Application of MATLAB in the design of automatic control systems in the state space

Do Quang Thong

Abstract: **The article analyses the advantages and disadvantages of existing methods for synthesizing automatic control systems in the state space. Based on this, a suitable method is proposed for synthesizing automatic control systems in the MATLAB environment, which has the advantages and eliminates the disadvantages of existing methods; an algorithm for implementing the proposed method in the MATLAB environment is presented. The proposed algorithm makes it possible to significantly reduce the complexity and time of automatic control systems synthesis in the state space. In addition, a method for synthesizing an automatic control system with two integral links in the state space is proposed. The optimal choice of the dominant pole ensures the high quality of the system. This system allows you to track the change in the input signal at a constant rate.**

Keywords: **Synthesis of an automatic control system, state space, Ackermann method, speed tracking system.**

I. INTRODUCTION

Currently, MATLAB is widely used in research (including synthesis) of automatic control systems. For example, Control System Toolbox [1, 2] and Simulink [2] are used in the study of automatic control systems (determination of transfer functions, zeros and poles of the system; construction of logarithmic-amplitude characteristics and logarithmic-phase-frequency characteristics of an open system; construction of the Nyquist curve; construction of a root hodograph; transition characteristics of a closed system)... In addition, MATLAB has the Sisotool Tool [3], the Ackermann, and place commands [4, 5], which help designers synthesise automatic control systems. The author wishes to expand the possibility of using MATLAB in the design of automatic control systems in the state space. Therefore, the article proposes a synthesis method for an automatic control system in the state space and an algorithm for its implementation in MATLAB. The proposed method of system synthesis has advantages and eliminates the disadvantages of existing methods. After the software implementation of the proposed algorithm in the form of a subroutine, a system synthesis command can be formed, similar to the Ackermann, and placed in MATLAB.

In practice, it is sometimes necessary to track the input signal, which changes at a constant rate. To perform such a task, the automatic control system must have two integrators in its structure. The article proposes a synthesis method for such a system. The result is a high-quality system.

II. ADVANTAGES AND DISADVANTAGES OF EXISTING AUTOMATIC CONTROL SYSTEM DESIGNING METHODS IN THE STATE SPACE

Let a continuous object be given:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} \\ \mathbf{y} = \mathbf{Cx} \end{cases}$$

or a discrete object:

$$\begin{cases} \mathbf{x}(i+1) = \mathbf{Ax}(i) + \mathbf{Bu}(i) \\ \mathbf{y}(i) = \mathbf{Cx}(i) \end{cases}$$

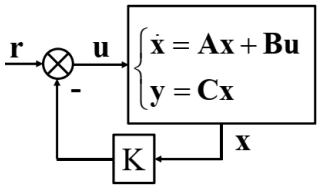

Fig. 1. The block diagram of an automatic control system in the state space

where **A** is the matrix of the object with dimension nxn, **B** is the control matrix with dimension nxm, **C** is the output matrix with dimension rxn, **x** is the column-vector of states of the object with dimension nx1, **u** is the input column-vector with dimension mx1. It is required to define a controller **K** that implements state feedback (Fig. 1) so that the synthesised system has the desired poles (desired eigenvalues of the system matrix):

$$\mathbf{s}_d = \left(s_{d1} \quad s_{d2} \quad \cdots \right.$$

Currently, it is possible to apply the Ackermann, place Comanches [4-8] in MATLAB, Bass-Gura method, direct comparison method [6, 7], controllable canonical form method [8], the Roppennecker method, and the modal method [9] to solve this problem. The matrix of the synthesised system is defined in this way [4-9]:

$$\mathbf{A}_c = \mathbf{A} - \mathbf{BK}. \qquad (1)$$

The general advantage of the above methods is the possibility of using them in the design of continuous and discrete systems. The advantages of the Ackermann method, the place command, the Bass-Gura method, the direct comparison method, the controllable canonical form method, and the Roppennecker method are the possibility of using them to design systems with complete initial poles (eigenvalues of the matrix A) of objects and the use of complete and real desired poles. However, the Ackermann method, Bass-Gura method, and controllable canonical form

method can only be applied in SISO objects. The "place" command can be used to synthesize SISO and MIMO systems, but it cannot be applied with a multiplicity of desired poles exceeding the rank of the B matrix. Roppennecker and modal methods can be applied to design SISO and MIMO systems. The Roppennecker method cannot be applied in two cases: (1) when any pole of the projected object coincides with two (or more) desired poles of the systems; (2) when the multiplicity of the desired poles is greater than two. The modal method can be applied when any pole of the projected object coincides with any number of desired poles. However, this method cannot be applied in the case of an object with complete initial poles and complete desired poles of systems. The method can only move the maximum number of m poles. If it is necessary to move the pole numbers greater than m, multiple syntheses are required. In addition, this method may not be applied in the case of an object with two (or more) identical initial poles.

From the above analysis of the advantages and disadvantages of existing methods, it is suggested to apply two control system design methods sequentially. First, apply the Roppennecker method to design a system with any n distinct real desired poles. Then apply the modal method to design a system with real desired poles. The complexity of the proposed approach is eliminated by using MATLAB in the design of the system. This approach has all the advantages listed above. The disadvantage of this approach is that it can only be applied in cases of real desired poles.

III. THE ROPPENNECKER METHOD

The Roppennecker method is as follows:

For each desired s_{dk} pole, define the vector $\mathbf{e}_k$ as follows:

If s_{dk} is not an eigenvalue of matrix **A**, then $\mathbf{e}_k$ is determined by the formula:

$$\mathbf{e}_k = \left(s_{dk}\mathbf{I} - \mathbf{A}\right)^{-1}\mathbf{B}\mathbf{t}_k \tag{2}$$

where $\mathbf{t}_k$ is arbitrarily chosen so that the vectors $\mathbf{e}_k$ are linearly independent; vectors $\mathbf{t}_k$ and $\mathbf{e}_k$ are column vectors.

If s_{dk} is an eigenvalue of matrix **A**, then the controller should not move this pole, select $\mathbf{t}_k=0$ and $\mathbf{e}_k$ is the right eigenvector of matrix **A**:

$$\left(s_{dk}\mathbf{I} - \mathbf{A}\right)\mathbf{e}_k = \mathbf{0} \tag{3}$$

Define the controller **K** using the formula:

$$\mathbf{K} = -\left(\mathbf{t}_1\ \mathbf{t}_2\ \ldots\ \mathbf{t}_n\right)\left(\mathbf{e}_1\ \mathbf{e}_2\ \ldots\mathbf{e}_n\right)^{-1} \tag{4}$$

Thus, the difficulty of implementing the Roppennecker method on a computer lies in finding one solution other than zero ($\mathbf{e}_k\neq 0$) of the system of equations (3). The determinant of the matrix and the right part (3) are zero; therefore, it is impossible to apply the Kramer method to solve systems of equations (3). It is proposed to determine the solution $\mathbf{e}_k$ of the system of equations (3) by gradually reducing its dimension. It is assumed that the system of n equations (3) has the form:

$$\begin{cases} a_{11}e_1 + a_{12}e_2 + \ldots + a_{1n}e_n = 0 \\ a_{21}e_1 + a_{22}e_2 + \ldots + a_{2n}e_n = 0 \\ \ldots \\ a_{n1}e_1 + a_{n2}e_2 + \ldots + a_{nn}e_n = 0 \end{cases} \tag{5}$$

We find the first non-zero coefficient of the system of equations (5), starting from the coefficient a_{nn} (we can start from the coefficient a_{11}), and then the element e_j is defined in this way:

$$e_j = -\frac{a_{i1}}{a_{ij}}e_1 - \frac{a_{i2}}{a_{ij}}e_2 - \ldots - \frac{a_{in}}{a_{ij}}e_n \tag{6}$$

On the right side of equality (6) has (n-1) components. Substituting (6) into (5), we obtain a system of (n-1) equations:

$$\begin{cases} (a_{11} - a_{1j}\frac{a_{i1}}{a_{ij}})e_1 + (a_{12} - a_{1j}\frac{a_{i2}}{a_{ij}})e_2 + \ldots + (a_{1n} - a_{1j}\frac{a_{in}}{a_{ij}})e_n = 0 \\ (a_{21} - a_{2j}\frac{a_{i1}}{a_{ij}})e_1 + (a_{22} - a_{2j}\frac{a_{i2}}{a_{nj}})e_{2k} + \ldots + (a_{2n} - a_{2j}\frac{a_{in}}{a_{ij}})e_n = 0 \\ \ldots \\ (a_{n1} - a_{nj}\frac{a_{i1}}{a_{ij}})e_1 + (a_{n2} - a_{nj}\frac{a_{i2}}{a_{ij}})e_2 + \ldots + (a_{nn} - a_{nj}\frac{a_{in}}{a_{ij}})e_n = 0 \end{cases} \tag{7}$$

This operation is repeated (n-2 times) until a system of two equations is obtained:

$$\begin{cases} p_{xy}e_x + p_{xk}e_y = 0 \\ p_{zy}e_x + p_{zk}e_y = 0 \end{cases} \tag{8}$$

If at least one of the coefficients of the systems of equations (8) differs from zero, then the above operation can be performed to find e_x and e_y. In practice, all coefficients of the system of equations (8) may be zero. Solving systems of equations (8) on a computer is not difficult. It is necessary to consider all possible cases of coefficients of systems of equations (8) (other than zero or equal to zero). For example, if all the coefficients of the system of equations (8) are zero, then it is necessary to select any different real values of the elements e_x and e_y. The program for solving the system of equations (8) is expediently executed in the form of the hai_nghiem subroutine:

$$function\ \mathbf{a3} = hai\_nghiem(saiso\_pt, chiso, \mathbf{pp}) \tag{9}$$

where **pp** is the matrix of the left side of the equations (8); saiso_pt is the accuracy of comparing two fractional numbers in a computer; chiso-a variable that provides a choice of n linearly independent vectors $\mathbf{t}_i$.

From (2) and (4), we note that the value of the controller **K** depends on the value of the vector **t**, chosen arbitrarily. Therefore, there is an infinite number of controller **K**.

The algorithm for implementing the Roppennecker method in MATLAB:

1. Determine the dimensions n and m of the matrix **B** with the “size” command;

2. Check the controllability of the designed system according to [6-9];

3. Determine the maximum multiplicity of the desired poles q_s; determine the maximum number of desired poles q_{1s} coinciding with the poles of the initial object;

4. If $q_{1s}<2$ and $q_s\leq 2$, then perform the following actions for each pole of the object:

4.1. If this pole does not coincide with the desired pole s_{dk}, then select $\mathbf{t}_k$ arbitrarily, determine $\mathbf{e}_k$ using the formula (2);

4.2. If this pole coincides with the desired pole s_{dk}, then $\mathbf{e}_k$ is determined in this way:

4.2.1. Define the matrix (s_{dk}**I-A**)

4.2.2. If n=2 then apply subroutine (9) to find $\mathbf{e}_k$;

4.2.3. If n>2 then perform the following actions:

Direct move:

4.2.3.1. Write the matrix (s_{dk}**I-A**) to the first cell of the array f; Form a matrix of zero $\mathbf{A}_1=(0\ 0\ 0\ )^T$ and write it to the first cell of the array f_1; Form two vectors of n elements $\mathbf{e}_k=(0\ 0\ \dots\ 0)$ and $\mathbf{e}_2=(1\ 2\ \dots\ n)$. The elements of vector $\mathbf{e}_2$ are the number of elements of vector $\mathbf{e}_k$. Scan the variable n_1 from 1 to (n-2). For each value of n_1, perform the following actions:

4.2.3.2. Assign the value of cell No. 1 of the array f to the $\mathbf{A}_2$ matrix;

Determine the size n_2 of the matrix $\mathbf{A}_2$ with the "size" command;

Starting from the element in the last row and the last column (or from the element in the first row and the first column), find the first non-zero element of the $\mathbf{A}_2$ matrix and determine the corresponding element of the $\mathbf{e}_k$ vector by (6). It is assumed that the first non-zero element of the $\mathbf{A}_2$ matrix is located on row number i and column number j. This means that element No. j of the vector $\mathbf{e}_k$ has already been defined.

Determine the coefficients of the system of equations (7) and assign them to the elements of the $\mathbf{A}_3$ matrix;

Write the $\mathbf{A}_3$ matrix to cell No. (n_1+1) of the array f;

Assign the value j to the first element of the first row of matrix $\mathbf{A}_1$; elements from No. 2 to No. (n_2-1) of the first row are zero;

Assign the coefficient values of the right side (6) to the second row of matrix $\mathbf{A}_1$; Delete element No. j of vector $\mathbf{e}_2$;

Assign the values of vector $\mathbf{e}_2$ to the third row of matrix $\mathbf{A}_1$; Thus, matrix $\mathbf{A}_1$ contains the information of element No. j of the vector $\mathbf{e}_k$. Write matrix $\mathbf{A}_1$ to cell number (n_1+1) of array f_1; Delete matrices $\mathbf{A}_1$, $\mathbf{A}_2$, and $\mathbf{A}_3$.

After operating subparagraph 4.2.3.2 (n-2) times, we obtain a system of two equations (8). Arrays f and f_1 have (n-1) cells. The cell number (n-1) of the array f is the coefficients of the system of equations (8). The two remaining elements of the vector $\mathbf{e}_2$ are the number of elements e_x and e_y. Here we finish scanning n_1 from 1 to (n-2).

All operations 4.2.2-4.2.3.2 are expediently performed in the form of a subroutine:

$$\mathbf{e}k = tinh\_vector\_ek(\text{saiso\_pt},\text{chiso},\text{n},\mathbf{sdki\_A}) \quad (10)$$

where **sdki_A**-matric [s_{dk}**I-A**].

Reverse move:

4.2.3.3. Determine the values of the two elements e_x and e_y of the vector $\mathbf{e}_k$ in (8) by applying subroutine hai_nghiem (9):

$$\mathbf{a}4 = hai\_nghiem(saiso\_pt, chiso, f\{n-1\})$$

Assign the values of two elements e_x and e_y to the corresponding elements of vector $\mathbf{e}_k$ based on elements of the vector $\mathbf{e}_2$.

4.2.3.4. Read the cells of the f_1 array from cell No. (n-1) to cell No. 2. For each cell, we get the $\mathbf{A}_1$ matrix. The first element of the matrix $\mathbf{A}_1$ is the number of the desired element of vector $\mathbf{e}_k$; each element of the second row of matrix $\mathbf{A}_1$ is the value of the coefficients of the right part (6); each element of the third row is the number of certain elements of vector $\mathbf{e}_k$ in the right part (6). Based on the second and third rows of matrix $\mathbf{A}_1$, we determine the value of the corresponding element in the vector $\mathbf{e}_k$. Assign the found value to the corresponding element of vector $\mathbf{e}_k$.

After finding the values of all the elements of all vectors $\mathbf{e}_k$, we define the controller **K** by (4).

To obtain independent vectors $\mathbf{e}_k$, when forming the vector **t** in (2) and the chiso variable in (9), it is advisable to use the "normrnd(x,y)" command.

IV. THE MODAL METHOD

The modal method is as follows:

Determine the m left eigenvectors $\mathbf{b}_1^T, \mathbf{b}_2^T, ..., \mathbf{b}_m^T$ of the matrix **A** by the formula:

$$\mathbf{b}_i^T(\mathbf{g}_i\mathbf{I}-\mathbf{A}) = \mathbf{0}^T \quad (11)$$

Define the controller **K** by the formula:

$$\mathbf{K} = -\mathbf{T}_m(\mathbf{s}_m - \mathbf{g}_m)\mathbf{M}_m \quad (12)$$

where the matrices $\mathbf{T}_m$, $\mathbf{M}_m$, $\mathbf{s}_m$, and $\mathbf{g}_m$ are defined as follows:

$$\mathbf{T}_m = \begin{pmatrix} \mathbf{b}_1^T\mathbf{B} \\ \vdots \\ \mathbf{b}_m^T\mathbf{B} \end{pmatrix}^{-1} \quad (13)$$

$$\mathbf{M}_m = \begin{pmatrix} \mathbf{b}_1^T & \cdots & \end{pmatrix}^T; \quad (14)$$

$$\mathbf{s}_m = \begin{pmatrix} s_1 & 0 & \cdots & \\ 0 & s_2 & \cdots & \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \end{pmatrix} \quad (15)$$

$$\mathbf{g}_m = \begin{pmatrix} g_1 & 0 & \cdots & \\ 0 & g_2 & \cdots & \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \end{pmatrix} \quad (16)$$

where $s_1, s_2, ..., s_m$ are the desired poles of the system; $g_1, g_2, ..., g_m$ are the eigenvalues of the matrix **A**. To apply subroutines (10) to determine the vector $\mathbf{b}_i$, transform (11) as:

$$(\mathbf{g}_i\mathbf{I}-\mathbf{A})^T\mathbf{b}_i = \mathbf{0} \quad (17)$$

The algorithm for implementing the modal method in MATLAB:

1. Determine the dimension of matrix **B**;

2. Check the controllability of the designed system [6-9];

3. Determine the vector of eigenvalues $\mathbf{tr\_rA}_1$ of the matrix **A**; assign vector $\mathbf{s}_d$ to vector $\mathbf{s}_{d1}$.

4. Move the elements of the vectors $\mathbf{tr\_rA}_1$ so that the elements other than the elements of the vector $\mathbf{s}_{d1}$ take the first places:

Select all the elements of the vector $\mathbf{tr\_rA}_1$ that match the elements of the vector $\mathbf{s}_{d1}$ to form the vector $\mathbf{tr\_rA}_2$. Simultaneously exclude these elements from the vector $\mathbf{tr\_rA}_1$ and the vector $\mathbf{s}_{d1}$. If the vector $\mathbf{tr\_rA}_1$ is not empty,

then form the vector **tr_r** by connecting the vectors $\mathbf{tr\_rA_1}$ and $\mathbf{tr\_rA_2}$; and if the vector $\mathbf{tr\_rA_1}$ is empty, then go to point 10;

5. For each of the first m elements of the vector **tr_r**, determine the vector $\mathbf{b_i}$ by (17) using the subroutine (10). To increase the accuracy of the calculation, it is advisable to multiply $[\mathbf{g_iI-A}]^T$ by a large factor (for example, 10^{50}), then divide the resulting elements of the vector $\mathbf{b}_i$ by this factor. The accuracy of the calculation significantly depends on the value of this multiplier and saiso_pt.

6. If the number of elements of the vector $\mathbf{s}_{d1}$ is less than m, then it is necessary to supplement it with the corresponding elements of the vector **tr_r**;

7. Determine the controller $\mathbf{K}_i$ by (10)-(16) using $\mathbf{s}_{d1}$ and **tr_r** and write it to cell No. i of the array p. The value of the controller $\mathbf{K_i}$ depends on the method of selecting the variable chiso in subroutine (10). Thus, we can get an infinite number of the controller $\mathbf{K}_i$.

8. Define a new matrix **A** using the formula (1):

$$\mathbf{A} = \mathbf{A} - \mathbf{BK}_i.$$

9. Repeat steps (3)-(8) until the system has all the desired poles (vector $\mathbf{tr\_rA_1}$ is empty).

10. Read all cells of the array p and determine the controller **K** by the formula:

$$\mathbf{K} = \sum_i \mathbf{K}_i.$$

V. COMBINATION OF THE ROPPENNECKER AND THE MODAL METHODS

It is proposed to first apply the Roppenecker method to synthesize a system with any different temporal real poles, for example

$$\mathbf{s}_{dr} = (-0{,}1 \; -0{,}2 \; ... -0{,}1n).$$

As a result, the controller $\mathbf{K}_r$ gets knocked out. Define a new matrix of the system:

$$\mathbf{A}_{n1} = \mathbf{A} - \mathbf{BK}_r.$$

From matrixes $\mathbf{A}_{n1}$, **B**, $\mathbf{s}_d$, synthesize the controller $\mathbf{K}_m$ using the modal method;

The desired controller **K** is defined as follows:

$$\mathbf{K} = (\mathbf{K}_r + \mathbf{K}_m).$$

Determine the final matrix of the system:

$$\mathbf{A}_n = \mathbf{A} - \mathbf{BK}_r - \mathbf{BK}_m = \mathbf{A} - \mathbf{BK}.$$

All the above operations are expediently performed in the form of a subroutine:

$$\mathbf{K} = \text{rop\_mod}(\mathbf{A},\mathbf{B},\mathbf{s}_d).$$

Thus, we can use the command rop_mod as the command place in Matlab.

Example: Let the object have matrices **A** and **B**:

$$\mathbf{A} = \begin{pmatrix} 0{,}2 & 0 & 1 & 0{,}5 & 0{,}6 \\ 0{,}1 & 0 & 0{,}1 & 0{,}8 & 0{,}7 \\ 1 & 0 & 3 & 4 & 2 \\ 1 & 0{,}3 & 0{,}7 & 0{,}6 & 1 \\ 0{,}5 & 0{,}4 & 0{,}2 & 0{,}1 & 0{,}8 \end{pmatrix};$$

$$\mathbf{B} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

1. It is assumed that this object belongs to continuous ones; it is required to define a controller **K** that implements state feedback so that the system has the desired poles:

$$\mathbf{s}_d = (-0{,}2601 \quad -0{,}2601 \quad -0{,}2601 \quad -0{,}2601 \quad -0{,}2601).$$

2. It is assumed that this object belongs to discrete ones with a sampling period T_0=0,1 s; it is necessary to determine a controller **K** that implements state feedback so that the system has the desired poles:

$$\mathbf{s}_d = (0{,}6 \quad 0{,}6 \quad 0{,}6 \quad 0{,}7 \quad 0{,}7).$$

Solution: 1. The matrix of system **A** has its eigenvalues:

$$\mathbf{tr\_rA} = (4{,}5939 \quad 0{,}7291 \quad -0{,}2315 + 0{,}5148j \quad -0{,}2315 - 0{,}5148j \quad -0{,}2601).$$

Thus, s_{dk} is an eigenvalue of matrix **A**. First, apply the Roppennecker method to determine the $\mathbf{K}_r$ controller, which implements state feedback so that the system has intermediate desired poles:

$$\mathbf{s}_{dr} = (-0{,}1 \quad -0{,}2 \quad -0{,}3 \quad -0{,}4 \quad -0{,}5)$$

The intermediate matrix of the system is defined in this way:

$$\mathbf{A}_{n1} = \mathbf{A} - \mathbf{BK}_r$$

Then, by applying the modal method in defining the $\mathbf{K}_m$ controller for the $\mathbf{A}_{n1}$ object, implementing state feedback so that the system has the desired poles:

$$\mathbf{s}_d = (-0{,}2601 \quad -0{,}2601 \quad -0{,}2601 \quad -0{,}2601 \quad -0{,}2601)$$

This is how we use the normrnd(x,y) command to determine the elements of the vector $\mathbf{t}_i$ in (2) and $\mathbf{e}_k$ in (3), so each time we call the rop_mod subroutine, we get different values of the controller **K**. One options of the controller **K** is:

$$\mathbf{K} = \begin{pmatrix} 0{,}51411 & -0{,}029788 & 0{,}528262 & 0{,}395402 & 0{,}15706 \\ 0{,}714609 & 0{,}208766 & 1{,}505221 & 1{,}8753 & 1{,}4017 \end{pmatrix}.$$

Another values of the controller **K** is:

$$\mathbf{K} = \begin{pmatrix} 0{,}51407 & -0{,}029603 & 0{,}521628 & 0{,}39065 & 0{,}152782 \\ 0{,}714621 & 0{,}208708 & 1{,}507275 & 1{,}876771 & 1{,}403025 \end{pmatrix}.$$

2. One values of the controller **K** is:

$$\mathbf{K} = \begin{pmatrix} -3{,}560647 & -2{,}619262 & 1{,}705573 & 3{,}627515 & 1{,}494992 \\ 2{,}909701 & 1{,}752375 & 0{,}176024 & -0{,}888658 & 0{,}293391 \end{pmatrix}.$$

Another values of the controller **K** is:

$$\mathbf{K} = \begin{pmatrix} -5{,}0340777 & -4{,}086992 & 2{,}276506 & 5{,}691405 & 2{,}235311 \\ 3{,}365817 & 2{,}206726 & -0{,}000714 & -1{,}527557 & 0{,}064218 \end{pmatrix}.$$

VI. TRACKING AN INPUT SIGNAL CHANGING AT A CONSTANT RATE

The designed system, which does not have two integrators in its composition, cannot monitor the input signal, which changes at a constant rate. In this case, it is necessary to supplement the system with two integrators. The system's structure is detailed in Fig. 2. Now the system has the order (n+2).

If we define the new state as:

$$\left(\quad 2 \quad \cdots \quad \quad c_{n+1} \quad x_{n+2} \right)^T,$$

then the system has a new mathematical model:

$$\begin{pmatrix} \dot{} & & | & / \\ \vdots & & | & \mathbf{Ax}+\mathbf{BU} \\ \dot{} & & = | & 0\ldots0\ 1 \\ \vdots & & | & r-\mathbf{Cx} \\ \dot{} & \cdots & | & \end{pmatrix}$$

$$= \begin{pmatrix} a_{11} & a_{12} & \cdots & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & & a_{nn} & 0 & 0 \\ 0 & 0 & \cdots & & 1 & \\ -c_1 & -c_2 & \cdots & & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ \\ \vdots \\ x_{n+1} \\ x_{n+2} \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ \vdots \\ \\ 0 \\ 1 \end{pmatrix} r$$

When designing a system, it is necessary to determine the

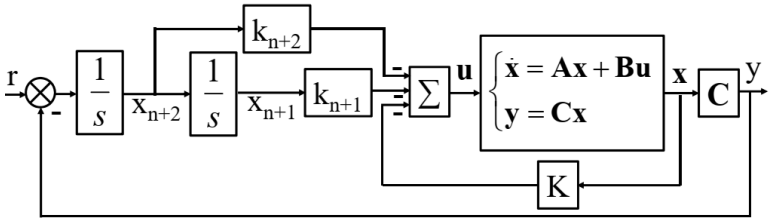

Fig. 2. The system with two integrators

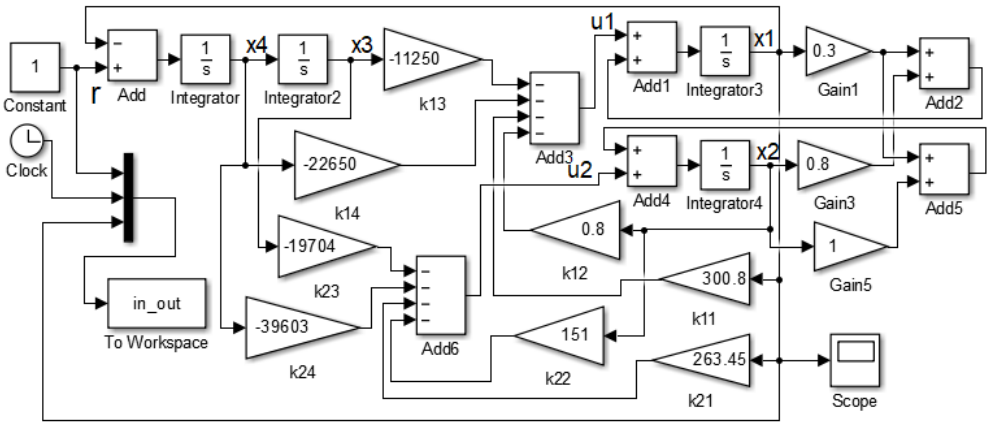

Fig. 3. The block diagram of the synthesised system in the Simulink

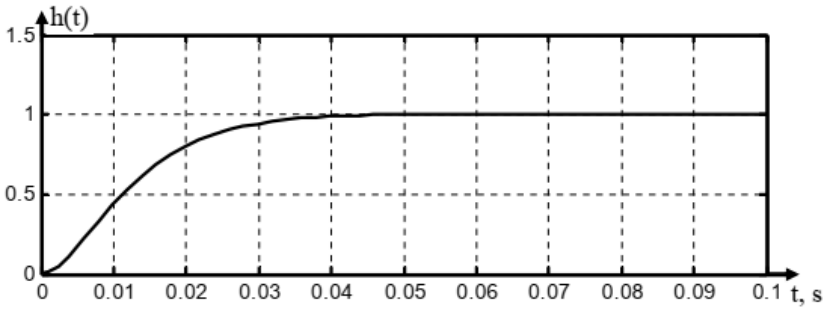

Fig. 4. The transitional characteristics of the synthesised system

desired poles that determine its quality in the transient and steady-state modes. Our system has two integrators, so the steady-state velocity error is zero. The quality of the system in the transition process is mainly determined by the value of the dominant poles [3]. Therefore, it is supposed to apply a vector of desired poles in the form:

$$\mathbf{s_d} = \left(s_1 \quad s_2 \quad s_2 \quad \cdots \right.$$

where s_2-dominant pole.

Let's define these dominant poles using the parametric optimization method. Let's choose the minimum transition time of the system as the target function. The algorithm for determining the desired poles will include the following steps:

Scan pole s_1 from s_{1b} to s_{1f} with a scan step of ds_1. For each value s, scan poles s_2 in from s_{2b} to s_{2f} with a scan step of ds_2.

For each pair of (s_1, s_2), apply the higher proposed methodology to determine the value of controller **K** and construct a transient characteristic of the system.

If the maximum value of the transition characteristic is less than 1,25, determine the transition time of the system.

Determine the pair of poles (s_1, s_2) and an appropriate value of controller that provides the minimum transition time.

Example: An object with a mathematical model is specified:

$$\begin{cases} \dot{\mathbf{x}} = \begin{pmatrix} 0,3 & 0,8 \\ 0,3 & 1 \end{pmatrix}\mathbf{x} + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\mathbf{u} \\ y = \begin{pmatrix} 1 & 0 \end{pmatrix}\mathbf{x} \end{cases}$$

It is required to synthesize a system tracking the input signal, which changes at a constant rate.

Solution:

After entering two integrators into the system, the matrices A, B, and C have the form:

$$A = \begin{pmatrix} 0,3 & 0,8 & 0 & 0 \\ 0,3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \end{pmatrix}; B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}; C = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix}.$$

After applying the previously proposed system synthesis technique, one value of the controller turned out to be:

$$\mathbf{K} = \begin{pmatrix} 300,8 & 0,8 & -11250 & -22650 \\ 263,45 & 151 & -19704 & -39603 \end{pmatrix}.$$

The block diagram of the synthesized system in the Simulink environment is shown in Fig. 3. Its transient characteristic is shown in Fig. 4.

VII. CONCLUSION

Thus, the consistent application of the Roppennecker method and the modal method makes it possible to design an automatic control system in the state space with any multiplicity of real desired poles. To ensure the required quality of the system, it is necessary to select the appropriate set of desired poles. For one set of desired poles, you can get an infinite number of K controllers. This approach can be applied to the design of a system of any dimension. The method can be applied to the synthesis of continuous and discrete systems.

The proposed method of synthesizing an automatic control system with two integrators makes it possible to build a high-quality tracking system.

DECLARATIONS

Conflict of Interest

The author declare that there is no competing financial interest or personal relationship that could have appeared to influence the work reported in this paper.

Authors' Contributions

The author is responsible for the entire work, including the conceptualization, methodology, data collection, simulation, analysis, and writing of the manuscript.

Funding

Not Applicable.

REFERENCES

[1] Metvedev V.S., Pochemkin V.G., Control System Toolbox, M.: Ed. DIALOG MEPHI, P. 20-102 1999.

[2] V. P. Dyakonov. MATLAB 7.*/R2006/R2007, Self-instruction, DMK Publishing House, Moscow, P. 20-200, 2008

[3] Nguyen Thi Phuong Ha, Huynh Thai Hoang, The theory of automatic control, Ho Chi Minh City State University, P. 187-190, 2005.

[4] Gene F. Franklin, J. Dovit Powell, Abbas Emami-Naeini, Feedback Control of Dynamic Systems, Eighth Edition, Stanford University, P. 508-516, 2020

[5] Katsuhiko Ogata, Modern control engineering, Fifth edition, Prentice Hall. P. 722-800, 2010.

[6] Robert L, William II, Douglas A, Lawrence, Linear State-Space Control Systems, , Ohio Univercity, P. 271-278, Copyright© 2007 John Wiley & Sons.

[7] Arie Nakhmani, Modern control State-Space Analysis and Design Methods, Univercity of Alabama at Birmingham, P.35-40, Copyright© 2020 by McGraw Hill

[8] Roland S, Burns. Butterworth Heinemann, Advanced Control Engineering, P.232-254, 2001.

[9] Nguyen Doan Phuoc, Theory of linear automatic control systems, s, Hanoi. Scientific technique, P. 306-319, 1984.

Do Quang Thong (email: thongdq@lqdtu.edu.vn)

Year of birth: 1966.
Place of birth: Province: Hanam; Country: Vietnam.
Academic title: Candidate of Technical Sciences (from BSTU named D.Ph. Uctinov, 2005), country: Russia Fderation.
Specialties: System analysis, Control, and Information Processing.
Associate Professor (2022), Le Qui Don Technical University, Hanoi, Vietnam.
Place of work: Le Qui Don Technical University, Hanoi, Vietnam.
Position: Teacher.

Valuable published articles:
1. (Scopus). Synthesis of a missile homing system taking into account the dynamics of measuring elements (2019), Mechatronics, Automation, Control/ ISSN 1684-6427 (print), ISSN 2619-1253 (online).
2. (Scopus). Synthesis of High-Precision Missile Homing System Using Proportional Guidance Method (2020), Mechatronics, Automation, Control/ ISSN 1684-6427 (print), ISSN 2619-1253 (online).
3. (Scopus). Synthesis of the Anti-Ship Missile Homing System while Ensuring an Acceptable Oscillation Index of the Stabilization System (2022), Russian Aeronautics ISSN 1068-7998.