Eng. Sci., 2024, **29**, 1130

Engineered Science

DOI: https://dx.doi.org/10.30919/es1130

System Synthesis for Motion along the Trajectory by Evolutionary Machine Learning Control

Askhat Diveev,[1,*] Elena Sofronova,[1,*] Nurbek Konyrbaev[2,*] and Sabit Ibadulla[2]

Abstract

The article discusses the problem of synthesizing a system for stabilising the movement of an object along a given trajectory. Solving the control synthesis problem involves finding the control function on the deviation of the object from a given trajectory. A trajectory stabilisation system is necessary for the object to maintain its trajectory under real conditions in the presence of external disturbances. In the work, machine learning control by symbolic regression was used to solve the control function synthesis problem. Symbolic regression methods allow to find the mathematical expressions of the desired functions in the form of special code. To find the mathematical expression of the desired function, the symbolic regression method uses a special genetic algorithm that searches the code space for the optimal solution according to the given optimisation criterion. An example of motion stabilisation of two quadcopters along optimal trajectories is presented.

Keywords: Control synthesis; Evolutionary computing; Machine learning control; Symbolic regression; Quadcopter.
Received: 15 January 2024; Revised: 25 March 2024; Accepted: 27 March 2024.
Article type: Research article.

1. Introduction

When controlling groups of robots, it is often necessary to change their position. In this case, the robots have to move in a space with obstacles and must not collide with each other. If the control quality criterion for a given problem is known, and in this case this criterion can be the time taken for the robots to reach new states, then such a problem can be easily formulated as an optimal control problem and its solution can be obtained numerically on the robot's on-board computer in real time, taking into account the existing phase parameters, restrictions defining the areas where obstacles are located, and conditions for the absence of collisions between robots. Control functions as functions of time for each robot are the solution to this problem. These control functions, inserted into the robot model, ensure that the simulation results in program trajectories. However, when these functions are inserted into real robots, the real robots will not move along the calculated trajectories due to the inaccuracy of the model and external disturbances. This is where the second problem arises, the construction of a stabilisation system for the movement of the robot along a given program trajectory. Such a stabilisation system should take into account the deviation of the robot's real position from the given program trajectory, *i.e.* the system for stabilising the robot's movement along a given trajectory is a function whose argument is the vector of the robot's state space. Finding such a function requires solving the control synthesis problem. This problem cannot be solved in real time on the robot's on-board processor. The motion stabilisation function must be built in advance at the design stage of the robot control system, and this function must ensure the stabilisation of the robot's motion along any given program trajectory.

In this paper, the symbolic regression method is used to solve the control synthesis problem. This method makes it possible to find a mathematical expression for the desired function. In this case, a control function has been obtained to stabilise the movement of the robot along the program trajectory. This control function is inserted into the control system of each robot. Later in the process, when one or more robots need to move to new positions, the optimal control problem is formulated and solved, possibly on the robot's on-board computer. The resulting solution is inserted into a reference model of the robot, also implemented on the on-board processor. The reference model generates a program

[1] *Department of Robotics Control, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 119333, Vavilova Str., 44, Build. 2, Moscow, Russia.*
[2] *Department of Computer Science, Institute of Engineering and Technology, Korkyt Ata Kyzylorda University, 120014, Aiteke bi Str., 29A, Kyzylorda, Kazakhstan.*
**Email:* aidiveev@mail.ru (A. Diveev), sofronova_ea@mail.ru (E. Sofronova), n.konyrbaev@mail.ru (N. Konyrbaev)

trajectory. The deviation of the robot's actual position from the program trajectory is fed to a previously synthesised stabilisation system for movement along the program trajectory. The result of the operation of the motion stabilisation system is the value of the control vector which ensures that the robot moves close to the program trajectory.

To ensure the control object movement along a given trajectory, it is usually necessary to solve the trajectory tracking problem[1-6], using model predictive control,[7,8] sliding mode control,[9,10] *etc*., paying special attention to dynamic, adaptive and smooth trajectory tracking.[11-15]

This is typically done by stabilising the object relative to a point in the state space and then placing these points along a given trajectory. The control object moves along the trajectory from one active point to the next one. Since the object is stable relative to a point on the trajectory, model inaccuracies and small disturbances have little effect on the motion of the object. The disadvantage of this approach is that the speed of the object along the trajectory is not controlled. Note that the speed of the control object slows down near the stable equilibrium point and is zero at the point itself. If the points are placed far apart from the trajectory, then the object between the points may leave the trajectory.

Another problem is that the solution of the control synthesis problem to ensure the stability of an object relative to a certain point in the state space is usually solved manually, based on the analysis of the control object model and control channels, without optimizing the quality criterion.[16] This approach cannot be called a universal numerical method.

There are several approaches to solving the problem of stabilising movement along a given spatial trajectory. One approach is to transform the kinematic models into the chained form.[17,18] Another approach is to bring the system of differential equations describing the controlled movement of the robot into a form that allows linearization by feedback using variable substitution, and then solve the problem of stabilising control synthesis taking into account phase constraints and control constraints.[19,20]

The problem of stabilisation system synthesis for movement along an optimal trajectory by machine learning control was first considered in Ref. [21]. A control system was obtained that included a reference model for generating an optimal trajectory over time. Research has shown that the stabilisation system depends on the type of optimal trajectory. To overcome this drawback, a universal stabilisation system was proposed in Ref. [22]. The universality of the stabilisation system assumes that a stabilisation system is constructed for an object moving along different trajectories belonging to the same class. To do this, we first solve the problem of stabilisation system synthesis for several given trajectories. In this case, the training set is the class of trajectories, and training the control system for the training set is the synthesis of the stabilisation system. The optimal control problem is solved for an object model that includes a stabilisation system and a subsystem with a reference model for generating a given trajectory. The resulting stabilisation system is used to move along a trajectory that is not included in the training set. To solve the problem of stabilisation system synthesis symbolic regression methods are used.[23,24]

The work presents a universal computational algorithm for solving the problem of stabilisation system synthesis for movement of an object along a given trajectory. The algorithm is based on machine learning control by the network operator method,[25] one of symbolic regression methods. Symbolic regression allows to find mathematical expressions of control functions. When this function is inserted into the mathematical model of control object instead of the control vector, it changes the model in such a way that it has a particular solution that almost coincides with the given trajectory. This particular solution of the system of differential equations is a special one because it has a neighborhood with an attractor property. The obtained control system with the stabilisation system for the movement along the given trajectory in feedback is insensitive to small external disturbances and to small changes of an initial state, since the attractor property of the particular solution reduces the error between the states of the real object and the model.

2. The motion stabilisation system synthesis problem

The mathematical model of control object in the form of ODE system is given

$$\dot{\mathrm{x}} = \mathrm{f}(\mathrm{x}, \mathrm{u}) \tag{1}$$

where x is a vector of a control object state, $\mathrm{x} \in \mathbb{R}^n$, u is a control vector, $\mathrm{u} \in \mathrm{U} \subseteq \mathbb{R}^m$, U is a compact set, that often defines constraints on control

$$\mathrm{u}^- \leq \mathrm{u} \leq \mathrm{u}^+ \tag{2}$$

u^-, u^+ are lower and upper boundaries of control vector values respectively.

The initial state is given

$$\mathrm{x}(0) = \mathrm{x}^0 = [x_1^0 \dots x_n^0]^{\mathrm{T}} \tag{3}$$

The program trajectory is given

$$\mathrm{x}^*(t) = [x_1^*(t) \dots x_n^*(t)]^{\mathrm{T}}, t \in [0; t_f] \tag{4}$$

where $\mathrm{x}^*(0) = \mathrm{x}^0$, t_f is a given terminal time.

It is necessary to find a control function in the form

$$\mathrm{u} = \mathrm{h}(\mathrm{x}^* - \mathrm{x}) \in \mathrm{U} \tag{5}$$

to minimize the following quality criterion

$$J_0 = \int_0^{t_f} \|\mathrm{x}^*(t) - \mathrm{x}(t, \mathrm{x}^0)\|_2 \, dt \to \min, \tag{6}$$

where $\mathrm{x}(t, \mathrm{x}^0)$ is a particular solution from the initial state (2) of the control object mathematical model (1) with the inserted control function (5)

$$\dot{\mathrm{x}} = \mathrm{f}\big(\mathrm{x}, \mathrm{h}(\mathrm{x}^* - \mathrm{x})\big). \tag{7}$$

In another problem statement the trajectory (4) is given in the space as one-dimension manifold

$$\mathrm{g}_i(\mathrm{x}) = 0, \quad i = 1, \dots, n-1, \tag{8}$$

where $\mathrm{g}_i(\mathrm{x}^0) = 0, i = 1, \dots, n-1$.

In this problem statement the terminal state should be given

$$\mathrm{x}(t_f) = \mathrm{x}^f = [x_1^f \dots x_n^f]^{\mathrm{T}}, \tag{9}$$

where $\mathrm{g}_i(\mathrm{x}^f) = 0, i = 1, \dots, n-1$.

Here, initially, it is necessary to obtain the reference model. This can be done, if the following optimal control problem is solved in the classical statement.

The mathematical model is given (1). The initial state is given (3). The terminal state is given (9). It is necessary to find a control function of time

$$\mathrm{u} = \mathrm{u}^*(t) \in \mathrm{U}\,, t \in [0; t_f], \tag{10}$$

to minimize the following quality criterion

$$J_1 = \int_0^{t_f} \sqrt{\sum_{i=1}^{n-1} \mathrm{g}_i^2(\mathrm{x}^*(t, \mathrm{x}^0))}\, dt \to \min\,, \tag{11}$$

where $\mathrm{x}^*(t, \mathrm{x}^0)$ is a particular solution of the ODE system from the initial state (2)

$$\dot{\mathrm{x}}^* = \mathrm{f}(\mathrm{x}^*, \mathrm{u}^*(t)). \tag{12}$$

In the third case the trajectory is given as the set of points in the state space

$$\mathrm{X}^* = \{\mathrm{x}^{*,0}, \ldots, \mathrm{x}^{*,k}, \ldots, \mathrm{x}^{*,N}\}, \tag{13}$$

where $\mathrm{x}^{*,0} = \mathrm{x}^0, \mathrm{x}^{*,N} = \mathrm{x}^f$.

Then in order to obtain the trajectory in the form (4) it is necessary to solve the optimal control problem (1), (2), (3), (9), (10) with the foll owing quality criterion

$$J_2 = \sum_{k=0}^{N} \min_{t\in[0;t_f]} \left\|\mathrm{x}^{*,k} - \mathrm{x}^*(t, \mathrm{x}^0)\right\| \to \min\,, \tag{14}$$

where $\mathrm{x}^*(t, \mathrm{x}^0)$ is a particular solution of the reference model (12) or the given trajectory (4) in the main problem.

So, in all cases the given trajectory is obtained in the form (4). The given trajectory is a set of functions of time. For calculations it is necessary store each function in the form a set of points in some moments of time

$$\mathrm{X}^* = \{\mathrm{X}_1^*, \ldots, \mathrm{X}_n^*\}\,, \tag{15}$$

where

$$\mathrm{X}_i^* = \{x_i^*(t_0), x_i^*(t_1), \ldots, x_i^*(t_k), \ldots, x_i^*(t_f)\}, i = 1, \ldots, n, \tag{16}$$

$$t_j = t_{j-1} + \Delta t, j = 1, \ldots, N, \tag{17}$$

$\Delta t = \left\lfloor \frac{t_f}{N} \right\rfloor$, $N + 1$ is a given number of points, $t_0 = 0$, $x_i^*(t_0) = x_i^*(0) = x_i^0, x_i^*(t_f) = x_i^f$, $i = 1, \ldots, n$.

Storing the given trajectory as a set of numbers in time is inconvenient for calculations, because it is can take up a lot of memory. It is better to use the reference model. If there is no reference model, as in the first case, then it can be obtained as the solution of the following optimal control problem (1), (2), (3), (6) with a control function as a function of time (10). Here it is necessary to keep the control function (10) for the reference model (12) instead of functions of the given trajectory (4). This is usually better than storing the functions of the given trajectory, because the dimension of control vector is smaller than the dimension of the state space vector. The given trajectory is continuous, therefore more point are needed to store it than to store the control function, which may have a first order function discontinuity.

Now, the synthesis problem of motion stabilisation along the given trajectory has the following form. The mathematical model of control object (1), the control constraints (2), the initial state (3), the terminal state (9), the reference model (12), and the control function for the reference model (11) are given. It is necessary to find a control function as in (5), where x^* is a solution of the reference model (12) for the control function (11), such that a value of the quality criterion (6) is minimal. To solve the control synthesis problem (1) – (3), (5), (6), (9), (11), (12) a machine learning control by symbolic regression is used. If a solution to the control synthesis problem is found only for one initial state, then the motion stabilisation system will be sensitive to changes in the initial states, so it is necessary to include some additional conditions in the statement of the control synthesis problem. An initial state is replaced by the set of initial states from a neighborhood of the given initial state

$$\mathrm{X}_0 = \{\mathrm{x}^{0,1}, \ldots, \mathrm{x}^{0,K}\}\,, \tag{18}$$

where

$$\left\|\mathrm{x}^{0,i} - \mathrm{x}^0\right\|_2 \le \Delta\,, \tag{19}$$

Δ is a positive value that defines a domain of low sensitivity to changes in the initial state. Relatively, the quality criterion (6) is transformed into the sum of the criterion values for each initial state

$$J_4 = \sum_{i=1}^{K} \int_{\mathbf{0}}^{t_{f,i}} \left\|\mathbf{x}^*(t) - \mathbf{x}(t, \mathbf{x}^{0,i})\right\|_2 dt \to \min\,, \tag{20}$$

where $t_{f,i}$ is a time of reaching the terminal state (10) from the initial state $\mathbf{x}^{0,i}$

$$t_{f,i} = \begin{cases} t, \text{ if } t < t^+ \text{ and } \left\|\mathbf{x}^f - \mathbf{x}(t, \mathbf{x}^{0,i})\right\| \le \varepsilon \\ t^+, \text{ otherwise} \end{cases}, i = 1, \ldots, K\,, \tag{21}$$

ε, t^+ are given positive values, $t^+ \ge t_f$.

3. Symbolic regression for search of mathematical expression

Symbolic regression is a relatively new computational method that, like artificial neural networks, is a universal approximator of any function. An artificial neural network is a function with a given structure with a large number of configurable parameters. The function approximation by the artificial neural network is performed by finding the corresponding parameter values. Searching for parameter values in an artificial neural network is called learning it. Symbolic regression, unlike an artificial neural network, allows to find not only the parameters of a function, but also its structure. All symbolic regression methods encode the function in the form of a special code. To encode a function, symbolic regression methods use an alphabet of elementary functions. The search for the desired function is carried out by a special genetic algorithm in the code space. In the genetic algorithm, the basic crossover operation is specifically designed for codes of the symbolic regression method so that, as a result of this crossover operation, a new correct code of symbolic regression method is obtained.

In this work, symbolic regression is used to solve the problem of synthesis a system for movement stabilization along a trajectory. Note that when solving any control synthesis problem, a feedback function is sought for. This function which is inserted into the right parts of the differential equation system of the control object model. Any function in the feedback significantly changes the dynamics of the object. Since the feedback control function is not known in advance, it is very difficult to obtain a high-quality training sample for the control synthesis problem. Therefore, an artificial neural network is not used to solve the control synthesis problem.

Here, the network operator method[25] is used to solve the control synthesis problem. This symbolic regression method encodes a mathematical expression as an oriented graph. The method uses only functions with one and two arguments. Let us consider an example of encoding a mathematical expression using a network operator. Let the following mathematical expression be specified

$$y = x_1 \exp\left(-q_2 x_2\right) \sin\left(x_2 \cos(q_1 x_1 + q_2)\right) \quad (22)$$

To encode this mathematical expression, the alphabet of elementary functions of the network operator method must include the following functions:

- functions with one argument

$$\mathrm{F}_1 = \{f_{1,1}(z) = z,\ f_{1,2}(z) = -z,\ f_{1,3}(z) = \exp(z),\ f_{1,4}(z) = \sin(z),\ f_{1,5}(z) = \cos(z)\}; \quad (23)$$

- functions with two arguments

$$\mathrm{F}_2 = \{f_{2,1}(z_1, z_2) = z_1 + z_2,\ f_{2,2}(z_1, z_2)\}; \quad (24)$$

-arguments of mathematical expression or functions without arguments

$$\mathrm{F}_0 = \{f_{0,1} = x_1,\ f_{0,2} = x_2,\ f_{0,3} = q_1,\ f_{0,4} = q_2\}. \quad (25)$$

All elements of the alphabet have two lower indexes, the first index is a number of arguments, the second index is the element number in the set. The set of functions with one argument (23) includes identity function $f_{1,1}(z) = z$. All functions with two arguments (24) are commutative, associative, and have their own unit element.

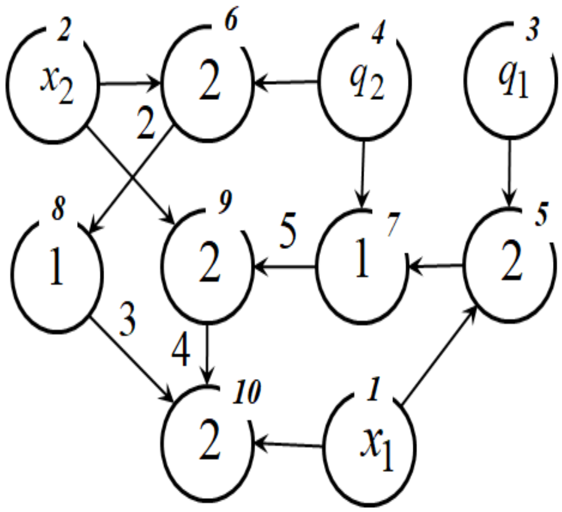

Fig. 1 The network operator graph for the mathematical expression (22)

Figure 1 shows the network operator graph of the mathematical expression (22). On the graph, the source nodes are associated with the arguments of the mathematical expression. The remaining nodes are associated with functions of two arguments. The numbers of function of two arguments are specified in these nodes. Graph arcs are associated with functions with one argument. Near the arcs are function numbers with one argument. If there is no function number near the arc, then it is associated with the identity function $f_{1,1}(z) = z$. All nodes are indexed in the upper parts. The node indices are topological sorted. It means that the index of the node from which an arc exits is less, than the index of the node where it enters. The topological sorting is always possible for the graph without loops.

In the computer memory, the network operator is stored as an integer matrix. Each row of the matrix corresponds to a node of the graph. If the node numbers of the graph are topologically sorted, then the matrix has an upper triangular form. The network operator presented in Fig. 1 has the following matrix

$$\Psi = [\psi_{i,j}] = \begin{bmatrix} 0&0&0&0&1&0&0&0&0&1 \\ 0&0&0&0&0&1&0&0&1&0 \\ 0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&1&1&0&0&0 \\ 0&0&0&0&2&0&1&0&0&0 \\ 0&0&0&0&0&2&0&2&0&0 \\ 0&0&0&0&0&0&1&0&5&0 \\ 0&0&0&0&0&0&0&1&0&3 \\ 0&0&0&0&0&0&0&0&2&4 \\ 0&0&0&0&0&0&0&0&0&2 \end{bmatrix} \quad (26)$$

where $i, j = 1, \ldots, L$, L is a number of nodes in the graph, $L = 10$.

On the main diagonal are zeros or the numbers of functions with two arguments. Zero specifies on node-source of the graph. Remain non-zero elements in the matrix are the numbers of functions with one argument. If $\psi_{i,j} \neq 0$, then between the node i and node j is the arc from node i to node i associated with the function with one argument under number $\psi_{i,j}$.

In order to calculate the mathematical expression by the network operator matrix, a nodes vector is used. Each component of nodes vector is associated with a network operator node. The number of elements in the nodes vector is equal to the number of nodes. The initial value of the nodes vector consists of arguments of a mathematical expression and unit elements of functions with two arguments.

The initial value of the nodes vector of the network operator matrix (26) is

$$\mathrm{z}^{(\mathbf{0})} = [\mathrm{z}_1^{(0)} \ldots z_{10}^{(0)}]^{\mathrm{T}} = [x_1\ x_2\ q_1\ q_2\ 1\ 1\ 0\ 0\ 1\ 1]^{\mathrm{T}}. \quad (27)$$

Here 1 is unit element for the multiplication function $f_{2,2}(z_1, z_2)$, 0 is unit element for the function addition $f_{2,1}(z_1, z_2)$. Next, all rows of the network operator matrix are sequentially viewed. The component values of the nodes vector change according to the following formula

$$\mathrm{z}_j^{(i)} \leftarrow \begin{cases} f_{2,\psi_{j,j}}\left(z_j^{(i-1)}, f_{1,\psi_{i,j}}\left(z_i^{(i-1)}\right)\right), \text{if } \psi_{i,j} \neq 0 \\ \mathrm{z}_j^{(i-1)},\ \text{otherwise} \end{cases}, \quad (28)$$

where $i = 1, \ldots, L-1$, $j = L+1, \ldots, L$.

The nodes vector (27) after viewing the penultimate row of the matrix (26) has the following form

$$\mathrm{z}^{(9)}=\begin{bmatrix} x_1 \\ x_2 \\ q_1 \\ q_2 \\ x_1q_1 \\ x_2q_2 \\ x_1q_1+q_2 \\ -x_2q_2 \\ x_2\sin(x_1q_1+q_2) \\ x_1\exp(-x_2q_2)\cos(x_2\sin(x_1q_1+q_2)) \end{bmatrix}. \quad (29)$$

The last component of the nodes vector (29) is equal to the mathematical expression (22).

At the search of the mathematical expression the network operator applies the variational genetic algorithm. This algorithm uses the principle of small variations of basic solution. According this principle in an initial population only one basic solution is codded by code of the network operator method. All other possible solutions are codded by ordered set of codes of small variations. In the network operator method to code a small variation the integer vector with four components is used

$$\mathrm{w}=[w_1\ w_2\ \ w_3\ \ w_4]^{\mathrm{T}}, \quad (30)$$

where w_1 is a type of a small variation, w_2 is the number of line of the network operator matrix, w_3 is the number of column of the network operator matrix, w_4 is a new value of element of the network operator matrix.

The network operator method uses four types of small variation: $w_1=0$ is an exchange of the function with one argument: if $\psi_{w_2w_3}\neq 0$, then $\psi_{w_2w_3}\leftarrow w_4$; $w_1=1$ is an exchange of the function with two argument: if $\psi_{w_2w_2}\neq 0$, then $\psi_{w_2w_2}\leftarrow w_4$; $w_1=2$ is an insertion of the additional function with one argument: if $\psi_{w_2w_3}=0$, then $\psi_{w_2w_3}\leftarrow w_4$; $w_1=3$ is an elimination of the function with one argument: if $\psi_{w_2w_3}\neq 0$ and $\exists\psi_{w_2,j}\neq 0$, $j>w_2$, $j\neq w_3$ and $\exists\psi_{i,w_3}\neq 0$, $i\neq w_2$, then $\psi_{w_2w_3}=0$.

In the population any possible solution is coded as ordered set of small variation vectors

$$\mathrm{W}_i=(\mathrm{w}^{i,1},\dots,\mathrm{w}^{i,d}).\ i=1,\dots,H, \quad (31)$$

where d is a depth of variations, H is a number of possible solutions in the population except the basic solution.

Any possible solution Ψ_i is a result of application the set of small variation to the basic solution

$$\Psi_i=\mathrm{W}_i\diamond\Psi_0=\mathrm{w}^{i,d}\diamond\dots\diamond\mathrm{w}^{i,1}\diamond\Psi_0, \quad (32)$$

where Ψ_0 is a network operator matrix of the basic solution.

At the performing crossover operation two possible solutions in the form of ordered sets of small variation vectors are selected randomly

$$\mathrm{W}_\alpha=(\mathrm{w}^{\alpha,1},\dots,\mathrm{w}^{\alpha,d}),\ \mathrm{W}_\beta=(\mathrm{w}^{\beta,1},\dots,\mathrm{w}^{\beta,d}). \quad (33)$$

A crossover point is determined randomly, $c\in\{1,\dots,d\}$.

Two new possible solutions are obtained by the exchange of tails after the crossover point of selected possible solutions

$$\mathrm{W}_{H+1}=(\mathrm{w}^{\alpha,1},\dots,\mathrm{w}^{\alpha,c},\mathrm{w}^{\beta,c+1},\dots,\mathrm{w}^{\beta,d}), \quad (34)$$

$$\mathrm{W}_{H+2}=(\mathrm{w}^{\beta,1},\dots,\mathrm{w}^{\beta,c},\mathrm{w}^{\alpha,c+1},\dots,\mathrm{w}^{\alpha,d}). \quad (35)$$

4. Computational experiment

Consider the motion stabilisation system synthesis along the trajectory for two quadcopters. The mathematical model of spatial motion of quadcopters consists of six ordinary differential equations for each control object

$$\dot{x}_1^j=x_4^j, \quad (36)$$

$$\dot{x}_2^j=x_5^j, \quad (37)$$

$$\dot{x}_3^j=x_6^j, \quad (38)$$

$$\dot{x}_4^j=u_4^j\left(\sin(u_3^j)\cos(u_2^j)\cos(u_1^j)+\sin(u_1^j)\sin(u_2^j)\right), \quad (39)$$

$$\dot{x}_5^j=u_4^j\cos(u_3^j)\cos(u_1^j)-\mathrm{g}. \quad (40)$$

$$\dot{x}_6^j=u_4^j\left(\cos(u_2^j)\sin(u_1^j)+\cos(u_1^j)\sin(u_2^j)\sin(u_3^j)\right), \quad (41)$$

where j is the number of quadcopter, g is the acceleration of gravity, $\mathrm{g}=9.80665$,

$\mathrm{x}^j=[x_1^j\dots x_6^j]^{\mathrm{T}}$ is state vector of the quadcopter j, $\mathrm{u}^j=[u_1^j\dots u_4^j]^{\mathrm{T}}$ is the control vector of the quadcopter j, $j=1,2$.

The components of control vector are limited

$$u_1^-=-\frac{\pi}{12}\le u_1^j\le\frac{\pi}{12}=u_1^+, \quad (42)$$

$$u_2^-=-\pi\le u_2^j\le\pi=u_2^+, \quad (43)$$

$$u_3^-=-\frac{\pi}{12}\le u_3^j\le\frac{\pi}{12}=u_3^+, \quad (44)$$

$$u_4^-=0\le u_4^j\le 12=u_4^+. \quad (45)$$

The initial states for quadcopters are given

$$\mathrm{x}^1(0)=[0\ 5\ 0\ 0\ 0\ 0]^{\mathrm{T}},\ \mathrm{x}^2(0)=[0\ 5\ 10\ 0\ 0\ 0]^{\mathrm{T}}. \quad (46)$$

The terminal states are

$$\mathrm{x}^{1,f}=[10\ 5\ 10\ 0\ 0\ 0]^{\mathrm{T}},\ \mathrm{x}^{2,f}=[10\ 5\ 0\ 0\ 0\ 0]^{\mathrm{T}}. \quad (47)$$

There are phase constraints in the spatial of flight

$$\varphi_i(\mathrm{x})=r_i-\sqrt{(a_i-x_1)^2+(b_i-x_3)^2}\le 0, \quad (48)$$

where $i=1,\dots,4$, $r_1=r_2=r_3=r_4=1.5$, $a_1=2$, $b_1=2$, $a_2=8$, $b_2=2$, $a_3=2$,
$b_3=8$, $a_4=8$, $b_4=8$.

The distance between quadcopters is limited

$$\chi(\mathrm{x}^1,\mathrm{x}^2)=r-\sqrt{\sum_{i=1}^{3}(x_i^1-x_i^2)^2}\le 0, \quad (49)$$

where $r=1.5$.

Initially, the problem of optimal control by the criterion of the shortest time to reach the terminal state was solved. The maximum terminal time of the control process was $t^+=5.6$. When solving the optimal control problem, a direct approach was used. All phase constraints and terminal-state accuracy were included in the quality criterion

$$J_4=t_f+p_1\left(\|\mathrm{x}^{1,f}-\mathrm{x}^1(t_f)\|+\|\mathrm{x}^{2,f}-\mathrm{x}^2(t_f)\|\right)+ p_2\sum_{i=1}^{4}\int_0^{t_f}\left(\vartheta(\varphi_i(\mathrm{x}^1))+\vartheta(\varphi_i(\mathrm{x}^2))\right)dt+ p_3\int_0^{t_f}\vartheta\left(\chi(\mathrm{x}^1,\mathrm{x}^2)\right)dt\to\min, \quad (50)$$

where p_1, p_2, p_3 are penalty coefficients, $p_1=1$, $p_2=3$, p_3=3, $t_f=\max\{t_{f,1},t_{f,2}\}$, $t_{f,j}$ is a time of achievement of the terminal state of quadcopter j, $j=1,2$, $\vartheta(\alpha)$ is a Heaviside step function

$$\vartheta(\alpha)=\begin{cases}1, & \text{if }\alpha>0\\ 0, & \text{otherwise}\end{cases}.$$

Coefficient p_1 takes into account the accuracy of each object reaching the terminal state. Coefficient p_2 takes into account the violation of phase constraints; if the phase constraints are violated by one of the objects, then the value of the Heaviside function is equal to 1, respectively, the integral of 1 is equal to the time when the object violates phase constraints. The coefficient p_3 takes into account dynamic phase constraints when objects approach a distance closer than the permissible r. To check the conditions for violation of phase constraints, the Heaviside function is also used. All non-zero values for penalty coefficients are added to the main criterion - the time of the control process.

To solve the problem, a piecewise linear approximation of control functions was used

$$u_i^{j,*}(t) = \begin{cases} u_i^+, & \text{if } \hat{u}_i^{j,*}(t) \geq u_i^+ \\ u_i^-, & \text{if } \hat{u}_i^{j,*}(t) \leq u_i^-, = 1,2, i = 1,\dots,4, \\ \hat{u}_i^{j,*}(t), & \text{otherwise} \end{cases} \quad (51)$$

where

$$\hat{u}_i^{j,*}(t) = q_{i+4(j-1)+8(k-1)} + (q_{i+4j+8(k-1)} - q_{i+4(j-1)+8(k-1)})\frac{t-k\Delta t}{\Delta t}, \quad (52)$$

$j = 1,2,\ i = 1,\dots,4,\ k = 1,\dots,M,\ \Delta t = 0.4,$

$$M = \left\lfloor \frac{t^+}{\Delta t} \right\rfloor = \left\lfloor \frac{5{,}6}{0.4} \right\rfloor = 14\,, \quad (53)$$

$\mathrm{q} = [q_1 \dots q_{2\cdot 4\cdot(M+1)}]^{\mathrm{T}} = [q_1 \dots q_{120}]^{\mathrm{T}}$ is a vector of required parameters.

To solve the optimal control problem, an evolutionary hybrid algorithm[26] was used, which includes a combination of three evolutionary algorithms, a genetic algorithm,[27] particle swarm optimization algorithm[28] and a gray wolf optimizer algorithm.[29] Parameters of hybrid GA: population size – 512, generations – 64, crossovers in one generation - 64; probability of mutation – 0.75, depth of variations – 7. As a result, a solution with a quality criterion value was found $J_4 = 5.63$.

At the second stage, the stabilisation system synthesis problem for motion of a control object along the optimal program trajectory was solved. Reference models were included into the object models that generated the optimal program trajectories. It was necessary to find one stabilization system for both quadcopters. The network operator method was used to solve the problem. Parameters of NOP: dimension 36 x 36, number of differential equations – 24. As a result, the network operator method found the following solution

$$u_i = h_i(\mathbf{x}^* - \mathbf{x}) = \begin{cases} u_i^+, & \text{if } \tilde{u}_i \geq u_i^+ \\ u_i^-, & \text{if } \tilde{u}_i \leq u_i^-, i = 1,\dots,4, \\ \tilde{u}_i, & \text{otherwise} \end{cases} \quad (54)$$

where

$$\tilde{u}_1 = \mu(G) + \rho_{19}(G), \quad (55)$$

$$\tilde{u}_2 = \tilde{u}_1 - \tilde{u}_1^3, \quad (56)$$

$$\tilde{u}_3 = \rho_{19}\big(A + \mu(G)\big) + \rho_{17}(q_1(x_1^* - x_1) + q_4(x_4^* - x_4)) \quad (57)$$

$$\tilde{u}_4 = \tilde{u}_3 + \ln(|\tilde{u}_2|) + \mathrm{sgn}\big(A + \mu(G)\big)\sqrt{|A + \mu(G)|} + \rho_{19}(A) + \arctan\big(D + \tanh(E) + \rho_{18}(H)\big) + \mathrm{sgn}(E) + \arctan(F) + \sqrt{q_1} \quad (58)$$

$A = C + \tanh(D) + \exp(G) + 1,$

$B = C + \mathrm{sgn}(D)\sqrt{|D|},$

$C = \exp\big(D + \tanh(E) + \rho_{18}(H)\big) + \cos(q_6(x_6^* - x_6)),$

$D = E + \sqrt[3]{F} + \sin(q_1(x_1^* - x_1) + q_4(x_4^* - x_4)),$

$E = G + \mathrm{sgn}(x_5^* - x_5) + (x_2^* - x_2)^3 + G^{-1} + \arctan(G) - H,$

$F = G + \mathrm{sgn}(x_5^* - x_5) + (x_2^* - x_2)^3,$

$G = \rho_{17}(q_6(x_6^* - x_6) + q_3(x_3^* - x_3) + x_5^* - x_5) + H^3 + q_1(x_1^* - x_1) + q_4(x_4^* - x_4) + \vartheta(x_5^* - x_5) + (x_5^* - x_5)^2$

$H = \sin\big(q_6(x_6^* - x_6)\big) + q_5(x_5^* - x_5) + q_2(x_2^* - x_2) + \cos(q_1) + \vartheta(x_2^* - x_2),$

$$\mu(\alpha) = \begin{cases} \alpha, & \text{if } |\alpha| < 1 \\ \mathrm{sgn}(\alpha), & \text{otherwise} \end{cases},$$

$\rho_{17}(\alpha) = \mathrm{sgn}(\alpha)\ln(|\alpha| + 1),$

$\rho_{18}(\alpha) = \mathrm{sgn}(\alpha)(\exp(\alpha) - 1),$

$\rho_{19}(\alpha) = \mathrm{sgn}(\alpha)\exp(-|\alpha|),$

$q_1 = 13.20679$, $q_2 = 10.64478$, $q_3 = 9.22241$, $q_4 = 14.14917$, $q_5 = 9.29492$,

$q_6 = 7.82690.$

Figures 2 and 3 show the simulation results for two quadcopters with the obtained stabilisation system. Fig. 2 shows projections of quadcopters trajectories from eight different initial states onto the horizontal plane. Solid black lines are the trajectories of the first quadcopter, dotted lines of the trajectory of the second quadcopter. Red circles are phase constraints. Cyan indicates the program trajectories generated by the reference models. Fig. 3 shows the projections of the same trajectories from eight initial states and the optimal program trajectories on the vertical plane. As it can be seen from the simulation results, the stabilisation system obtained by symbol regression satisfactorily tracks the program trajectories. There is a slight error in the reaching the terminal state.

Figure 4 shows the trajectories of movement of objects (quadrotor 1 a-c, quadrotor 2 d-f) with a stabilisation system without disturbances of the initial conditions (black) and the desired program trajectory (blue), obtained from the reference model, over the remaining coordinates $x_4 - x_6$.

Figure 5 shows real (black) and program (blue) controls for both objects (quadrotor 1 a-d, quadrotor 2 e-h). As it can be seen from the figures the real control does not coincide with the program control, although it retains some features, except for the component u_2. This is due to the fact that the stabilization system did not provide program control tracking. Program control was needed in order to generate a program trajectory. Object control systems tracked only the program trajectory. The large discrepancy between the graphs of real and program control for the component u_2 is explained by the fact that this component practically does not participate in the spatial movement of the center of mass of the quadrotors, since this component is the angle of rotation around the vertical axis, and the position of the center of mass of the object is most influenced by the projection of the force u_4, which depend on the rotation angles u_1 and u_3 around other axes x_1 and x_3.

Computational experiment was performed on PC with

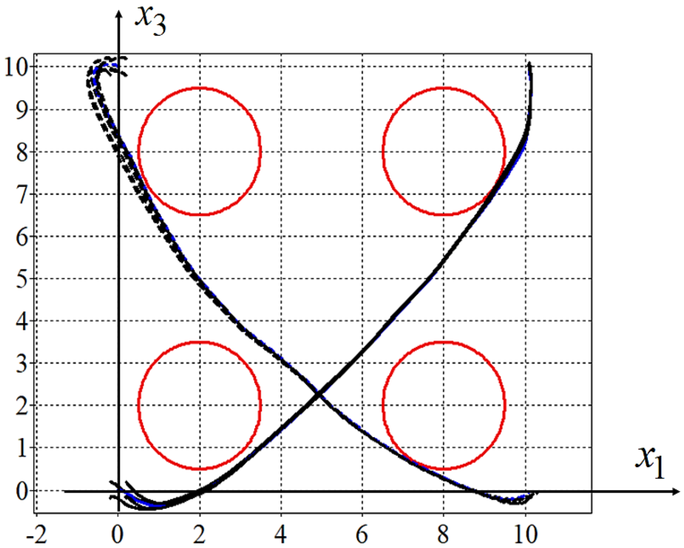

Fig. 2 Projections of trajectories of both quadcopters on the horizontal plane.

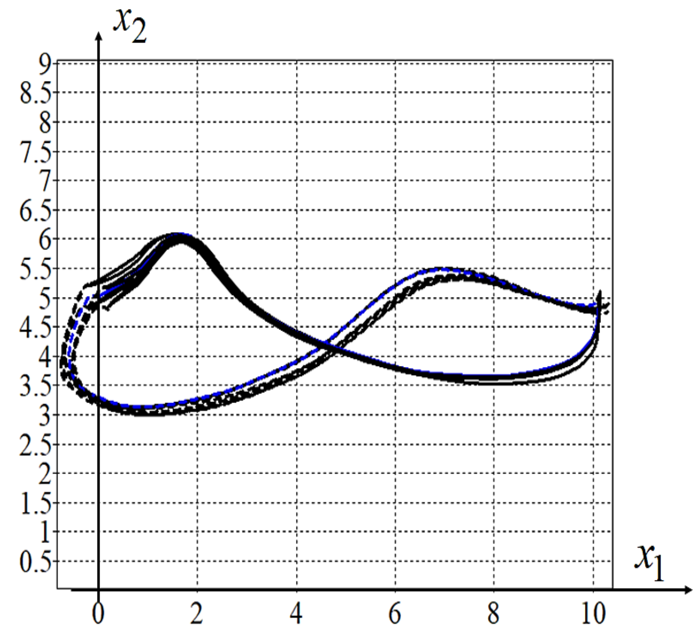

Fig. 3 Projections of trajectories of both quadcopters on the vertical plane.

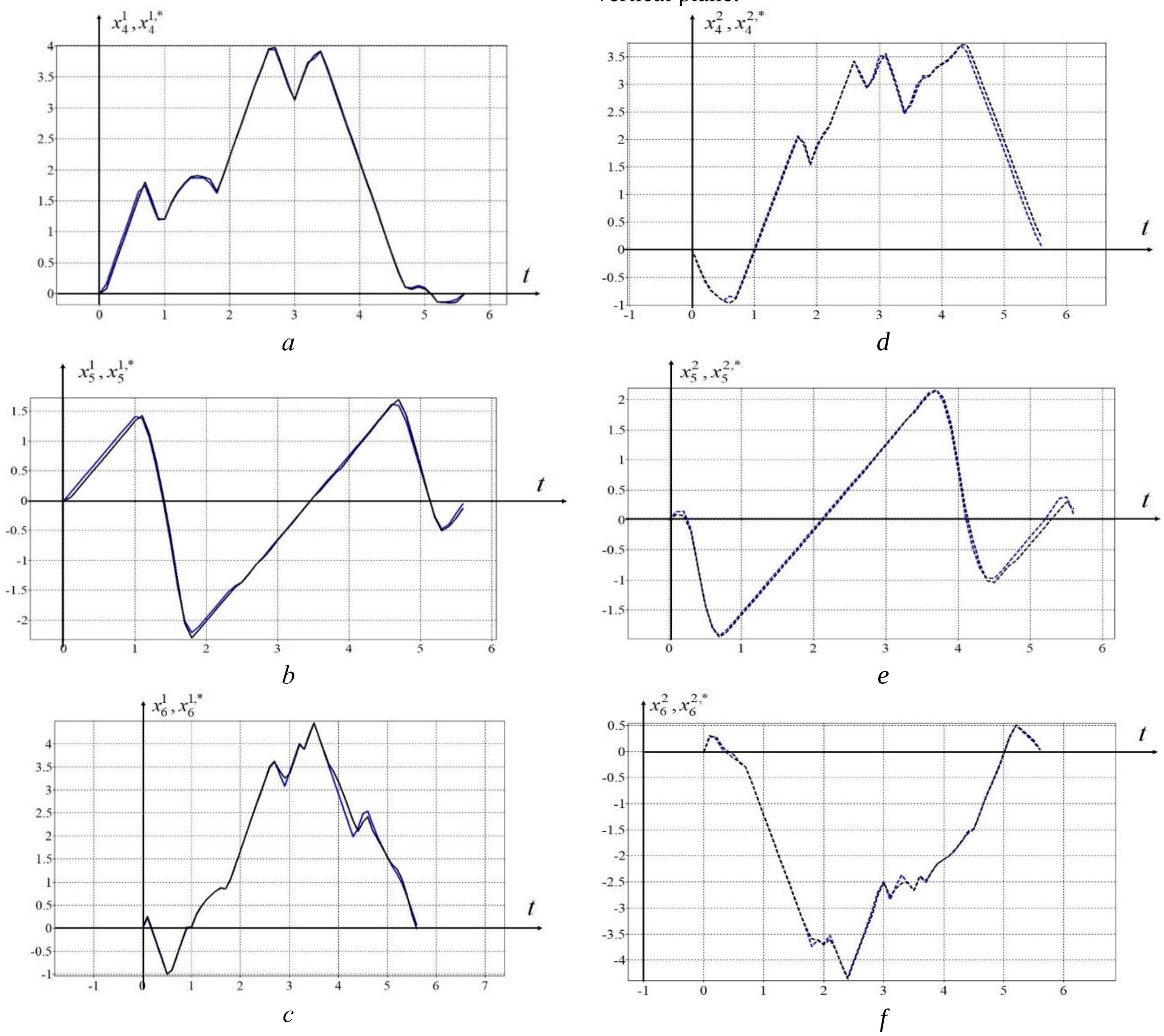

Fig. 4 Real (black) and program (blue) trajectories of quadcopter 1 (solid) and 2 (dots) movement over coordinates $x_4 - x_6$ without disturbance of initial conditions.

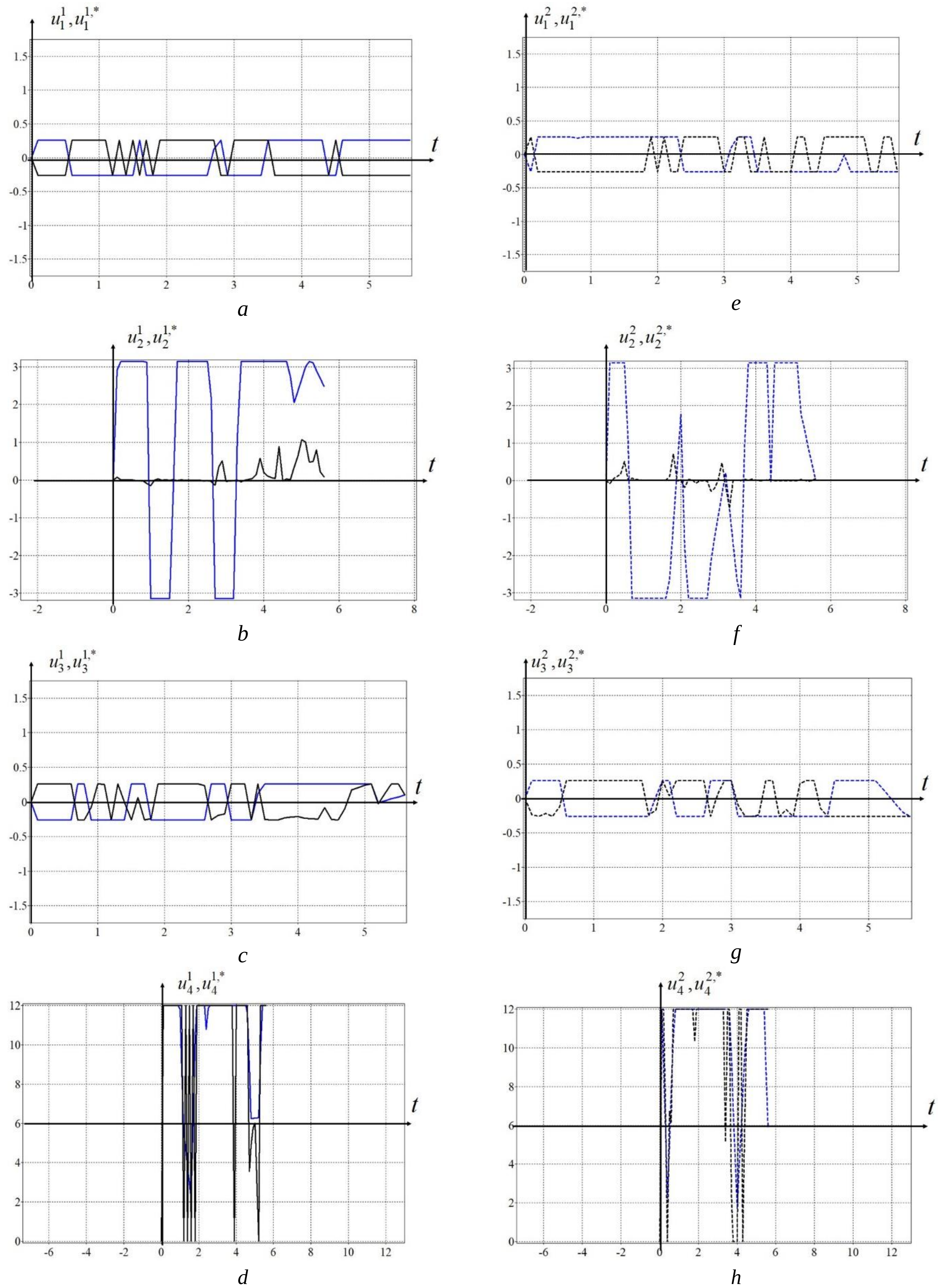

Fig. 5 Real (black) and program (blue) controls of quadcopter 1 (solid) and 2 (dots) without disturbance of initial conditions.

CPU Intel Core i7, 3.3 GHz. Computational time was approx. 4.5 hours.

5. Conclusions & future work

Symbolic regression methods allow to find a mathematical expression of the control function in the form of some code. During the search process to calculate the value of the optimization criterion, a mathematical expression is substituted into the right side of the model of the control object. This approach ensures automatic discovery of the structure and parameters of the control function and eliminates manual operations that are usually performed by the researcher when solving this problem. As a result of the approach based on machine learning, it was possible to obtain a control system that provides high-quality stabilization of the object relative to given trajectories and is little sensitive to disturbances in the initial conditions.

The paper also considers the problem of constructing a reference model for various forms of given trajectories. Three forms of specifying a trajectory are considered: as a function of time, as a one-dimensional manifold in state space, as a set of points in state space. To obtain a reference model for all forms of specifying trajectories, an optimal control problem is solved, in which the control function is found as a function of time. The control function provides a particular solution to the ODE system. This particular solution minimizes the value of the quality criterion. In all three cases, the optimal control problem is solved with a quality criterion that describes the deviation of a particular solution from the given trajectory shapes. As a result of solving the optimal control problem, a reference model is obtained that generates a program trajectory in time.

The statement and numerical solution of the stabilisation system synthesis for the movement of a group of two quadrotors moving in space with static and dynamic obstacles is presented. In the first stage, the optimal control problem with minimization of the time to reach the terminal state was solved using a direct approach through a hybrid evolutionary algorithm. The resulting control function had a piecewise linear form. In the second stage, a stabilisation system for the movement of the quadrocopters relative to the optimal program trajectory obtained in the first stage was synthesised. The synthesis was carried out using the network operator method, which belongs to the class of machine learning control methods. To test the obtained solution, we simulated the movement of a group of quadrocopters with a stabilisation system from different initial conditions. The simulation results showed that the objects moved along the program trajectories without collisions with obstacles and without collisions with each other, and reached the terminal state.

In the future, it is planned to apply this approach to the movement of objects along various trajectories and to study the trajectories of other shapes. Explore different forms of disturbances, such as wind for example. Explore other symbolic regression methods and explore approaches that provide the ability to obtain a more compact control function by limiting the nesting level of function compositions.

Acknowledgements
This research was funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP14869851).

Conflict of Interest
There is no conflict of interest.

Supporting Information
Not applicable.

References
[1] A. Samir, A. Hammad, A. Hafez, H. Mansour, Quadcopter Trajectory Tracking Control using State-Feedback Control with Integral Action, *International Journal of Computer Applications*, 2017, **168**, 1-7, doi: 10.5120/ijca2017914539.
[2] G. Walsh, D. Tilbury, S. Sastry, R. Murray, J. P. Laumond, Stabilization of trajectories for systems with nonholonomic constraints, *IEEE Transactions on Automatic Control*, 1994, **39**, 216-222, doi: 10.1109/9.273373.
[3] Y. Su, Y. Liang, Research on double closed-loop trajectory tracking control algorithm of wheeled mobile robot based on global stability, 2020 12th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC). Hangzhou, China. IEEE, 2020.
[4] N. Uddin, A robot trajectory tracking control system design using pole domination approach, 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). Vancouver, BC, Canada. IEEE, 2018, 506-512, doi: 10.1109/IEMCON.2018.8614899.
[5] A. K. Pamosoaji, Trajectory tracking control strategy using co-reference for rear-steered vehicle, 2018 3rd International Conference on Control and Robotics Engineering (ICCRE). Nagoya, Japan. IEEE, 2018, 74-78, doi: 10.1109/ICCRE.2018.8376437.
[6] A. Andreev, O. Peregudova, K. Sutyrkina, E. Filatkina, On global output feedback trajectory tracking control of a wheeled mobile robot, 2019 23rd International Conference on Mechatronics Technology (ICMT). Salerno, Italy. IEEE, 2019, 1-5, doi: 10.1109/ICMECT.2019.8932122.
[7] X. Yu, W. Zhu, L. Xu, Real-time Motion Planning and Trajectory Tracking in Complex Environments based on Bézier Curves and Nonlinear MPC Controller, 2020 Chinese Control and Decision Conference (CCDC). Hefei, China. IEEE, 2020, 1540-1546, doi: 10.1109/CCDC49329.2020.9163994.
[8] Z. Zhao, Han, Su, Guo, Yang, Anti-collision Trajectory Planning and Tracking Control based on MPC and Fuzzy PID Algorithm, 2020 4th CAA International Conference on Vehicular Control and Intelligence (CVCI). Hangzhou, China. IEEE, 2020, 613-618, doi: 10.1109/CVCI51460.2020.9338447.
[9] T. Su, X. Liang, G. He, Q. Zhao, L. Zhao, Robust trajectory tracking of delta parallel robot using sliding mode control, 2019 IEEE Symposium Series on Computational Intelligence (SSCI). Xiamen, China. IEEE, 2019, 508-512, doi: 10.1109/SSCI44817.2019.9003125.
[10] S. Souissi, M. Boukattaya, T. Damak, Robust sliding-mode control for the problem of trajectory tracking of

autonomous marines vehicles, 2020 20th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA). Monastir, Tunisia. IEEE, 2020, 144-148, doi: 10.1109/STA50679.2020.9329350.

[11] L. Fan, Y. Zhang, S. Zhang, Dynamic trajectory tracking control of mobile robot, 2018 5th International Conference on Information Science and Control Engineering (ICISCE). Zhengzhou, China. IEEE, 2018, 728-732, doi: 10.1109/ICISCE.2018.00156.

[12] L. Qiao, W. Zhang, Fast trajectory tracking control of underactuated autonomous underwater vehicles, 2018 IEEE 8th International Conference on Underwater System Technology: Theory and Applications (USYS). Wuhan, China. IEEE, 2018, 1-6, doi: 10.1109/USYS.2018.8779000.

[13] X. Shen, W. Shi, Adaptive trajectory tracking control of wheeled mobile robot, 2019 Chinese Control and Decision Conference (CCDC). Nanchang, China. IEEE, 2019, 5161-5165, doi: 10.1109/CCDC.2019.8833019.

[14] W. Xiao, J. Zou, H. Li, K. Xu, Smooth trajectory tracking using longitudinal distance constraint for A 4WS4WD unmanned ground vehicle, 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO). Dali, China. IEEE, 2019, 2105-2110, doi: 10.1109/ROBIO49542.2019.8961670.

[15] N. Wang, Z. Liu, Z. Zheng, M. J. Er, Global exponential trajectory tracking control of underactuated surface vehicles using dynamic surface control approach, 2018 International Conference on Intelligent Autonomous Systems (ICoIAS). Singapore. IEEE, 2018, 221-226, doi: 10.1109/ICoIAS.2018.8494037.

[16] Z. Yan, Z. Yang, J. Zhou, L. Wang, H. Jia, Horizontal trajectory tracking control of AUV using a two-way channel high gain observer, 2018 37th Chinese Control Conference (CCC). Wuhan, China. IEEE, 2018, 418-422.

[17] J.-P. Laumond (Ed.), Robot Motion Planning and Control, Springer Berlin Heidelberg, 1998, **229**, 343, doi: 10.1007/BFb0036069.

[18] P. Morin, C. Samson, Motion control of wheeled mobile robots. Siciliano B, Khatib O, Springer Handbook of Robotics. Berlin, Heidelberg: Springer, 2008: 799-826.10.1007/978-3-540-30301-5_35.

[19] R. F. Gilimyanov, A. V. Pesterev, L. B. Rapoport, Motion control for a wheeled robot following a curvilinear path, *Journal of Computer and Systems Sciences International*, 2008, **47**, 987-994, doi: 10.1134/s1064230708060129.

[20] C. Guo, Z. Sun, Y. Chen, Y. Xie, S. Li, H. Qian, Trajectory tracking of unicycle-type robots with constraints, 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO). Kuala Lumpur, Malaysia. IEEE, 2018, 1700-1705, doi: 10.1109/ROBIO.2018.8665273.

[21] A. Diveev, E. Sofronova, N. Konyrbaev, A. Bexeitova, Stabilization of Movement along an Optimal Trajectory and Its SolutionINTELS'22. Basel Switzerland: MDPI, 2023 33, 12, doi: 10.3390/engproc2023033012.

[22] A. Diveev, E. Sofronova, Universal stabilisation system for control object motion along the optimal trajectory, *Mathematics*, 2023, **11**, 3556, doi: 10.3390/math11163556.

[23] D. Angelis, F. Sofos, T. E. Karakasidis, Artificial intelligence in physical sciences: symbolic regression trends and perspectives, *Archives of Computational Methods in Engineering*, 2023, **30**, 3845-3865, doi: 10.1007/s11831-023-09922-z.

[24] M. Quade, Symbolic regression for identification, prediction, and control of dynamical systems, Dissertation, University of Potsdam, 2018.

[25] A. Diveev, D. Kazaryan, E. Sofronova, Symbolic regression methods for control system synthesis, 22nd Mediterranean Conference on Control and Automation. Palermo, Italy. IEEE, 2014, 587-592, doi: 10.1109/MED.2014.6961436.

[26] A. Diveev, Hybrid evolutionary algorithm for optimal control problem. Arai K, Proceedings of SAI Intelligent Systems Conference. Cham: Springer, 2023: 726-738.10.1007/978-3-031-16078-3_50

[27] L. Davis, Handbook of Genetic Algorithms, Thomson Publishing Group; First Edition, 1991, 385, doi: 10: 0442001738.

[28] J. Kennedy, R. Eberhart, Particle swarm optimization, Proceedings of ICNN'95 - International Conference on Neural Networks. Perth, WA, Australia. IEEE, 1995, 1942-1948, doi: 10.1109/ICNN.1995.488968.

[29] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Advances in Engineering Software*, 2014, **69**, 46-61, doi: 10.1016/j.advengsoft.2013.12.007.

