Fault Accommodation in Discrete Time Dynamic Systems: Fault Decoupling Based Approach


Abstract—Solution to the problem of fault accommodation in dynamic systems described by nonlinear discrete-time models is based on the control law which provides full decoupling with respect to fault effects. Techniques from the algebra of functions are used to solve fault decoupling problem. Solvability conditions are formulated and calculating relations are given for the control law determination.

I. INTRODUCTION

The demand on fault tolerance imposed on critical purpose control systems calls for the use of fault adaptation techniques. There exist two typical ways for adaptation to faults. The first is self-tuning approach called fault accommodation [1] that computes on-line the control law preserving, in case of faults, main performances of the system while minor performances may degrade. The second approach is self-organization that involves system reconfiguration to replace the faulty parts of the system with healthy ones. This paper focuses on the fault accommodation problem.

Conventional approach to the fault accommodation problem solution consists of on-line fault detection and estimation to construct the model of faulty system (so-called model tuning) followed by finding the new (post-fault) control law determination on the base of the tuned model [1]. Different solutions have been proposed to fault detection, estimation as well as the construction of the post fault control law, involving the methods of optimal control [2], H-infinity optimization [3], model matching [4] and adaptive control [5].

Success in application of conventional approach depends on delays regarding the detection and estimation of the faults, and also on the quality and computational time of the post-fault control law. The quality of the post-fault control law is closely related to accuracy of the tuned model, while obtaining accurate model is usually time expansive. To improve the situation, the progressive approach to fault accommodation has been proposed in [2] with step-by-step model tuning / control law updating whenever more accurate information on faults becomes available.

Another approach was proposed in [6] for dynamic systems described by smooth nonlinear differential equations. The feature of this approach is the use of full decoupling with respect to fault effects via dynamic output feedback. The advantage, when compared to the standard approach, is that the new approach does not require on-line fault estimation, being favorable in situations when such estimation is problematic (e.g., due to high frequency deviations of the system parameters caused by the faults). However, even when on-line fault estimates are available, the suggested approach may be applied at the first stage of the progressive fault accommodation, decreasing that way the time to complete this stage. The approach of [6] uses mathematical techniques, called algebra of functions [7].

In this paper, an approach proposed in [6] will be extended to a class of discrete-time dynamic systems described by nonlinear difference equations. Additionally, besides a dynamic feedback solution, we also study the possibility to use the static output feedback. The rest of the paper is organized as follows. Section 2 presents the problem formulation and Section 3 recalls shortly the concepts of algebra of functions, necessary to understand the results of this paper. In Section 4 solution to the problem is proposed. Illustrative and practical examples are considered in Section 5.

II. PROBLEM FORMULATION

Let the system under control be described by the model of the form

\[ x_{k+1} = f(x_k, u_k, \vartheta_k), \quad y_k = h(x_k) \quad (1) \]

where \( k \) stands for discrete time, \( x_k \in X \subseteq \mathbb{R}^n \), \( u_k \in U \subseteq \mathbb{R}^m \) and \( y_k \in Y \subseteq \mathbb{R}^l \) are the vectors of directly unmeasurable state, control and measurable output respectively, \( \vartheta_k \) is the vector of the system parameters subjected to fault actions. The nonlinear vector functions \( f \) and \( h \) in (1) are not necessarily smooth. It is assumed that for healthy system \( \vartheta_k = \vartheta^0 \), where \( \vartheta^0 \) is some known nominal value of the system parameters vector. Under the presence of faults, \( \vartheta_k \) becomes unknown vector function of time. It prevents to the solution of the control problem for the system immediately on the base of the model (1). Therefore, the goal is to eliminate the effect of faults from the system model. To do this, find such model of maximal dimension with a new control that is free from the vector \( \vartheta_k \).
Let the fault in the system has been detected and isolated by known methods [1]. After this, the post-fault control is generated according to the scheme, whose structural representation is given in Fig. 1.

![Figure 1. The structure interpretation of fault tolerant control](image)

In this scheme, the control \( u_k \) is generated according to the rule

\[
 u_k = g(x_k^0, y_k, u_k^*)
\]  

(2)

where \( u_k^* \in U \) is a new vector of control, \( x_k^0 \in \mathbb{R}^q, q \leq n \), is the state vector of the auxiliary dynamic system \( \Sigma^0 \) specified by the equation

\[
x_{k+1}^0 = f(x_k^0, y_k, u_k^*)
\]  

(3)

and \( g \) is the vector function to be found.

An idea of the approach proposed in [6] is based on the assumption that the system (1) with control (2) may be reduced to the system \( \Sigma^* \) of the form (in contrast to [6] given below in a discrete time form)

\[
x_{k+1}^* = f(x_k^*, u_k^*)
\]  

(4)

with a new state vector \( x_k^* \in \mathbb{R}^p, p \leq q \). Description (4) does not depend on the unknown (in a faulty case) vector \( \dot{\theta}_k \). The states of the systems (1) and (4) are assumed to be linked through relationship

\[
x_k^* = \varphi(x_k)
\]  

(5)

for some vector function \( \varphi \). As a result, one may construct fault tolerant control for the system based on the model (4). However, due to the restriction \( p \leq n \), the applicability of such control law is limited: in general, one can eliminate the effect of faults only for some function of state (and not on the full state).

So, the problems to be solved include finding the description for the systems \( \Sigma^0, \Sigma^* \) as well as the new control law (2).

### III. THE ALGEBRA OF FUNCTIONS

In this Section we recall briefly the main definitions and concepts of the mathematical techniques of algebra of functions to be used for the problem solution. The elements of this algebra are vector functions and its main ingredients are: (1) relation of partial preorder, denoted by \( \leq \), (2) binary operations, denoted by \( \times \) and \( \oplus \), (3) binary relation, denoted by \( \Delta \), (4) operators \( m \) and \( M \). The first two elements are defined on the set \( \Phi_X \) of vector functions with the domain being the arbitrary set \( S \) whereas the last two are defined for the set \( \Phi_X \) of vector functions with the domain being the state space \( X \).

**Definition 1** (Relation of partial preorder) Given \( \alpha, \beta \in \Phi_S \), one says that \( \alpha \leq \beta \) iff there exists a function \( \gamma \) such that \( \beta(s) = \gamma(\alpha(s)) \forall s \in S \).

It follows from above that every component of \( \beta \) should be expressed from the components of \( \alpha \), i.e. following equality holds (for differentiated vector functions):

\[
\text{rank } \left( \frac{\partial \alpha}{\partial s} \right) = \text{rank } \left( \frac{\partial \beta}{\partial s} \right).
\]

**Definition 2** (Equivalence) If \( \alpha \leq \beta \) and \( \beta \leq \alpha \), then \( \alpha \) and \( \beta \) are called strictly equivalent, denoted by \( \alpha \equiv \beta \).

Note that relation \( \equiv \) is reflexive, symmetric and transitive. The equivalence relation divides the set \( \Phi_S \) into the equivalence classes containing the equivalent functions. If \( \Phi_S \backslash \equiv \) is the set of all these equivalence classes, then the relation \( \leq \) is partial order on this set. Recall that a lattice is a set with a partial order where every two elements \( \alpha \) and \( \beta \) have a unique supremum (least upper bound) \( \sup(\alpha, \beta) \) and infimum (greatest lower bound) \( \inf(\alpha, \beta) \). The equivalent definition of the lattice as an algebraic structure with two binary operations \( \times \) and \( \oplus \) may be given if for every two elements both operations are commutative and associative and moreover, \( \alpha \times (\alpha \oplus \beta) = \alpha \), \( \alpha \oplus (\alpha \times \beta) = \alpha \). The equivalence follows from the definition of the operations \( \times \) and \( \oplus \) as

\[
\alpha \times \beta = \inf(\alpha, \beta), \quad \alpha \oplus \beta = \sup(\alpha, \beta).
\]  

(6)

Therefore, the triple \( (\Phi_S \backslash \equiv, \times, \oplus) \) is a lattice. In lattice theory it is customary not to operate with \( \inf(\alpha, \beta) \) and \( \sup(\alpha, \beta) \) but with binary operations \( \times \) and \( \oplus \), respectively. In the simple cases, (6) may be used to compute...
\[ \alpha \oplus \beta. \] The rule for operation \( \times \) is simple:
\[ (\alpha \times \beta)(s) = [\alpha(s), \beta(s)]^T. \] However, the product may contain functionally dependent components that have to be found and removed.

**Example 1.** (Computation of the functions \( \alpha \times \beta \) and \( \alpha \oplus \beta \).) Let \( \Phi_S \subseteq \mathbb{R}^3 \), \( \alpha(s) = [s_1 + s_2, s_3]^T \), \( \beta(s) = [s_1 s_3, s_2 s_3]^T. \) To compute \( \alpha \times \beta \) remove the functionally dependent components \( s_2 s_3 \) in \([\alpha(s), \beta(s)]^T = [s_1 + s_2, s_3, s_1 s_3, s_2 s_3]^T\) to get \( \alpha \times \beta(s) \equiv [s_1 + s_2, s_3, s_1 s_3]^T. \) Clearly, both \( \alpha \) and \( \beta \) may be expressed via components of \( \alpha \times \beta \) and therefore by Definition 1 \( \alpha \leq \alpha \times \beta, \ \beta \leq \alpha \times \beta. \) Moreover, by Definition 1, \( \alpha \leq s_3(s_1 + s_2) \) and \( \beta \leq s_3(s_1 + s_2) \) and therefore, \( \alpha \oplus \beta(s) \equiv s_3(s_1 + s_2). \)

**Definition 3** (Binary relation \( \Delta \)). Given \( \alpha, \beta \in \Phi_X \), there exists a function \( f_\alpha \) such that for all \((x, u) \in X \times U\),
\[ (\alpha, \beta) \in \Delta \iff \beta(f(x, u, \theta^0)) = f_\alpha(\alpha(x), u). \]

When \((\alpha, \beta) \in \Delta\), it is said that \( \alpha \) and \( \beta \) form an ordered pair.

Binary relation \( \Delta \) is used for definition of the operators \( m \) and \( M \).

**Definition 4** Operator \( m(\alpha) \) is a function in \( \Phi_X \) that satisfies the following conditions:
\[ (\alpha, m(\alpha)) \in \Delta, \]
\[ (\alpha, \beta) \in \Delta \Rightarrow m(\alpha) \leq \beta. \]

**Definition 5** Operator \( M(\alpha) \) is a function in \( \Phi_X \) that satisfies the following conditions:
\[ (M(\alpha), \alpha) \in \Delta, \]
\[ (\alpha, \beta) \in \Delta \Rightarrow \alpha \leq M(\beta). \]

From definitions 4 and 5 it is obvious that given \( \alpha, m(\alpha) \) is the minimal function, forming a pair with \( \alpha \), and given \( \beta, \) \( M(\beta) \) is the maximal function, forming a pair with \( \beta \).

**Computation of the operator \( m \).** It has been proven that the function \( \gamma \) exists that satisfies the condition
\[ (\alpha \times u) \oplus f \equiv \gamma(f); \]
given \( \alpha, m(\alpha) \equiv \gamma, \) see [7].

**Computation of the operator \( M \).** In the special case when
\[ \beta(f(x, u, \theta^0)) = \sum_{i=1}^{d} a_i(x) b_i(u) \]
where \( a_1(x), a_2(x), \ldots, a_d(x) \) are arbitrary functions and \( b_1(x), b_2(x), \ldots, b_d(x) \) are linearly independent, then
\[ M(\beta) = a_1 \times a_2 \times \ldots \times a_d. \]

**Definition 6** The function \( \alpha \) is said to be \((h, f)-\)invariant if \( (\alpha \times h, \alpha) \in \Delta \).

This definition is a generalization for the discrete-time systems the concept of conditioned invariant (called alternatively \((h, f)-\)invariant) distribution (or codistribution) for continuous-time systems. Notice, Definition 6 is more general since the function \( f \) as well as the function \( \alpha \) is not necessarily differentiable. The next algorithm is important for the problem solution.

**Algorithm 1** (Computation of the minimal \((h, f)-\)invariant function \( \alpha \), satisfying the condition \( \alpha^0 \leq \alpha \)). Given \( \alpha^0 \), compute recursively for \( i \geq 1 \), using the formula
\[ \alpha^{i+1} = \alpha^i \oplus m(\alpha^i \times h). \]

Consider the sequence of nondecreasing functions \( \alpha^0 \leq \alpha^1 \leq \ldots \leq \alpha^j \leq \ldots \). By Theorem 1 in [7], there exists a finite \( j \) such that \( \alpha^j \equiv \alpha^j \).

To support the algebraic operations and operators computation, a number of new Mathematica functions have been developed in order to expand the functionality of the symbolic software package NLControl. These functions are integrated into the package NLControl and using the tools of webMathematica are available for researchers, not having access to Mathematica [8].

**IV. PROBLEM SOLUTION**

**A. The Auxiliary System Design**

Let \( \alpha^0 \) be the vector function with maximal number of functionally independent components such that the forward shift, i.e. \( \alpha^0(f(x_k, u_k, \theta_k)) \) does not depend on \( \theta_k \). In case of smooth \( \alpha^0 \) and \( f \) this requirement results in
\[ \frac{\partial}{\partial \theta} \alpha^0(f(x, u, \theta) = 0. \]

We first construct the auxiliary system \( \Sigma^{\prime} \) of maximal dimension specified by
\[ x_{k+1}^{\prime} = f'(x_k^{\prime}, y_k, u_k) \] (7)
whose state transition map does not depend on \( \theta_k \) and \( x_k^{\prime} \) is related to the state \( x_k \) of the original system (1) via relationship
\[ x_k^{\prime} = \alpha(x_k) \] (8)
where \( \alpha \) is the vector function computed from Algorithm 1, initialized by \( \alpha^0 \). The system \( \Sigma' \) will be used for the systems \( \Sigma^0, \Sigma^* \) design as well as for the control law (2) determination. The features of this system are discussed below.

According to Definition 6 and Algorithm 1,
\[
\alpha^0 \leq \alpha, \quad (\alpha \times h, \alpha) \in \Delta.
\]
The function \( f' \) of the system (7) may be found from known defining equation [7]:
\[
f'(\alpha(x), h(x), u) = \alpha(f(x, u, \hat{\psi}^0)).
\]

By construction, the function \( f' \) does not depend on \( \psi_k \) explicitly but since one of its arguments \( y_k = h(x_k) \), the state \( x' \) may be distorted by the faults for some future time instant. To eliminate this possible dependence of faults, we examine the role of different type of output components \( y_k \).

For that purpose we split the vector \( y_k \) components in three disjoint subvectors. The first one, denoted by \( Y_g \) (where \( g \) stands for good) includes the components of \( y_k \) the function \( f' \) does not depend on. The procedure for determining \( Y_g \) is
\[
\alpha \leq h_i \Rightarrow y_{k,i} \in Y_g.
\]
Next \( y_{k,j} \) is a component of the second subvector \( Y_{ab} \) (\( ab \) stands for absolutely bad) if there exists at least one component of \( f' \) that does not depend on control but depends on \( y_{k,j} \). The remaining components of \( y_k \) are from \( Y_{cb} \) (\( cb \) stands for comparatively bad).

By definition, the components of \( Y_g \) do not need compensation by correction of the control according to the rule (2). Therefore if \( Y_{cb} = Y_{ab} = \emptyset \), then \( u_k = u_k^r \) and the system \( \Sigma^0 \) is excluded from the scheme Fig.1. If \( Y_{cb} \) is not empty subvector, its components may be compensated by a proper choice of the control law (2). However, the components of \( Y_{ab} \) cannot be compensated via control law immediately though it may be possible for components of \( x' \) whose state transition map depends on \( Y_{ab} \). The Algorithm 2 below will analyze \( Y_{ab} \) regarding this possibility.

**Algorithm 2** (the subset \( Y_{ab} \) analysis).

1. For every \( y_{k,i} \in Y_{ab} \) find the components \( x_{k,j} \) such that the function \( f'_{j}(x_{k,j}, y_k, u_k) \) contains \( y_{k,i} \); denote the set of all these components by \( X_b \) and take \( Y_{ab} = \emptyset \).
2. For every \( x_{k,j} \in X_b \) find the function \( f'_{j}(x_{k,j}, y_k, u_k) \), containing \( x_{k,j} \); if this function contains the control vector components then \( x_{k,j} \) is included into \( Y_{cb} \), otherwise \( x_{k,j} \) is included into \( Y_{ab} \). If analysis for every element from \( X_b \) is finished, take \( X_b = \emptyset \).

3. If \( Y_{ab} \neq \emptyset \) go to step 2 under \( Y_{ab} = X_b \).

If after several iterations the set \( Y_{ab} \) is repeated, algorithm stops its work.

The elements of \( Y_{ab} \) obtained as the result of Algorithm 2 can not be compensated by control, and by this reason the equations of (7) that depend on them have to be removed from the model (7). To do this, exclude from \( \alpha \) the components that depend on the elements of \( X_b \) and denote the rest \( \alpha^0 \). Run with this new \( \alpha^0 \) Algorithm 1 to get new \( \alpha \) and finally new system \( \Sigma' \). For this system, \( Y_g, Y_{cb}, Y_{ab} \) should be found and Algorithm 2 reapplied. The procedure ends when at certain iteration one obtains \( Y_{ab} = \emptyset \).

**B. The Control Law Determination**

Equations of the system \( \Sigma' \) contain the elements of \( Y_{cb} \) which may be compensated by appropriate correction of the control vector. Consider an idea of such compensation. Find in the function \( f' \) all terms of the form \( \gamma_i(x_k', y_k, u_k) \), \( i = 1, \ldots, r \), containing the output vector components from \( Y_{cb} \) and the control \( u_k \) as their arguments; it is assumed that the function \( \gamma_i \) contains minimal number of variables. Note that not necessary (though it may happen for some \( i \) \( \gamma_i = f'_i \). Denote \( \gamma = (\gamma_1, \ldots, \gamma_r)^T \) the vector function composed from these components. Assume that
\[
\text{rank } (\partial \gamma / \partial u) = s
\]
for some integer \( s \) and all \( x', y, \) and \( u \) (excluding the set of measure zero). Let
\[
u_{k,i} = \gamma_i(x_k', y_k, u_k), \quad 1 \leq i \leq r.
\]

Suppose that the function \( \gamma \) contains \( m' \) components of the vector \( u \) as its arguments. Clearly, \( m' \geq s \) and \( r \geq s \). Consider four cases, assuming that the function \( \gamma \) contains only functionally independent components and non-smooth components do not include the control vector components as their arguments.

The first case corresponds to \( m' = r = s \). In this case equations (9) are solvable for some \( m' \) components of the control vector. Without loss of generality assume that these components are \( u_1, u_2, \ldots, u_{m'} \). As a result
\[
u_{k,i} = g_{i}'(x_k', y_k, u_k^s), \quad 1 \leq i \leq m',
\]
with some function \( g_{i}' \). For the rest components one can take
\[ u_{k,i} = u_{k,i}^*, \quad m' + 1 \leq i \leq m. \] (11)

In the second case when \( m' > r = s \), the function \( \gamma \) contains \( m' - r \) redundant components of the control vector. Without loss of generality assume that these components are \( u_{r+1}, \ldots, u_m \). Using additional equations of the form

\[ u_{k,i} = u_{k,i}^*, \quad r + 1 \leq i \leq m, \] (12)

solution is given in the form (9) for \( 1 \leq i \leq r \).

In the third case when \( m' \geq r > s \), the matrix \( P \) with \( s \) rows is found to satisfy the condition \( \text{rank} \left( P \frac{\partial \gamma}{\partial u} \right) = s \) for all \( x', y \), and \( u \) (excluding the set of measure zero). As before, let \( u_{k,i} = u_{k,i}^*, \quad s + 1 \leq i \leq m \). In this case equation

\[ u^* = P \gamma \] (13)

is solvable in the form (10) for \( 1 \leq i \leq s \).

In the forth case when \( r > m' \), the number \( r \) of functionally independent equations (9) exceeds the number \( m' \) of the control vector components to be determined from these equations. As a result, equations (9) are not compatible and fault accommodation problem has no solution.

C. The Systems \( \Sigma^0 \) and \( \Sigma^* \) Design

For the system \( \Sigma^0 \) design, the vector \( u^* \) is substituted for the vector \( u \) in the function \( f' \) according to relations (10) and (12). To obtain the system \( \Sigma^0 \) of minimal dimension, the components of \( x' \) in right hand side of (10) are collected into new vector \( z \), denoting this in the form \( z = \rho(x') \) with some vector function \( \rho \). For the composition \( \rho(\alpha) \), find \((h, f)\)-invariant function \( \psi \) with the minimal number of components such that \( \psi \leq \rho(\alpha) \). An algorithm for such function computation has been proposed in [7]; for convenience this algorithm is given below.

Algorithm 3 (Computation of the maximal \((h, f)\)-invariant function).

1. Take \( i = 0, \beta^0 = \rho(\alpha) \).
2. Compute the function \( \gamma' = M(\beta^i) \).
3. If \( h \times \beta^0 \times \beta^1 \times \ldots \times \beta^i \leq \gamma' \) then go to the step 5.
4. Find the function \( \beta^{i+1} \) with minimal number of components, satisfying condition \( h \times \beta^0 \times \beta^1 \times \ldots \times \beta^{i+1} \leq \gamma^i \), take \( i := i + 1 \) and go to the step 2.

5. Take \( \psi = \beta_0^0 \times \beta_1^1 \times \ldots \times \beta_j^j \). End.

The function \( f^0 \) of the system \( \Sigma^0 \) is obtained on the base of the function \( \psi \) by replacing the vector \( x \) with the vectors \( x^0 = \psi(x) \) and \( y \) in the function \( f \). To reduce control law (10) to the form (2), the vector \( x' \) is denoted as \( x_0^0 \).

If the function \( \gamma \) (or \( P \gamma \)) does not contain \( x' \) as its argument, then relation (2) is reduced to the form

\[ u_k = g(y_k, u_k^*) \] (14)

and, therefore, the static feedback exists. Sufficient condition for the existence of the static feedback is given by the following theorem.

Theorem 1. If there exists feedback that provides solution to the fault accommodation problem and \( \alpha^0 \oplus m(h) \neq \text{const} \) then this feedback may be found in the static form.

Proof. Let \( \alpha = \alpha^0 \oplus m(h) \neq \text{const} \). Above condition is equivalent to two functional inequalities: \( \alpha^0 \leq \alpha \) and \( m(h) \leq \alpha \), or \((h, \alpha) \in \Delta \). It follows from the last inclusion that the function \( f'(x', y, u) \) does not depend on \( x' \). It means that the function \( g(x', y, u) \) (if it exists) also does not depend on \( x' \) that corresponds to the static solution.

Notice, the system \( \Sigma^0 \) design completes the fault accommodation problem solution. Under this fulfilling the condition \( \alpha \leq h \) means that all the vector \( y \) components does not depend on the fault actions.

For the system \( \Sigma^* \) design, find equations \( x_{k+1}^i = f_j(x_k^i, y_k, u_k) \), \( x_k \in X_h \) which are being determined by the variables from the set \( X_h \). These equations are excluded from the function \( f' \). After this, replace \( u \) with \( u^* \) in the rest equations of the system \( \Sigma^* \). Notice, obtained equations may contain only such components of the output vector depending on \( x' \), i.e. \( y_{k,j} = h_j^j(x_k^j) \) for some function \( h_j^j \). Replacing \( y_{k,j} \) with \( h_j^j(x_k^j) \), one obtains the function with arguments \( u^* \) and \( x' \). Finally, denoting \( x' \) as \( x^* \), one obtains description of the system \( \Sigma^* \). The state vector of the system \( \Sigma^* \) is linked with the state vector of the initial system (1) by the function \( \phi \) obtained from the function \( \alpha \) by excluding the components related to the variables from the set \( B \). Exactly this function determines the accuracy of the accommodation problem solution.

So, an algorithm of accommodation problem solution looks as follows.
Algorithm 4:
1. Design of the auxiliary system $\Sigma'$.
2. Determining the sets $Y_{cb}$ and $Y_{ab}$.
3. Finding the control law $g$.
4. Design of the system $\Sigma^0$.
5. Design of the system $\Sigma^*$.

V. EXAMPLES

A. Academic Example

This example allows comparing former result [6] with a new one proposed in present paper. Remember that the approach of former papers involved dynamic feedback to eliminate fault effects from the system model. An approach of present paper allows finding more simple (static) solution. To illustrate this fact we use the system from [7].

Consider the problem of full decoupling with respect to the variable $\Theta_k$ in the system described by difference equations

$$ x_{k+1} = \begin{cases} u_k \exp(-\sqrt{x_{k,1}}) \\ x_{k,1} \cos(\Theta_k) \\ x_{k,1} \sin(\Theta_k) \\ x_{k,2} \exp(-\sqrt{x_{k,4}^2 + x_{k,5}^2}) \\ x_{k,3} \exp(-\sqrt{x_{k,4}^2 + x_{k,5}^2}) \end{cases}, \quad y_k = \begin{pmatrix} x_{k,4} \\ x_{k,5} \end{pmatrix}. $$

According to [7], the function $\alpha(x) = (x_1, x_2, x_3)^T$ and the system $\Sigma'$ has a form

$$ x_{k+1} = \begin{cases} u_k \exp(-\sqrt{x_{k,1}^2}) \\ x_{k,2} \exp(-2\sqrt{x_{k,2}}) \end{cases}. $$

In contrast to former result, using the result of this paper it may be shown that since $Y_{cb} = Y_{ab} = \emptyset$, there is no need in control law correction, the system $\Sigma^0$ is absent and the system $\Sigma^*$ coincides with the system $\Sigma'$ (up to designation of variables).

B. The Tracking System

Consider tracking system (Fig.2) with the transfer functions $W_1 = a_1 / p$, $W_2 = a_2 / (1 + T_2 p)$, $W_3 = a_3 T_3 p / (1 + T_3 p)$, $W_4 = a_4 / (1 + T_1 p)$, where $p$ is the operator of differentiation, $\sigma$ describes the saturation: $\sigma(z) = z$ if $|z| \leq z_0$ and $\sigma(z) = z_0 \cdot \text{sign}(z)$, if $|z| > z_0$; $a_1$, $1 \leq i \leq 5$, are the constant coefficients, $T_1$, $T_2$ are the time constants and $T_3 = T_3(\Theta_k)$. Discrete-time model of this system has following form

$$ x_{k+1} = \begin{pmatrix} \tau a_1 x_{k,2} + x_{k,1} \\ (\tau a_2 / T_2) \sigma(x_{k,3}) + x_{k,2}(1 - \tau / T_2) \\ (\tau / T_1)(a_4 a_5 (u_k - x_{k,1}) - a_3 a_4 (x_{k,2} - x_{k,4}) + x_{k,3}(1 - \tau / T_1)) \\ (\tau / T_3(\Theta_k)) x_{k,2} + x_{k,4}(1 - \tau / T_3(\Theta_k)) \end{pmatrix}. $$

where $\tau$ is the sampling period. Find the function $\alpha^0(x) = (x_1, x_2, x_3)^T$. Due to $\alpha^0 \cdot h = (x_1, x_2, x_3, x_4)^T$ one obtains $\alpha(x) = \alpha^0(x)$.

![Figure 2. The tracking system](image_url)

The system $\Sigma'$ takes a form

$$ x_{k+1} = \begin{pmatrix} \tau a_1 x_{k,2} + x_{k,1} \\ (\tau a_2 / T_2) \sigma(x_{k,3}) + x_{k,2}(1 - \tau / T_2) \\ (\tau / T_1)(a_4 a_5 (u_k - x_{k,1}) - a_3 a_4 y_k) + x_{k,4}(1 - \tau / T_1) \end{pmatrix}. $$

It is easy to show that $Y_{cb} = \{y\}$, $Y_{ab} = \emptyset$. One can take $u^* = a_4 a_5 u - a_4 y$. In this case a new control law is given as follows: $u = (u^* + a_4 y) / a_4 a_5$ and the solution has a static form. Under this the system $\Sigma^*$ coincides with the system $\Sigma'$ (up to designation of variables).

REFERENCES