. The system is implemented as a policy optimized using meta-RL.

Unfortunately, at present, there are no results demonstrating the successful application of RL methods for on-off SSC relative control. At the same time, this approach may provide the following benefits:

1. A better control performance compared to the conventional PWM-based approach.

2. To optimize the frequency of the TH firings.

Such an RL-based approach is investigated in this article, for the implementation of which the following tasks are solved:

1. To build a model of the plant dynamics in a form that allows an RL-based algorithm to be applied.

2. To select the structure and parameters of the intelligent agent (AI).

3. To train the intelligent agent.

4. To analyze the performance of the RL-based controller for SSC relative control.

2. MODEL OF SPACECRAFT RELATIVE DYNAMICS

An orbital reference frame (ORF) *Oxyz* is used for the mathematical description of the SSC motion relative to the SO. The origin of the ORF coincides with the center of mass of the SSC. The axis *Ox* coincides with the direction of the position vector, which determines the SSC center of mass relative to the Earth center of mass, the *Oz* axis coincides with the normal to the plane passing through the axis *Ox* and the vector of the SSC orbital velocity, and points towards the positive values of the orbital angular momentum. The axis *Oy* complements the reference frame to the right one.

The position of the SO relative to the ORF is determined by the position vector *L*. The relative dynamics of the "SSC — SO" system can be described using the following linearized system of equations [29]:

$$\ddot{x} - \omega^2 x - 2\omega\dot{y} - \dot{\omega}y - kx = \frac{f_x^d}{m^d} - \frac{f_x^s}{m^s}, \qquad (1)$$

$$\ddot{y} - \omega^2 y + 2\omega\dot{x} + \dot{\omega}x + ky = \frac{f_y^d}{m^d} - \frac{f_y^s}{m^s}, \qquad (2)$$

$$\ddot{z} + kz = \frac{f_z^d}{m^d} - \frac{f_z^s}{m^s}, \qquad (3)$$

where x, y, are the projections of the vector L on the ORF axes; m^s, m^d are the mass of SSC and SO, respectively; f_x^d, f_y^d, f_z^d are the ORF projections of the total force vector F^d, acting on the SO; f_x^s, f_y^s, f_z^s are the ORF projections of the total vector F^s, acting on the SSC.

The total force vector F^s includes control thrust and external disturbances acting on the SSC. The forces F^d and F^s may also include J2-disturbances, the gravity of the Sun and the Moon, atmospheric drag, and solar radiation pressure.

The parameters ω, $\dot{\omega}$ and k in Eq. (1)—(3) are determined as follows:

$$\omega = \sqrt{\frac{\mu}{p^3}}(1+\varepsilon\cos\nu),$$

$$\dot{\omega} = -2\varepsilon\sqrt{\frac{\mu}{p^3}}\sin\nu(1+\varepsilon\cos\nu)\omega,$$

$$p = a(1-\varepsilon^2),$$

$$k = \frac{\mu}{r^3},$$

$$r = \frac{a(1-\varepsilon^2)}{1+\varepsilon\cos\nu},$$

where μ is the Earth's gravitational constant, ε is the orbit eccentricity, ν is the true anomaly, a is the semi-major axis, r is the orbital radius.

Equations (1) and (2) describe the dynamics of the system in the orbital plane, and (3) describes its motion out of the orbital plane.

Neglecting the influence of external disturbances and considering the state vectors

$$X_{in} = \left[x, y, \dot{x}, \dot{y}\right]^{\mathrm{T}}, \; X_{out} = [z, \dot{z}]^{\mathrm{T}},$$

and control

$$U_{in} = \left[u_x, u_y\right]^{\mathrm{T}}, \; U_{out} = u_z,$$

model (1) can be represented in the state space form as

$$\begin{aligned} \dot{X}_{in} &= A_{in}X_{in} + B_{in}U_{in}, \\ \dot{X}_{out} &= A_{out}X_{out} + B_{out}U_{out}, \end{aligned} \qquad (4)$$

where

$$A_{in} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \omega^2 + 2k & \dot{\omega} & 0 & 2\omega \\ -\dot{\omega} & \omega^2 - k & -2\omega & 0 \end{bmatrix},$$

$$B_{in} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -\frac{1}{m^s} & 0 \\ 0 & -\frac{1}{m^s} \end{bmatrix},$$

$$A_{out} = \begin{bmatrix} 0 & 1 \\ -k & 0 \end{bmatrix}, \quad B_{out} = \begin{bmatrix} 0 \\ -\frac{1}{m^s} \end{bmatrix}.$$

The magnitudes of various components of the state vector are significantly different. This can complicate the training of the NNs. To eliminate this issue, the state vector is normalized as follows:

$$X_{in}^n = \left[x / x_m, y / y_m, \dot{x} / \dot{x}_m, \dot{y} / \dot{y}_m \right]^{\mathrm{T}}, \quad X_{out}^n = \left[z / z_m, \dot{z} / \dot{z}_m \right]^{\mathrm{T}}, \tag{5}$$

where $x_m, y_m, z_m, \dot{x}_m, \dot{y}_m, \dot{z}_m$, are the maximum values of the corresponding states. For the normalized state vector, the dynamic model has the following form:

$$\dot{X}^n = A^n X^n + B^n U, \tag{6}$$

where

$$A^n = N^{-1} A N,$$

$$B^n = N^{-1} B,$$

$$N = \mathrm{diag}\left(x_m, y_m, z_m, \dot{x}_m, \dot{y}_m, \dot{z}_m \right).$$

Since the modern controller of the spacecraft is implemented as a discrete computer system, the following discrete form of the model (6) is used:

$$X_{k+1} = A_k X_k + B_k U_k, \tag{7}$$

where $A_k = (I + A^n T), B_k = B^n T, T$ is the sampling time, k is the sample number.

We also assume that the full state vector is measurable and that these measurements are not corrupted by noise.

3. DISCRETE LINEAR QUADRATIC REGULATOR WITH PWM

For comparison reasons, we consider one of the conventional approaches for spacecraft relative control, namely the combination of a linear-quadratic regulator (LQR) with PWM. Methods of synthesis of the optimal linear-quadratic controller with discrete time (DLQR) [26] are a widely used methodology for designing control systems (SC). The goal of the DLQR synthesis is to find a constant gain matrix K for the full feedback law that minimizes the quadratic cost function:

$$J = \min \sum_{k=0}^{\infty} \left(Q^{\mathrm{T}} X_k Q + R^{\mathrm{T}} U_k R \right), \tag{8}$$

where Q, R are the weight matrices that penalize system states X_k and control U_k, respectively.

The impressive robust stability properties of DLQR allow developers to use it for systems whose real parameters differ significantly from the nominal ones. DLQR implements the control law with full feedback for SSC in the following form:

$$u_k = K(X^r - X_k),$$

where X^r is the reference value of the state vector, which determines the necessary relative position between the SSC and SO.

The matrix of the optimal feedback gain is determined as follows

$$K = (R + B^{\mathrm{T}} P B)^{-1} B^{\mathrm{T}} P A,$$

where A, B are the matrices of the state space representation of the dynamic model, P is a unique semidefinite solution of the discrete-time Riccati equation

$$P = Q + A^T P A - A^T P B (R + B^T P B)^{(-1)} B^T P A.$$

When the output of the actuators has only two states, on and off, the DLQR is used in conjunction with PWM, which approximates the output of DLQR by a sequence of pulses of variable width. The pulse width on each sample period is determined as follows:

$$t_f = \frac{u_k}{u_f} T, \; t_f \le T,$$

where u_f is the nominal thrust of a TH.

4. REINFORCEMENT LEARNING BASED CONTROL

The RL-based control setup assumes that the control system learns by analyzing the results of its actions [27]. These results are evaluated by a scalar signal (reinforcement), which is received from the plant with which the control system interacts. The reinforcement signal can be interpreted as a criterion allowing the intelligent control system to change its control algorithms, taking into account the achievement of the long-term goal.

A general RL algorithm includes the following steps:

1) at a time t_k, the plant is in a state X_k;

2) in this state, the control system selects one of the possible control actions U_k;

3) the control system applies this action, which leads to the transition of the plant to a new state X_{k+1}, and the control system receives the reinforcement signal C_k;

4) the algorithm continues being applied from step 2, taking into account the received reinforcement, or the algorithm stops if the new state is final.

We denote χ as a set of states and A as a set of control actions. Then, reinforcement C_k is a consequence of the action U_k selected in the state X_k. The reinforcement signal is a function that depends on a vector defined in the space $\chi \times A$.

The control system selects actions in such a way as to minimize the total cost, which is determined as follows:

$$G_k = C_k + \gamma C_{k+1} + \gamma^2 C_{k+2} + ... = \sum_{i=0}^{\infty} \gamma^i C_{k+i},$$

$$0 \le \gamma \le 1.$$

The discount factor γ determines the importance of the predicted cost values during the selection of the control actions.

One of the key elements of the RL is the value function. Suppose that in each state X_k, the SC apply a control action according to a certain algorithm, which is called a policy π:

$$U_k = \pi\left(X_k\right),$$

then the value function determines the total cost that is paid by moving from the initial state X_k selecting control actions according to the policy π. This function can be represented as:

$$V^{\pi}\left(X_k\right) = \sum_{i=0}^{\infty} \gamma^k C_{k+i}\left(X_{k+i}, U_{k+i}\right) =$$

$$= C_k\left(X_k, U_k\right) + \gamma V^{\pi}\left(X_{k+1}\right).$$

Reinforcement learning can be implemented using actor-critic architecture. In this case, the critic provides predictions of the value function for each state, and the actor maps the state vector to the control actions.

According to the methodology of deep RL, the actor and critic are implemented in the form of feedforward multilayer neural networks, which approximate the control law and cost function, respectively: $V^{\pi}\left(X_k, \phi\right), \pi\left(X_k, \phi\right)$, where θ, ϕ are the vectors of critic and actor parameters, respectively.

There are many different RL algorithms. In this study, the Proximal policy optimization (PPO) algorithm is used [26]. This algorithm is implemented as follows:

1. To find the total cost of G_t, which is the sum of the cost for this time step and the discounted future cost [21]

$$G_t = \sum_{k=t}^{ts+m} \left(\gamma^{k-t} C_k\right) + b\gamma^{N-t+1} V\left(X_{ts+N}, \theta\right),$$

where b is 0 if X_{ts+N} is the final state and 1 otherwise. That is, if X_{ts+N} is not the final state, the discounted future value includes a function of the discounted state value calculated using the critic neural net V.

2. To find the advantage function D_t

$$D_t = G_t - V\left(X_t, \theta\right).$$

3. To update the critic parameters by minimizing the loss function L_{critic} for all received mini-batch data

$$L_{critic}\left(\theta\right) = \frac{1}{M} \sum_{i=1}^{M} \left(G_i - V\left(X_i, \theta\right)\right)^2.$$

4. Update the actor parameters by minimizing the actor loss function L_{actor} of all received mini-batch data as follows

$$L_{actor}\left(\phi\right) =$$

$$= \frac{1}{M} \sum_{i=1}^{M} \left(-\min\left(r_i\left(\phi\right) \cdot D_i, c_i\left(\phi\right) \cdot D_i\right) + w\mathcal{H}_i\left(\theta, X_i\right)\right),$$

$$r_i\left(\phi\right) = \frac{\pi\left(U_i \mid X_i, \phi\right)}{\pi\left(U_i \mid X_i, \phi_{old}\right)},$$

$$c_i\left(\phi\right) = \max\left(\min\left(r_i\left(\phi\right), 1+\varepsilon\right), 1-\varepsilon\right),$$

where D_i and G_i are the advantage and total cost function for the i-th element of the mini-batch, respectively; $\pi\left(U_i \mid X_i, \phi\right)$ is the probability of performing the action U_i in the state X_i, given the updated policy parameters ϕ; $\pi\left(U_i \mid X_i, \phi_{old}\right)$ is the probability of performing action U_i in state X_i, given the previous policy parameters ϕ_{old} before the current learning epoch; ε is the clip factor; $\mathcal{H}_i\left(\theta, X_i\right)$ is the loss entropy; w is the loss entropy weight.

The agent uses the following entropy value

$$\mathcal{H}_i\left(\theta, X_i\right) = -\sum_{k=1}^{PN} \pi\left(U_k \mid X_i, \phi\right) \ln \pi\left(U_k \mid X_i, \phi\right),$$

where PN is the number of possible discrete actions; $\left(U_i \mid X_i, \phi\right)$ is the probability of action U_i in state X_i according to the current policy.

We propose to use the following cost function:

$$C_k = Q^{\mathrm{T}} X_k Q + L_k{}^{\mathrm{T}} R^{\mathrm{T}} U_k L_k R . \qquad (9)$$

This function is similar to criterion (8), but the additional variable weight L_k allows us to optimize the control law more flexibly, for example, to encourage the agent to use wider pulses.

We studied four intellectual agents (IA), which use different input information as follows:

1) IA-1 receives an ordinary state vector X_k as an input, the dimensions of the input vectors for the in-plane and out-of-plane cases are $n_{in} = 4$ and $n_{out} = 2,$ respectively;

2) In addition to the state vector X_k, IA-2 also receives information about the control action on the previous control step as follows

$$\left[X_k, u_{k-1}\right]^{\mathrm{T}},\ n_{in} = 5, \text{ and } n_{out} = 3;$$

3) In addition to the state vector X_k, IA-3 also receives information about the normalized number i of the TH pulses within the LQR sample period as follows

$$\left[X_k, \frac{i}{i_m}\right]^{\mathrm{T}},\ n_{in} = 5, \text{ and } n_{out} = 3;$$

4) IA-4 receives the following input information

$$\left[X_k, \frac{i}{i_m}, u_{k-1}\right]^{\mathrm{T}},\ n_{in} = 6, \text{ and } n_{out} = 4.$$

Table 1. **Structure of neural networks**

Layer	Number of neurons			
	actor		critic	
	in-plain	out-of-plain	in-plain	out-of-plain
Input	n_{in}	n_{out}	n_{in}	n_{out}
1-st hidden	$10n_{in}$	$10n_{out}$	$10n_{in}$	$10n_{out}$
2-st hidden	$\sqrt{900n_{in}}$	$\sqrt{300n_{out}}$	$\sqrt{100n_{in}}$	$\sqrt{100n_{out}}$
3-st hidden	90	30	10	10
Output	9	3	1	1

The agent can apply three control actions $\left[-u_f, 0, u_f\right]$ in each control channel, so the total number of possible different states of the actuators is $3^2 = 9$ for the in-plane case and $3^1 = 3$ for the out-of-plane case. These values specify the number of outputs of the categorical actors, which determine the relationship between the input vector and the corresponding state of the actuators.

For z-channel, the outputs of the actor directly specify the probability of the following actions $-u_f, 0, u_f$. For channels x and y, at first, the decimal integer number corresponding to the state of the actuators at the actor's output is converted to its ternary representation, and then, the control vector is determined as follows

$$U_{in} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} = u_f \left(\begin{bmatrix} u_{xy}^1 \\ u_{xy}^2 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right),$$

where u_{xy}^1, u_{xy}^2, are the first and second digits of the ternary representation of the actor's output, respectively.

Actors and critics of these agents are implemented in the form of NNs, the structure of which is shown in Table 1. Almost all NN layers use the Relu activation function. The only actor's output uses the SoftMax activation function.

5. NUMERICAL RESULTS

The following system data were used for the training and studying the intelligent agent: a = 7017 km, m^s = 500 kg, m^d = 1575 kg, T = 200 sec, T_f = 10 sec, U_f = 1.6 N, $Q = 0.001 \times \mathrm{diag}\left(0.01, 0.01, 0.01, 1, 1, 1\right)$, $\mathrm{R} = 50 \times \mathrm{diag}\left(1, 1, 1\right)$.

The state vector has the following maximum component values: x_m = 800 m, y_m = 800 m, $\dot{x}_m$ = = 2 m/s, $\dot{y}_m$ = 2 m/s.

To speed up the learning process, all actors were pre-trained using supervised learning at the first stage on data obtained using DLQR with PWM.

The AI-1 and AI-2 are characterized by a significant steady error (Fig. 1, *a*). The AI-1 uses a large number of short pulses, and AI-2 uses a smaller number of long pulses (Fig. 1, *c*). The control accuracy of AI-3 and AI-4 is similar to that of the DLQR with PWM (Fig. 1, *b*, *d*), while it is assumed that the information about the control action on the previous control step as part of the input vector will make it

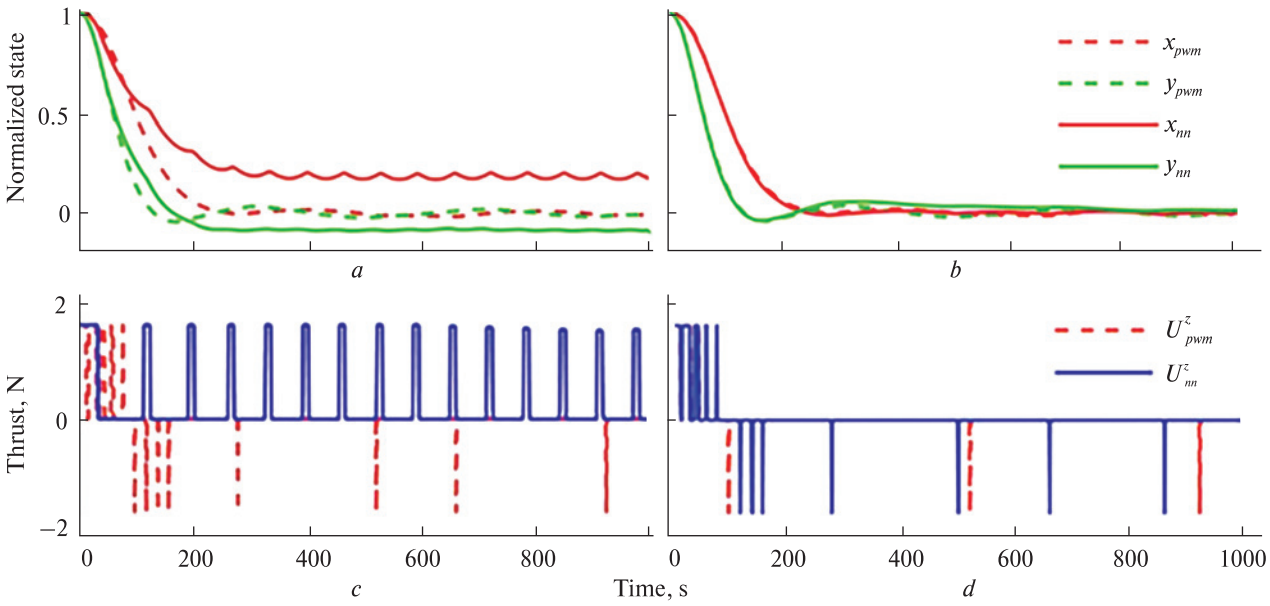

Figure 1. Normalized in-plane relative position for the supervise-trained agents (*a* — IA-2, *b* — IA-4) and TH thrust in *x*-direction for the supervise-trained agents (*c* — IA-2, *d* — IA-4)

Table2. **Performance metrics for AI-2 in case of RL with constant action weights**

No	$\tilde{L}$	Number of TH firings	Mon	Total momentum, s	Error, 10^{-3}		
					x	*y*	mean
0	PWM	27	61.6	1664	19	19	19
1	0.0009	10	377.6	3776	120	60	90
2	0.0006	17	256.0	4352	62	39	50.5
3	0.0003	12	304.0	3648	88	35	61.5
4	0.00001	76	89.6	6816	18	14	16

possible to optimize the frequency of the TH firings using RL.

At the second stage the pre-trained agents were trained using RL with the following hyperparameters:

experience horizon —1500,
clip factor $\varepsilon = 0.015$,
loss entropy weight $w = 0.005$,
mini batch size —1024,
discount factor $\gamma = 0.9994$.

The learning rates of the actor and critic were 1e-4 and 5e-5, respectively.

We used both constant weighting coefficients of actions $L_k^2 = \tilde{L}$ and variables ones formed as follows: $L_k{}^2 = \tilde{L}_1$ if $u_k \neq u_{k-1}$ and $L_k{}^2 = \tilde{L}_2$ if $u_k = u_{k-1}$.

Fig. 2 show the dependence of the normalized in-plane state vector and the TH thrust for AI-2 after being trained using RL with constant action weights. Performance metrics for these cases are presented in Table 2. In these cases, AI-2 exhibits similar behavior to the supervise-trained agent, namely a tendency to use too long control pulses. This, in most cases, does not allow AI-2 to outperform a PWM controller in terms of control accuracy. To estimate the agents' efficiency in terms of the TH firing, we use the ratio of the total momentum to the number of TH firings for the episode. This metric is denoted as Mon.

Fig. 3 show the variations of the normalized in-plane state vector and the TH thrust for AI-4 after being RL-trained with constant action weights. Per-

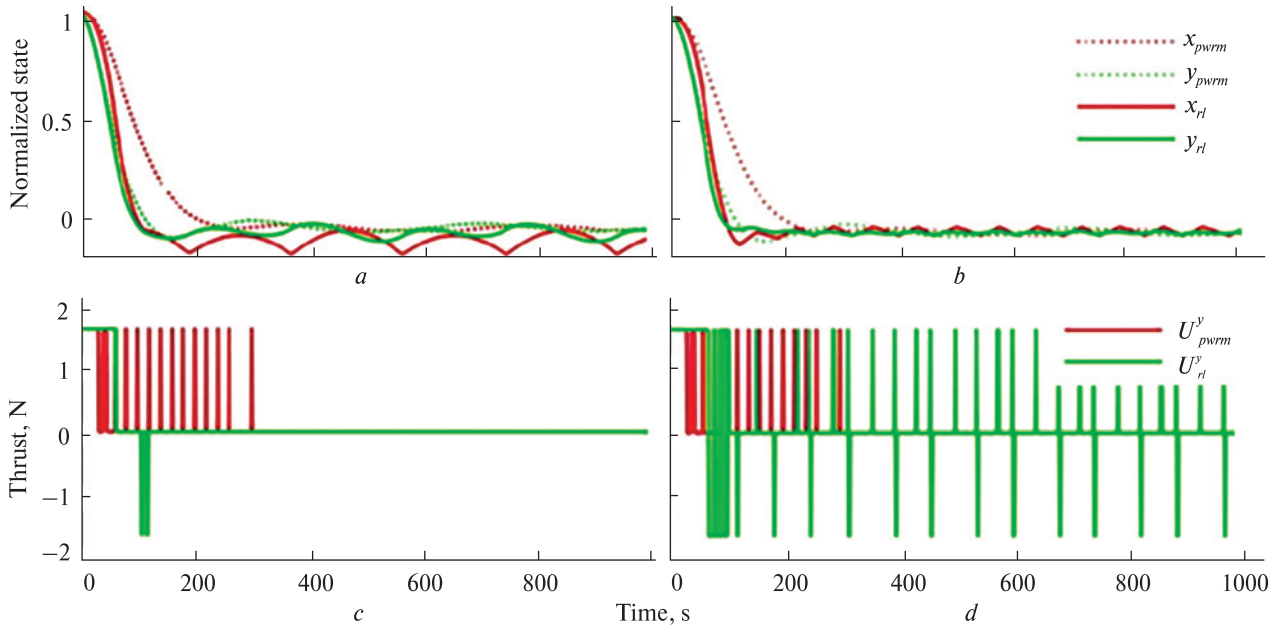

Figure 2. Normalized in-plane relative position for the RL-trained IA-2 (*a* — with $\tilde{L} = 0.0009$, *b* — with $\tilde{L} = 0.00001$) and TH thrust in *y*-direction for the RL-trained IA-2 (*c* — with $\tilde{L} = 0.0009$, *d* — with $\tilde{L} = 0.00001$)

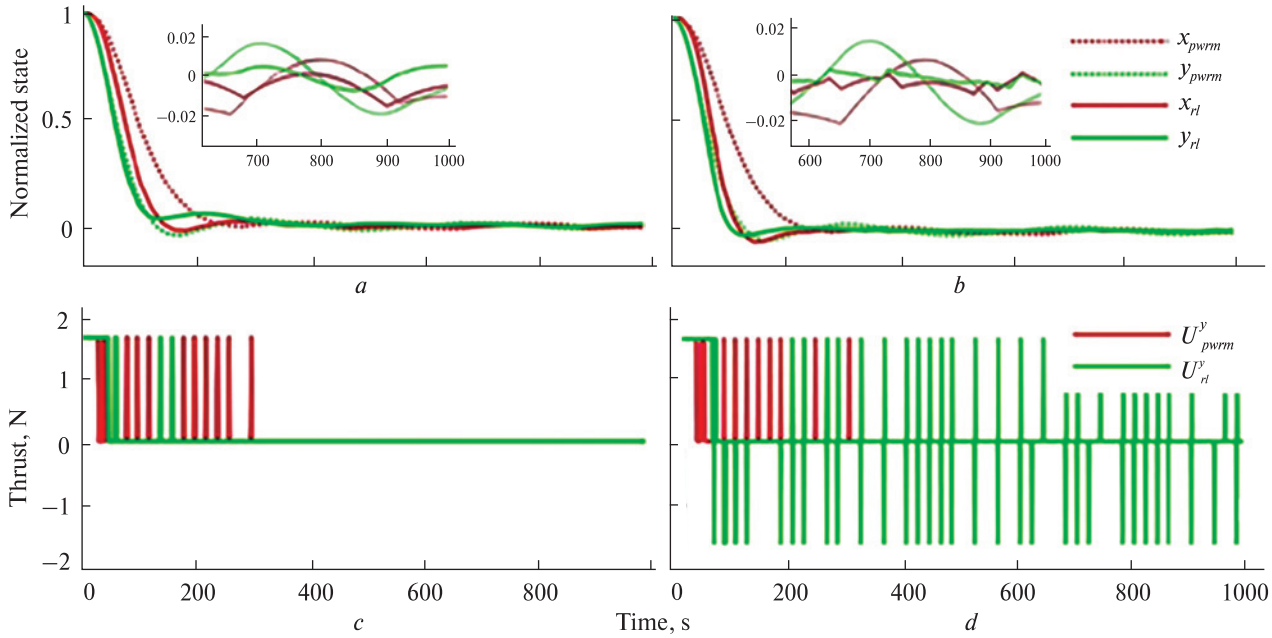

Figure 3. Normalized in-plane relative position for the RL-trained IA-4 (*a* — with $\tilde{L} = 0.0006$, *b* — with $\tilde{L} = 0.0001$) and TH thrust in *y*-direction for the RL-trained IA-4 (*c* — with $\tilde{L} = 0.0006$, *d* — with $\tilde{L} = 0.0001$)

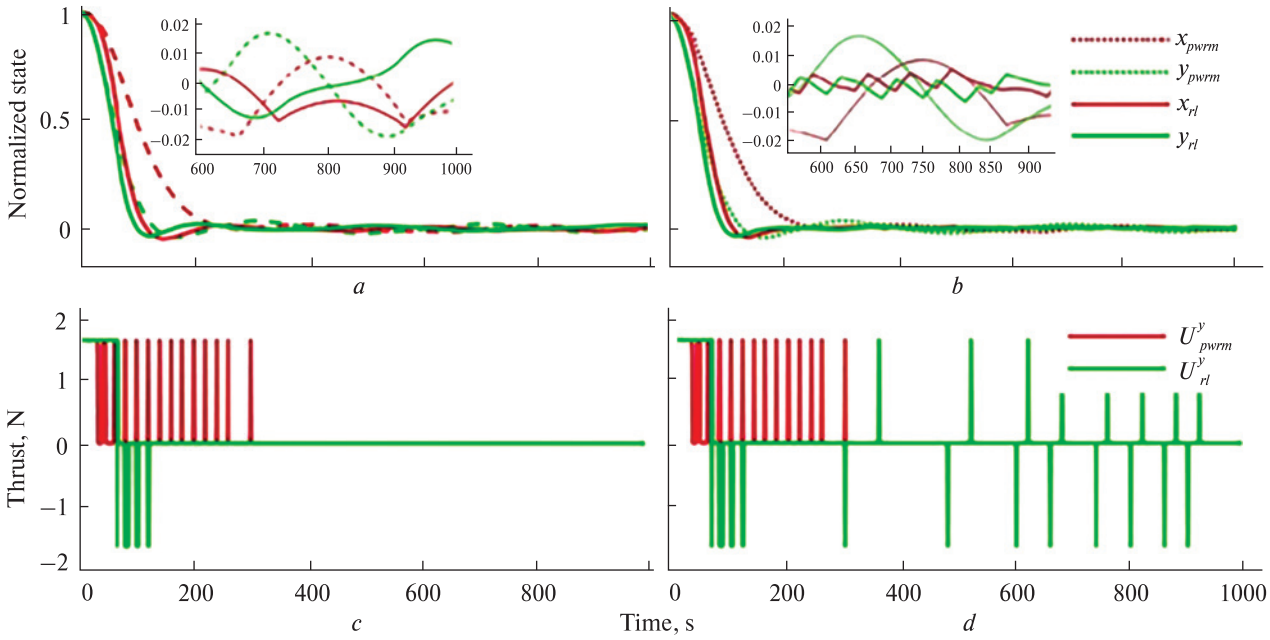

Figure 4. Normalized in-plane relative position for the RL-trained IA-4 (*a* — $\tilde{L}_1 = 0.0009$, $\tilde{L}_2 = 0$; *b* —$\tilde{L}_1 = 0.0001$, $\tilde{L}_2 = 0$) and TH thrust in *y*-direction for the RL-trained AI-4 (*c* — $\tilde{L}_1 = 0.0009$, $\tilde{L}_2 = 0$; *d* — $\tilde{L}_1 = 0.0001$, $\tilde{L}_2 = 0$)

Table 3. **Performance metrics for AI-4 in case of RL with constant action weights**

No	$\tilde{L}$	Number of TH firings	Mon	Total momentum, s	Error, 10^{-3}		
					x	*y*	mean
0	PWM	27	61.6	1664	19	19	19
1	0.0009	20	143.2	2864	17	19	18
2	0.0006	18	121.7	2192	15	7.6	11.3
3	0.0003	25	129.9	3248	13	12	12.5
4	0.0001	89	46.2	4112	6.7	4	5.35
5	0.00001	166	39.8	6608	9.8	2.5	6.15

Table 4. **Performance metrics for AI-4 in case of RL with varying action weights**

No	$\tilde{L}_1/\tilde{L}_2$	Number of TH firings	Mon	Total momentum, s	Error, 10^{-3}		
					x	*y*	mean
0	PWM	27	61.6	1664	19	19	19
1	0.0009/0	17	186.3	3168	15	15	15
2	0.0006/0	21	155.4	3264	15	7.3	11.2
3	0.0003/0	18	183.1	3296	13	6.5	9.75
4	0.0001/0	53	73.3	3888	4.9	6.1	5.5
5	0.00001/0	134	42.0	5632	5.6	11	8.3

formance metrics for these cases are presented in Table 3. These cases demonstrate that adding to the state vector X_k additional information about the control action on the previous control step and the control cycle number allows the agent to outperform the PWM-based controller in terms of control accuracy and the number of TH firings.

Fig. 4 show the variation of the normalized in-plane state vector and the TH thrust for AI-4 after being trained by RL with variable action weights. In these cases, control actions are only penalized if a new TH firing happens. This feature of the cost function encourages the agent to limit the number of TH firings. Performance metrics for these cases are presented in Table 4. These cases demonstrate that the variable action weights allow the agent to improve control performance in terms of numbers of the TH firing.

Comparing all four AIs, we can conclude that additional input information about sample ordering allows the agent to improve control accuracy while the information about the control actions on the previous control step in conjunction with the cost function with time-varying weights makes the agent more efficient in terms of TH firings.

This section presents results only for in-plane control because this case is more complex than out-of-plane control. In the case of in-plane control, we deal with a coupled multi-input – multi-output system, but in out-of-plane case, we just have a single-input – single-output system. Moreover, RL-trained agents for out-of-plane control demonstrate similar to the in-plane agents' behavior and performance.

CONCLUSION

The article studies the cases of the spacecraft relative on-off control when the resolution of the TH thrust does not allow a PWM to accurately approximate the linear control laws. For such cases, it is proposed to use RL to directly obtain policies of the TH firings for the spacecraft relative control.

A model of the relative motion of two satellites was built, which describes the control task in the form of a Markov decision process. For the RL-based training, a cost function with variable action weights is proposed, which allows the agent to optimize the number of TH firing in an explicit manner. To improve the control performance, it is proposed to extend the input state vector by the information about the control actions on the previous control step and the control cycle numbers.

Numerical results demonstrate that the reinforcement learning methodology can be used to design on-off relative controllers, which outperform conventional DLQR with PWM in terms of control accuracy, settling time, and numbers of the TH firings.

Acknowledgements. *The authors thank the National Academy of Sciences of Ukraine, the Fundamental Research Funds for the Central Universities of China (No. D5000220031), and the Key Research and Development Program of Shaanxi of China (No. 2023-GHZD-32) for their support of this study.*

REFERENCES

1. Alpatov A. P., Cichocki F., Fokov A. A., Khoroshylov S. V., Merino M., Zakrzhevskii A. E. (2015). Algorithm for determination of force transmitted by plume of ion thruster to orbital object using photo camera. *66th Int. Astronautical Congress* (Jerusalem, Israel), 2239—2247.
2. Alpatov A., Khoroshylov S., Lapkhanov E. (2020). Synthesizing an algorithm to control the angular motion of spacecraft equipped with an aeromagnetic deorbiting system. *Eastern-Eur. J. Enterprise Technol.*, **5** (103), 37—46.
3. Anthony T., Wie B., Carroll S. (1989). Pulse-modulated control synthesis for a flexible spacecraft. *J. Guid., Contr., and Dyn.*, **13** (6), 1014—1022.
4. *Artificial intelligence: a modern approach* (2010). Eds. S. J. Russell, P. Norvig. Pearson education. Inc. ISBN-13: 978-0134610993.
5. Bernelli-Zazzera F., Mantegazza P., Nurzia V. (1998). Multi-pulse-width modulated control of linear systems. *J. Guid., Contr., and Dyn.*, **21** (1), 64—70.
6. *Deep Learning* (2016). Eds. I. Goodfellow, Y. Bengio, A. Courville. The MIT press. ISBN 978-0262035613.
7. Gaudet B., Linares R., Furfaro R. (2020). Adaptive guidance and integrated navigation with reinforcement meta-learning. *Acta astronaut.*, **169**, 180—190.

8. Gaudet B., Linares R., Furfaro R. (2020). Seeker based adaptive guidance via reinforcement meta-learning applied to asteroid close proximity operations. *Acta astronaut.*, **171**, 1—13.
9. Golubek A. V., Dron M. M., Petrenko O. M. (2023). Estimation of the possibility of using electric propulsion systems for large-sized orbital debris post-mission disposal. *Space Science and Technology*, **29** (3), 34—46. DOI: 10.15407/knit2023.03.034
10. Hovell K., Ulrich S. (2020). On deep reinforcement learning for spacecraft guidance. AIAA SciTech Forum, 6—10 January 2020, Orlando, FL. DOI: 10.2514/6.2020-1600.
11. Ieko T., Ochi Y., Kanai K. (1997). A new digital redesign method for pulse-width modulation control systems. *AIAA proc. AIAA-97*, 3700.
12. Izzo D., Märtens M., Pan B. (2019). A survey on artificial intelligence trends in spacecraft guidance dynamics and control. *Astrodyn.*, **3**, 287—299. DOI: 10.1007/s42064-018-0053-6.
13. Khoroshylov S. V. (2018). Relative motion control system of spacecraft for contactless space debris removal. *Nauka innov.*, **14** (4), 5—16.
14. Khoroshylov S. V., Redka M. O. (2019). Relative control of an underactuated spacecraft using reinforcement learning. *Techn. Mechanics*, **4**, 43—54.
15. Khoroshylov S. V., Redka M. O. (2021). Deep learning for space guidance, navigation, and control. *Space Science and Technology*, **27** (6), 38—52.
16. Khosravi A., Sarhadi P. (2016). Tuning of pulse-width pulse-frequency modulator using PSO: An engineering approach to spacecraft attitude controller design. *Automatika*, No 57, 212—220.
17. Lapkhanov E., Khoroshylov S. (2019). Development of the aeromagnetic space debris deorbiting system. *Eastern-European J. Enterprise Technologies*, **5** (101), 30—37.
18. Lewis F. L., Vrabie D., Syrmos V. L. (2012). *Optimal Control* (3rd ed.). New York: John Wiley & Sons, Inc.
19. Li W., Cheng D., Liu X., et al. (2019). On-orbit service (OOS) of spacecraft: A review of engineering developments. *Progress in Aerospace Sci.*, **108**, 32—120.
20. *Machine Learning* (1997). Ed. T. Mitchell. New York: McGraw Hill. ISBN 0070428077.
21. Mnih V., Badia A., Mirza M., Graves A., Lillicrap T., Harley T., Silver D. (2016). Asynchronous Methods for Deep Reinforcement Learning. *arXiv preprint, ArXiv:1602.01783.*
22. OestreichC.E., Linaresy R., Gondhalekarz R. (2021). Autonomous six-degree-of-freedom spacecraft docking maneuvers via reinforcement learning. *J. Aerospace Inform. Syst.*, **18(**7). DOI: 10.2514/1.I010914.
23. Redka M. O., Khoroshylov S. V. (2022). Determination of the force impact of an ion thruster plume on an orbital object via deep learning. *Space Science and Technology*, **28**(5), 15—26.
24. *Reinforcement learning: an introduction* (1998). Eds. R. S. Sutton, A. G. Barto. MIT press. ISBN 978-0262193986.
25. Robinett R. D., Parker G. G., Schaub H., Junkins J. (1997). Lyapunov optimal saturated control for nonlinear systems. *J. Guid., Contr., and Dyn.*, **20** (6), 1083—1088.
26. Schulman J., Wolski F., Dhariwal P., Radford A., Klimov O. (2017). Proximal policy optimization algorithms. *arXiv preprint, arXiv:1707.06347.*
27. Silver D., Schrittwieser J., Simonyan K. (2017). Mastering the game of Go without human knowledge. *Nature*, **550**, 354—359. DOI: 10.1038/nature24270.
28. Song G., Buck N. V., Agrawal B. N. (1999). Spacecraft vibration reduction using pulse-width pulse-frequency modulated input shaper. *J. Guid., Contr., and Dyn.,* **22** (6), 433—440.
29. Yamanaka K., Ankersen F. (2002). New State Transition Matrix for Relative Motion on an Arbitrary Elliptical Orbit. *J. Guid., Contr., and Dyn.*, **25** (1), 60—66.

Стаття надійшла до редакції 18.12.2023 *Received 18.12.2023*
Після доопрацювання 18.01.2024 *Revised 18.01.2024*
Прийнято до друку 30.01.2024 *Accepted 30.01.2024*

С. В. Хорошилов [1], пров. наук. співроб., д-р техн. наук, проф.
ORCID.org/0000-0001-7648-4791
E-mail: skh@ukr.net,
Ч. Ван [2], проф., PhD
ORCID.org/0000-0002-3789-8614

[1] Інститут технічної механіки Національної академії наук України та Державного космічного агентства України
вул. Лешко-Попеля 15, Дніпро, Україна, 49005
[2] Північно-Західний політехнічний університет
Сіань Шаньсі, 710072, Китай

РЕЛЕЙНЕ КЕРУВАННЯ ВІДНОСНИМ РУХОМ КОСМІЧНИХ АПАРАТІВ З ВИКОРИСТАННЯМ НАВЧАННЯ З ПІДКРІПЛЕННЯМ

Розглядається задача керування відносним рухом космічних апаратів за допомогою реактивних установок, вихід яких має два стани: «увімкнено» та «вимкнено». Для випадків, коли роздільна здатність реактивних двигунів не забезпечує якісну апроксимацію лінійних законів керування з використанням широтно-імпульсного модулятора тяги, досліджено можливість застосування навчання з підкріпленням для прямого пошуку законів керування, що встановлюють зв'язок між вектором стану і командами вмикання-вимикання реактивних двигунів. Для реалізації такого підходу отримано модель керованого відносного руху двох супутників у формі марківського процесу прийняття рішень. Інтелектуальний агент представлено у вигляді нейромережевого «виконавця» та «критика» та визначено архітектури цих модулів. Запропоновано використовувати функцію вартості зі змінними ваговими коефіцієнтами керівних впливів, що дозволяє оптимізувати кількість увімкнень реактивних двигунів явним чином. Для підвищення якості керування запропоновано використовувати розширений вектор входу для нейромережевого виконавця та критика інтелектуального агента, який крім вектора стану ще містить інформацію про керівну дію на попередньому такті керування та номер такту керування. Для зменшення часу навчання використано попереднє навчання агента на даних, отриманих за допомогою традиційних алгоритмів керування. Чисельні результати демонструють, що використання методології навчання з підкріпленням дозволяє перевершити результати, що забезпечуються лінійним контролером із широтно-імпульсним модулятором, з точки зору точності керування, швидкодії та кількості включень реактивних двигунів.

Ключові слова: релейне керування, навчання з підкріпленням, відносне керування космічним апаратом, виконавець, критик, нейронна мережа, включення реактивного двигуна.