Automation and Remote Control, Vol. 66, No. 4, 2005, pp. 664–671. Translated from Avtomatika i Telemekhanika, No. 4, 2005, pp. 176–184.

AUTOMATIC CONTROL SYSTEMS

Structure of the Software for Computer-aided Logical Design of Automatic Control

A. G. Aleksandrov[*], R. V. Isakov[**], and L. S. Mikhailova[*]

[*] *Trapeznikov Institute of Control Sciences, Russian Academy of Sciences, Moscow, Russia*
[**] *Polytechnic Institute, Elektrostal', Russia*

Received March 25, 2004

Abstract—Requirements on the software for computer-aided logical design of automatic control were formulated and assumed as a basis of the software concept. According to this concept, the software for the computer-aided logical design of automatic control should have two levels: the environment of the control system designer and the (researcher) environment for development and updating of the first level. Realization of the concept was illustrated by the GAMMA-2RS system.

1. INTRODUCTION

Logical design of control is one of the most important stages of designing [1] any automatic control system. The software for automation of this stage was developed over the four recent decades. By the mid-1980's several dozens of application packages had been developed for analysis of the automatic control systems, identification, controller design, and so on. In particular, the domestic packages DISPAS [2], SAPRAS [3], RADIUS-2 [4] and GAMMA [5–8] as well as their foreign counterparts such as IDPAC, CYNPAC, KEDDS, and MATLAB deserve mentioning. The former are described in the reference book [1], and the latter, in [9].

Among the foreign systems for logical design of automation, MATLAB [10, 11] came to the forefront over the last decade owing its efficient high-level programming language which enables fast design of the software for the methods of the automatic control theory. The software of the traditional methods of the automatic control theory that was incorporated in the aforementioned foreign packages has been recreated in MATLAB. Additionally, MATLAB includes the software for newer methods such as the H^∞ suboptimal control, μ-analysis and design, robust control, and so on.

The designers of physical automatic control systems wishing to use MATLAB for logical design meet with some difficulties:

(a) they should have an intimate knowledge of the control theory in order to choose—depending on the problem at hand—the uncertain parameters of the method used such as the parameters of the quadratic functional of the LQ and H^∞ optimization providing the desired accuracy of control, form and parameters of the identification test signals, and so on;

(b) in order to unite the chosen MATLAB programs (functions), they need to know the MATLAB programming language;

(c) they must know how to design the user interface and generate the protocol of results.

The domestic GAMMA system was developed along with MATLAB. Its first versions GAMMA-1 and GAMMA-2 [5], which comprised the software for design and analysis of the MIMO systems featuring the given accuracy and performance, were oriented to the computers like M-220. GAMMA-1M [6] is their update oriented to the ES computers. The following version, GAMMA-1PC [7], which was intended for the IBM PC-compatible computers, was augmented by the software for the

newer design methods such as the H^∞ suboptimal control and provided with an improved user interface. The present paper formulates a structural concept of the software for logical design of automatic control and describes its implementation by the GAMMA-2PC system, which allows one to escape the difficulties encountered in MATLAB.

2. SOFTWARE STRUCTURE

We formulate the requirements on the software for computer-aided design (design facilities).

(a) Their user needs not to participate in their design.

(b) The design facilities should enable a "natural" problem description both in form and essence.

For the design facilities of the control algorithms, the "natural form" of problem description implies availability of a user interface allowing one, in particular, to input the model of the controlled plant either in a customary form of arbitrary differential equations or as the transfer matrix. "Natural problem description in essence" implies that the design facilities enable design if the aim of control is described in terms of the practical engineering factors such as permissible stable errors, time of control, or stability margins, whereas the external disturbances and noise are uncommon functions of which often only their boundaries are known.

(c) The users' problems must be solved automatically, their participation being confined to the possibility of aborting operation if the displayed or protocolled intermediate data are nonsatisfactory.

It follows from these requirements that the software must have a *two-level structure*: the first level is represented by the facilities (environment) of the designer of the automatic control system, the second level, by the tools (environment) for design and updating of the first level. To specify the software structure, we introduce some definitions. Along with an informal description [12], many notions of the automatic control theory have an *operational definition* (*description*) in the form of a sequence of computational operations. The operational definition needs not be unique. For example, stability of the stationary linear system may be determined operationally either by solving the Lyapunov equation or by calculating the eigenvalues of the matrix describing the system model in the state space.

By the *elementary design operation* is meant the operational definition of an indivisible (minimal) meaningful fragment of the automatic control theory. For example, elementary design operations are represented by stability, controllability, analytical design of controllers (LQ-optimization), and H^∞ suboptimal control.

By the *module* is meant a computer program realizing an elementary design operation.

Directive is a program consisting of three parts: (a) modules, (b) facilities for generating the user interface, and (c) facilities for output of the intermediate and final results (protocol). Each directive serves to solve a certain class of problems involved in the logical design of control.

Class of problems is characterized [1] by models of three kinds: model of control aims (indices of accuracy and performance), model of the plant and controller, and environment model (external disturbances and measurement errors).

The user interface is used to describe a particular problem from a certain class. *The environment of the designer of the automatic control system* (*user environment*) consists of directives. The users (designers) choose a directive, input through the interface a description of a particular problem, and then analyze the protocol of results. The problem is handled without their participation. The user environment is designed by the researcher who, being well versed in the automatic control theory and one of the programming languages, is able to design modules and directives.

Researcher environment is represented by the software facilities for design of new directives and generation of the user environment.

3. REALIZATION OF THE TWO-LEVEL SOFTWARE STRUCTURE BASED ON THE GAMMA-2PC SYSTEM

3.1. User Environment

According to the requirements on the software, the user environment is a collection of directives. The GAMMA-2PC directives enable solution of a wide range of the control theory problems.

The system has directives of five groups:

(a) controller design (directives: design of control satisfying the requirements on control precision, H^∞ optimal control, and so on);

(b) identification (directives: finite-frequency identification, least squares methods, and so on);

(c) adaptive control (directives: adaptive frequency control, adaptive PID controller);

(d) transformation of kinds and forms of models;

(e) system analysis.

The user chooses a necessary directive from their list. Then a form for input of the source data (Fig. 1) is displayed. The user inputs

(a) the plant model (model of the controlled process) or the results of its testing by the system-generated actions;

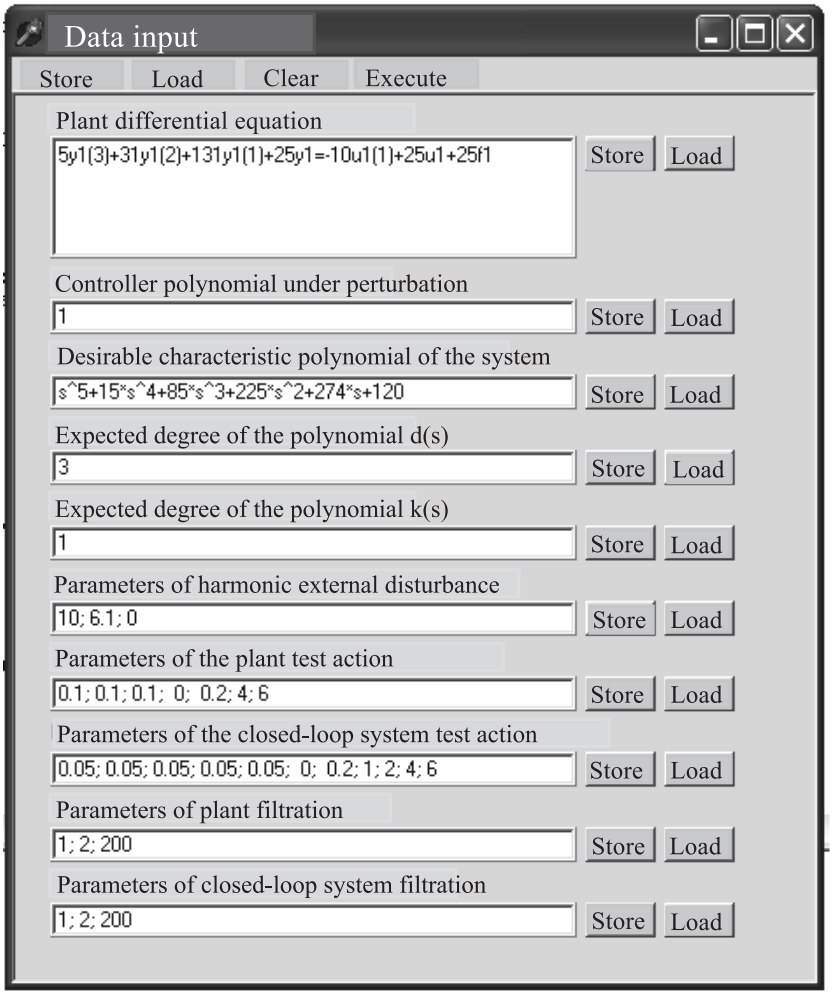

Fig. 1. Form for input of the source data of directive 311 "Modal adaptive frequency control."

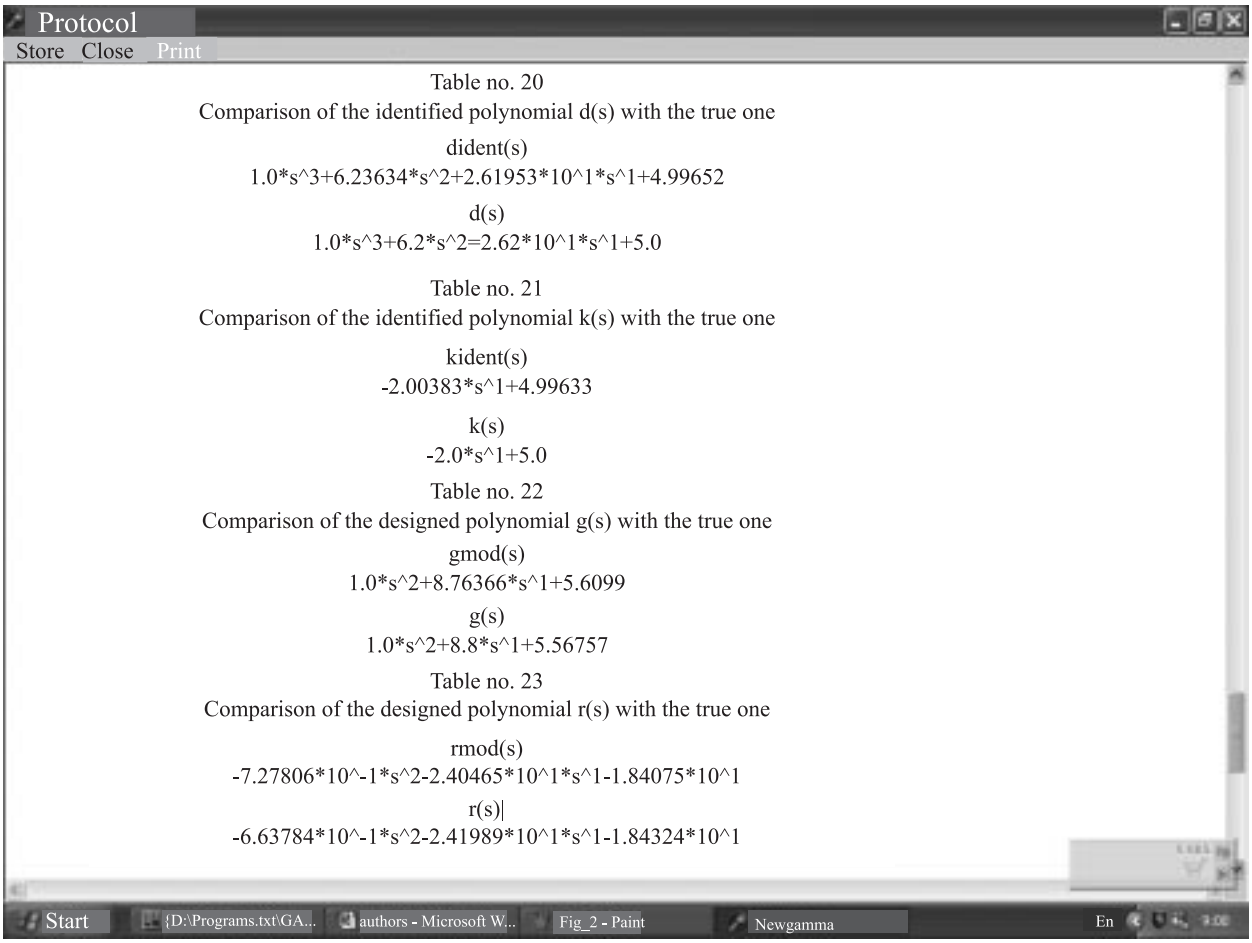

Fig. 2. Directive 311 "Modal frequency adaptive control."

(b) depending on the class of the problem, specifications of the automatic control system such as permissible control errors, control time, and so on.

The system maintains an archive of models (storage of the source data between the work sessions).

The system then operates automatically (without user participation) and produces algorithms for the automatic control system. The intermediate and final results are displayed in the protocol window (Fig. 2) and can be stored by the user.

3.2. Researcher Environment

The researcher environment is a toolkit for designing new directives. It consists of a module library, text and graphic editors (block diagram editor).

The *module library* is a collection of all GAMMA-2PC modules. The modules are executable files in any programming language. The only requirement on them lies in observing the system data exchange protocol. Each module has a corresponding graphic representation in the window of the module library and a description that allows the system to handle this module. Figure 3 depicts a window of the module library with the description of the "Fourier filter" module.

The module library is open: the designer not only can use the modules available, but also may add its own modules. This means that the corresponding executable file will be added to the archive, and a reference to it will be added to the module library.

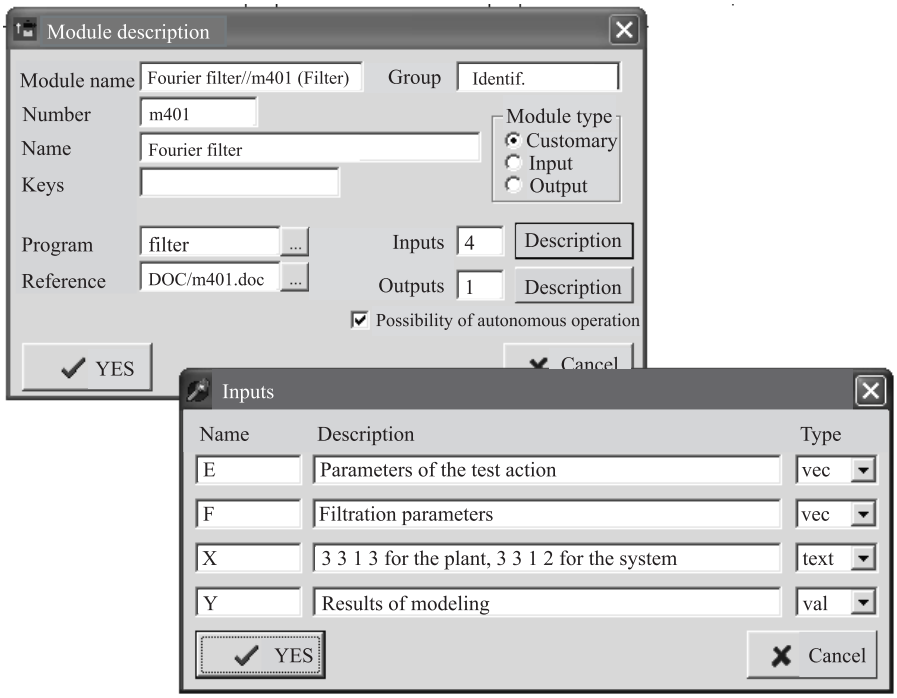

Fig. 3. Window of the module library with a description of the "Fourier Filter" module.

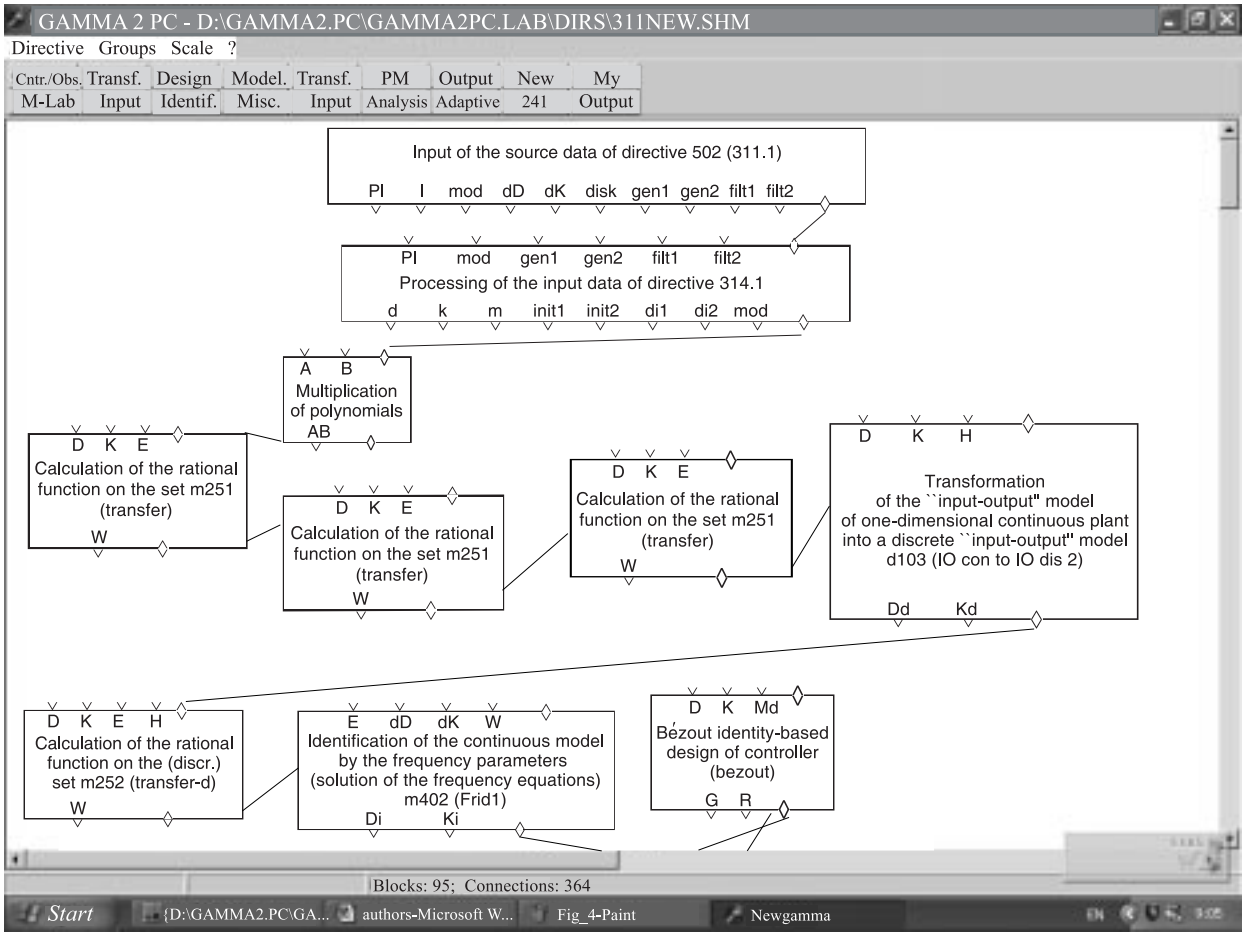

Fig. 4. Block diagram of directive 311 "Modal frequency adaptive control."

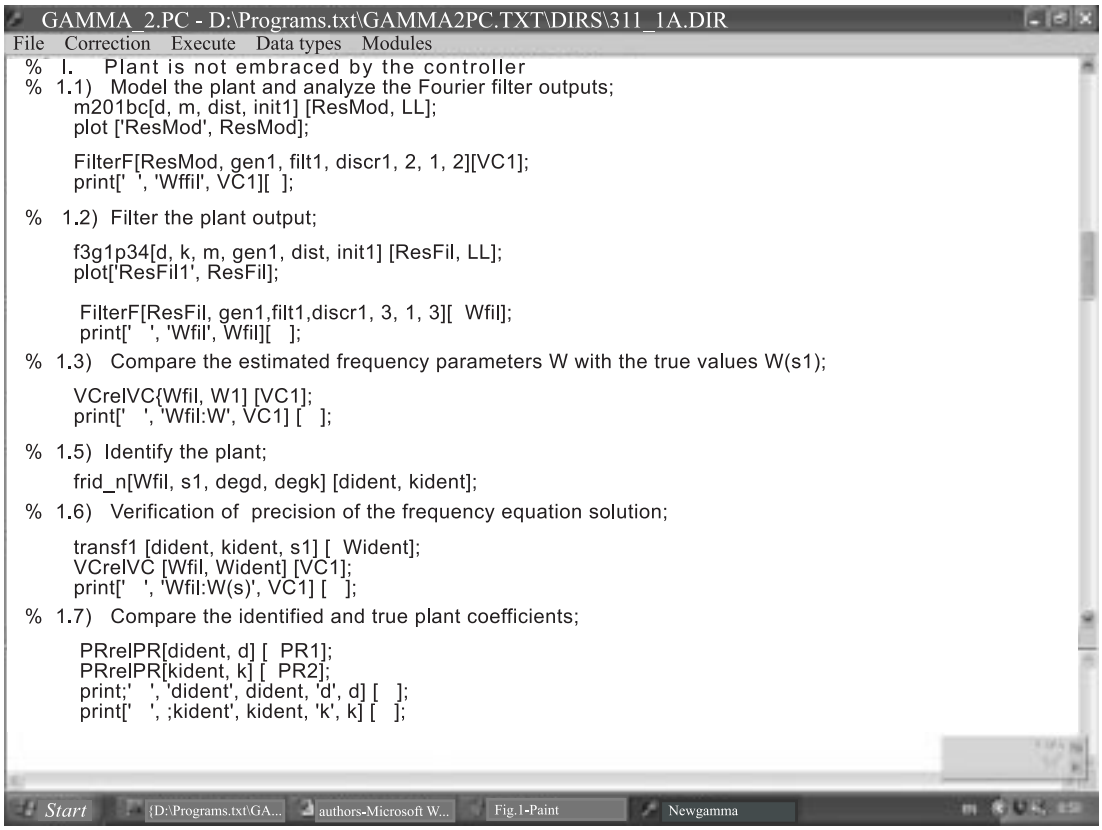

Fig. 5. Text of directive 311 in GAMMA-1.

The system has two alternative means to create new directives. The first is based on describing the control-theoretical methods by block diagrams. This method is popular among the engineers because it is only natural for them to think in terms of structural diagrams. The directive algorithm is decomposed into individual modules and represented as a block diagram. To create a new directive, the designer takes necessary modules from the library and in the *block diagram editor* generates of them a structural diagram of the directive.

The interface of a directive is based on the interface module which is only a description of the directive source data. By marking the inputs or outputs of the corresponding modules, the researcher can choose the data to be displayed during operation of the directive. Figure 4 depicts the window of the block diagram editor with the structural diagram of directive 311. The finished block diagram of the directive is archived and at any time can be modified by the researcher. When executing a directive, the system reads the file describing the block diagram, accesses the indicated modules in the module library, and executes the corresponding files.

The above method of creating directives is handy and evident, provided that the algorithm is simple and the block diagram of the directive consists of a moderate number of modules. However, construction in the block diagram editor of an involved directive with logical transitions and several dozens of modules is time consuming. Additionally, the errors made in the course of constructing the block diagram can be located with difficulty.

To handle such problems, the GAMMA-2PC system has a problem-oriented language GAMMA-1 [13] of directive development. It has facilities for design of the interface (the operators of data input and output of the results to the protocol) and the calculational part of the directive (call of the calculational modules and the control structures). A program in GAMMA-1 consists of sentences ending by ";". At the program head there are sections describing the variables and labels, they are

followed by creation of the dialogue for input of the source data, call of the calculational modules, and output of the results of calculations. The program text is input through the editor built in the user environment. An editor window with the text of a directive is shown in Fig. 5. The GAMMA-1 language is an interpreter. When a directive is activated, each sentence is recognized and executed. The fact that the same calculational modules from the module library are used both to develop a directive in the block diagram form and to program in GAMMA-1 is an important feature of the system.

4. USING THE GAMMA-2PC SYSTEM

We apply the GAMMA-2PC system to construct adaptive control by means of directive 311 "Modal adaptive frequency control" from the subgroup of "Adaptive frequency control" (group of "Adaptive control"). We consider a totally controllable and asymptotically stable controlled plant obeying the equation

$$\dddot{y} + d_2\ddot{y} + d_1\dot{y} + d_0 y = k_1\dot{u} + k_0 u + f, \quad t \geq t_0, \tag{1}$$

where $y(t)$, $u(t)$, and $f(t)$ are, respectively, measurable plant output, test signal, and external disturbance. The coefficients of this equation are unknown numbers; the disturbance boundary is known:

$$f^* \leq 10. \tag{2}$$

Needed is to establish a controller

$$g_2\ddot{u} + g_1\dot{u} + g_0 u = r_2\ddot{y} + r_1\dot{y} + r_0 y \tag{3}$$

such that beginning from time $t \geq t_N > t_0$ ($t_N - t_0$ is the adaptation duration) the characteristic polynomial of system (1), (3) and the desired polynomial

$$\psi(s) = s^5 + 15s^4 + 85s^3 + 226s^2 + 276s + 120 \tag{4}$$

are close.

We describe application of the directive at the stage of designing an experiment for determination of the adaptation algorithm parameters such as the amplitudes and frequencies of the test signal, duration of filtering, and so on. The technological model of the plant and disturbance that was constructed on the basis of the expert (technologist) knowledge is used for this purpose. The physical plant may differ very much from its technological model, but the parameters of the adaptation algorithm as established at this stage will be useful for adaptation based on the output of the physical plant.

The coefficients of the technological model are as follows: $d_2 = 6.2$, $d_1 = 2.62$, $d_0 = 5$, $k(1) = -2$, $k(0) = 5$, $f(t) = 10\sin 6.1t$.

Plant (1) was excited by the test action

$$u(t) = 0.1(\sin 0.2t + \sin 4t + \sin 6t). \tag{5}$$

The filtration delay is equal to one least-frequency period, the time of filtration is two periods, the number of partitions of the maximum-frequency period is 200 (the discrete interval $T = \frac{2\pi}{200\times 6}$).

System (1), (3) was excited by the test action

$$u(t) = 0.05\,(\sin 0.2t + \sin 1t + \sin 2t + \sin 4t + \sin 6t). \tag{6}$$

The delay is one period, the time of filtration is two periods, the number of divisions is 200, (the discrete interval $T = \frac{2\pi}{200\times 6}$).

As the result, the following controller was obtained:

$$\begin{aligned} &g_0 = 5.60, \quad g_1 = 8.76, \quad g_2 = 1; \\ &r_0 = -18.43, \quad r_1 = -24.19, \quad r_2 = -0.66. \end{aligned} \tag{7}$$

The source data input by the user can be seen in Fig. 1. The window with a fragment of the protocol of directive operation is shown in Fig. 2.

REFERENCES

1. *Spravochnik po teorii avtomaticheskogo upravleniya* (Handbook of the Theory of Automatic Control), Krasovskii, A.A., Ed., Moscow: Nauka, 1987.
2. *Dialogovaya sistema proektirovaniya sistem avtomaticheskogo upravleniya , versiya 2* (DISPAS, Dialogue System for Design of Automatic Control Systems, version 2), Moscow: Mosk. Aviat. Inst., 1981.
3. Andrievskii, B.R., Derevitskii, D.R., Spiridonov, A.A., *et al.*, CAD Systems of Adaptive Control Systems: Principles of Design and Input Language, in *Vopr. kibernetiki. Aktual'nye zadachi adaptivnogo upravleniya* (Problems of Cybernetics. Topical Problems of Adaptive Control), Moscow: Kibernetika, 1982.
4. Dorri, M.Kh. and Klimachev, S.N., Some Problems of Computer-aided Structural Design of the Control Systems of Continuous Plants, *Avtom. Telemekh.*, 1982, no. 3, pp. 10–18.
5. Aleksandrov, A.G., Nebaluev, N.A., Asmolova, L.Ya., and Krupenina, L.Ya., *Matematicheskoe obespechenie sinteza i analiza peredatochnykh matrits regulyatorov mnogomernykh lineinykh sistem avtomaticheskogo regulirovaniya (Kompleksy programm GAMMA-1, GAMMA-2 dlya EVM tipa M-220)* (Software for Analysis and Design of the Transfer Matrices of the Controllers of Multivariable Linear Automatic Control Systems (Software Systems GAMMA-1 and GAMMA-2 for the M-220 Computers)), Saratov: Saratov. Polytech. Inst., 1975.
6. Aleksandrov, A.A., Markov, A.A., and Stepanov, M.F., Interactive Package of Application Programs "GAMMA-1M" for Design and Analysis of Linear Multivariable Systems of Control by the Given Precision and Performance, in: *Mezhvuz. nauchn. sb. "Analiticheskie metody sinteza regulyatorov"* (Inst. Annals "Analytical Methods of Controller Design"), Saratov: Saratov. Pedag. Inst., 1982.
7. Alexandrov, A.G. and Panin, S.Yu., GAMMA-1PC as CACSD tools for Practicing Engineers, *Symp. Computer-aided Contr. Syst. Design (CAC'S97)*, Belgium, 1997.
8. Aleksandrov, A.G. and Panin, S.Yu., GAMMA-1PC System for Design of Controllers of Multivariable Systems, *Avtomatizatsiya v Promyshl.*, 2003, no. 3, pp. 18–22.
9. Jamshidi, M. and Herget, C.J., *Computer-aided Control Systems Engineering*, New York: North-Holland, 1985. Translated under the title *Avtomatizirovannoe proektirovanie sistem upravleniya* (Pod. red. M. Dzhamshidi and Ch.Dzh. Khergeta), Moscow: Mashinostoroenie, 1989.
10. Moler, C., *MATLAB—User's Guide*, Albuquerque: Univ. New Mexico, 1980.
11. *MATLAB User's Guide*, MathWorks, 2001.
12. Aleksandrov, A.G., On the Principles of Designing Systems for Analysis of Dynamics and Design of the Control Devices (CAD Systems of Automatic Control Systems), in *Mezhvuz. nauchn. sb. "Analiticheskie metody sinteza regulyatorov"* (Inst. Annals "Analytical Methods of Controller Design"), Saratov: Saratov. Pedag. Inst., 1982.
13. Mikhailova, L.S., Development of the Language Interpreter of the GAMMA-2PC System, in *Tr. EPI MISiS "Robastnoe upravlenie i chastotnaya identifikatsiya"* (Proc. EPI MISiS "Robust Control and Frequency Identification"), Elektrostal': EPI MISiS, 2003.

This paper was recommended for publication by V.V. Kul'ba, a member of the Editorial Board