SLIDING CONTROL WITH GENETIC ALGORITHM FOR MISMATCHED DISTURBANCE

Jeang-Lin Chang, Keh-Tsong Li and Yon-Ping Chen

ABSTRACT

This paper presents a new sliding control to deal with linear systems suffering from matched-to-part disturbance and mismatched disturbance. Only parts of the control inputs are designed using the sliding-mode theory in order to eliminate the matched-to-part disturbance. Based on an important criterion, the proposed controller employs the applicable genetic algorithm to effectively reduce the influence of mismatched disturbance during sliding motion. A numeric example is used to demonstrate the effectiveness of the developed controller.

Keywords: Sliding control, mismatched disturbance, genetic algorithm.

I. INTRODUCTION

Sliding control is an effective control algorithm for systems with matched disturbance which spans as same space as the control inputs [1]. If the disturbance spans only parts of the input space, as shown in our previous work [2], only some of the control inputs can be designed using the sliding-mode theory. In addition, many sliding control designers have recently studied mismatched disturbance [3-5]. However, during sliding motion, mismatched disturbance often generates undesirable system behavior and sometimes results in the stability problem [5]. How to effectively reduce the effect of mismatched disturbance in sliding mode is an important topic. Previous papers [3-6] did not give a solution for this problem. This paper will discuss the problem and propose a new sliding control technology which incorporates the appropriate genetic algorithm (GA) to handle systems with matched-to-part disturbance and mismatched disturbance. The genetic algorithms, first proposed by Holland [7], are robust search and optimization techniques and have been applied successfully to many practical problems [8-9]. Besides having effective search ability, the executive process of GA does not require additional data, such as gradient information about the solution space. It merely needs a defined criterion to search for the nearby optimal solution. Our genetic algorithm [11] is a real-coded rank-based method, which is represented by a chromosome containing floating-point parameters. The use of rank-based fitness increases the population diversity; the concepts of age and lifetime reduce the required computation resources. In addition, the novel form of the genetic operators in the evolution process enables generation of offspring.

After the desired eigenvalues for the overall system are assigned, the state feedback matrix and the closed-loop system eigenstructure can be obtained by using the pole assignment method [10]. From this eigenstructure, we can easily determine the sliding vector and predict the performance of a system affected by mismatched disturbance during sliding motion. Once the region for each desired eigenvalue is set up, by using an important criterion, the GA technique proposed in our previous work [11] can be applied to search for eigenvalues, which minimize the effect of mismatched disturbance. Five steps for controller design are summarized, and simulation results show that our algorithm can reduce the effect of mismatched disturbance. Five steps for controller design are summarized, and simulation results show that our algorithm can reduce the effect of mismatched disturbance. Five steps for controller design are summarized, and simulation results show that our algorithm can reduce the effect of mismatched disturbance. Five steps for controller design are summarized, and simulation results show that our algorithm can reduce the effect of mismatched disturbance. Five steps for controller design are summarized, and simulation results show that our algorithm can reduce the effect of mismatched disturbance. Five steps for controller design are summarized, and simulation results show that our algorithm can reduce the effect of mismatched disturbance.
scheme. Finally, concluding remarks are given in Section 5.

II. PROBLEM STATEMENT

Consider the following linear time-invariant system suffering from unknown disturbance:

\[ \dot{x}(t) = Ax(t) + Bu(t) + d(x, t), \]

where \( x \in \mathbb{R}^n, u \in \mathbb{R}^m(m < n), \) and \( d(x, t) \in \mathbb{R}^p, \) respectively, represent the system state, input, and unknown disturbance. Suppose that the system is controllable, and that the input matrix \( B \in \mathbb{R}^{n \times m} \) is of full rank, i.e., \( \text{rank}(B) = m \). The disturbance \( d(x, t) \) is composed of \( d_p(x, t) \in \mathbb{R}^p(p \leq m) \) and \( d_q(t) \in \mathbb{R}^q(q \leq n - m) \), in the form of

\[ d(x, t) = B_p d_p(x, t) + B_q d_q(t), \]

where \( B_p \in \mathbb{R}^{n \times p} \) contains \( p \) columns of \( B \) and \( B_q \in \mathbb{R}^{n \times q} \) satisfies \( B_p B_q = 0 \). Without loss of generality, let \( B_p \) contain the last \( p \) columns of \( B \), i.e., \( B = [B_p, B_q] \), where \( B_q \in \mathbb{R}^{n \times (m - p)} \). As for \( d_q(t) \), it spans only part of the space spanned by the input \( u \). Hence, for convenience, we call it matched-to-part disturbance. Clearly, if \( p = m \), then \( B = B_p \) and \( d_q(t) \) is so-called matched disturbance [1]. In addition, \( d_q(t) \) is generally called mismatched disturbance. From (1) and (2), the system can be written as

\[ \dot{x}(t) = Ax(t) + Bu(t) + B_p d_p(x, t) + B_q d_q(t). \]

Let the disturbance \( d_p(x, t) \) and \( d_q(t) \) be bounded as and

\[ \left| d_p(x, t) \right| \leq \delta_p \left( \left| x \right|, t \right) \quad \text{and} \quad \left| d_q(t) \right| \leq \delta_q(t), \]

where \( \delta_p \left( \left| x \right|, t \right) \) and \( \delta_q(t) \) are both known functions.

In the next section, a novel sliding-mode controller will be proposed, in which only \( p \) control inputs are designed by means of the sliding-mode theory so as to eliminate matched-to-part disturbance. In other words, only \( p \) switching functions have to be implemented.

III. CONTROLLER DESIGN WITH GENETIC ALGORITHM

Since system (3) is controllable and \( B \) is full rank, using the pole-assignment method [10], a state-feedback gain \( K \) can be obtained by assigning \( n \) desired eigenvalues for \( A - BK \). We decompose these \( n \) eigenvalues into two sets, \( \{\lambda_1, \ldots, \lambda_{n_p}\} \) and \( \{\omega_1, \ldots, \omega_q\} \). This implies that \( \{\lambda_1, \ldots, \lambda_{n_p}, \omega_1, \ldots, \omega_q\} \) are the desired eigenvalues for the matrix \( A - BK \). To appropriately design the sliding vector, these \( n \) desired eigenvalues must satisfy the following conditions:

(C1) The eigenvalues are all negative real numbers.
(C2) The value of any repeated eigenvalue is no greater than \( m \).
(C3) The eigenvalues are partitioned into \( \{\lambda_1, \ldots, \lambda_{n_p}\} \) and \( \{\omega_1, \ldots, \omega_q\} \).
(C4) No eigenvalue in \( \{\omega_1, \ldots, \omega_q\} \) is in the spectrum of \( A \).

Sinswat and Fallside [10] showed that \( A - BK \) can be diagonalizable if the value of any repeated eigenvalue of \( A - BK \) is no greater than \( m \), as given in (C2). Hence,

\[ \begin{bmatrix} W & V \end{bmatrix} (A - BK) = \begin{bmatrix} \Lambda & 0 \\ 0 & \Omega \end{bmatrix} \begin{bmatrix} W \\ V \end{bmatrix}, \]

where \( \Lambda = \text{diag}\{\lambda_1, \ldots, \lambda_{n_p}\} \) and \( \Omega = \text{diag}\{\omega_1, \ldots, \omega_q\} \). In addition, \( W \in \mathbb{R}^{(n - p) \times n} \) and \( V \in \mathbb{R}^p \) are the left eigenvectors corresponding to \( \{\lambda_1, \ldots, \lambda_{n_p}\} \) and \( \{\omega_1, \ldots, \omega_q\} \), respectively. Further decomposing (5) yields

\[ W(A - BK) = \Lambda W, \]

\[ V(A - BK) = \Omega V. \]

Note that one condition must be added so that a sliding vector can be chosen: (C5) the matrix \( V \) such that \( VB_q \in \mathbb{R}^{n \times p} \) is invertible.

If \( (VB_q)^{-1} \) exists, the sliding vector can be selected as

\[ s = Cx = Vx. \]

where \( s = [s_1, s_2, \ldots, s_p]^T \in \mathbb{R}^p \) and \( C \in \mathbb{R}^{n \times n} \). Since the system encounters the matched-to-part disturbance \( d_p(x, t) \in \mathbb{R}^p \), this sliding vector only requires \( p \) components, unlike the conventional one that always belongs to \( \mathbb{R}^n \).

Let the control algorithm be

\[ u = -Kx + \begin{bmatrix} 0 \\ v_p \end{bmatrix}, \]

where

\[ v_p = -(VB_q)^{-1} \left( \left| VB_q \delta_q(t) + \sigma \right| \right) \text{sat}(s, \varepsilon), \varepsilon > 0, \sigma > 0. \]

Obviously, if \( u = \begin{bmatrix} u_r \\ u_p \end{bmatrix} \), where \( u_r \in \mathbb{R}^{n - p} \) and \( u_p \in \mathbb{R}^p \), then \( u_r \) is a linear state-feedback control and \( u_p \) is designed
using the sliding-mode theory. Substituting this control algorithm (9) into (3) leads to

$$\dot{x} = (A - BK)x + B \nu_p + B_d \dot{d}_q(x, t) + B_d \dot{d}_q(t).$$ (11)

Then, from (7) and (8), the derivative of $s$ becomes

$$\dot{s} = V(A - BK)x + VB \nu_p + VB_d \dot{d}_q(x, t) + VB_d \dot{d}_q(t) = \Omega s + VB \nu_p + VB_d \dot{d}_q(x, t) + VB_d \dot{d}_q(t).$$ (12)

When $|s| > \epsilon$, we have $sat(s, \epsilon) = s |s|$. Then, pre-multiplying both sides of (12) by $s^T$ yields

$$s^T \dot{s} = s^T \Omega s - s^T \left[VB \nu_p \delta_p(x, t) + VB \delta_p(t) \sigma \right] + s^T \nu_p + s^T VB_d \dot{d}_q(t).$$ (13)

where $\lambda_{min}(\bullet)$ denotes the minimum eigenvalue of $\bullet$. Clearly, the reaching and sliding condition $s^T \dot{s} < -\sigma |s|$ is guaranteed [12]. One can conclude that the system enters the sliding layer in finite time when $|s| > \epsilon$. In addition, once the system enters the layer $|s| \leq \epsilon$, its trajectory will be restricted within [13].

Now, let us turn to system stability analysis in sliding mode. Suppose the system is successfully constrained in the sliding vector $s = 0$. According to the concept of equivalent control [14], we have $\dot{s} \big|_{\nu_p = \nu_p^eq} = 0$, where $\nu_p^eq$ represents the equivalent control. From (12) and $\dot{s} \big|_{\nu_p = \nu_p^eq} = 0$, $\nu_p^eq$ can be obtained as

$$\nu_p^eq = -d_q(x, t) - (CB \nu_p^eq)^{-1}VB_d \dot{d}_q(t) \text{ for } s = 0.$$ (14)

Substituting (14) into (11) yields the reduced-order system for

$$\dot{x} = (A - BK)x + (I - B_d(VB_p)^{-1}V)B_d \dot{d}_q(t) \text{ for } s = 0.$$ (15)

Note that (15) describes the system motion constrained in $s = 0$. Define

$$s = \begin{bmatrix} V \\
W \end{bmatrix} x = Mx \text{ or } x = M^T \begin{bmatrix} V \\
W \end{bmatrix} s = \begin{bmatrix} V^T \\
W^T \end{bmatrix} s.$$

(16)

where $W^T$ and $V^T$ are the generalized inverse matrices for $W$ and $V$, respectively. Pre-multiplying (15) by $W$ and $V$ results in

$$\dot{y} = W(A - BK)x + W(I - B_d(VB_p)^{-1}V)B_d \dot{d}_q(t) = \Lambda y + W(I - B_d(VB_p)^{-1}V)B_d \dot{d}_q(t),$$ (17)

$$s = V(A - BK)x = \Omega Vx = \Omega s = 0.$$ (18)

or in matrix form,

$$\begin{bmatrix} \dot{y} \\
\dot{s} \end{bmatrix} = \begin{bmatrix} \Lambda & 0 \\
0 & 0 \end{bmatrix} \begin{bmatrix} y \\
s \end{bmatrix} + \begin{bmatrix} W(I - B_d(VB_p)^{-1}V)B_d \dot{d}_q \end{bmatrix}.$$ (19)

Therefore, the system behavior in sliding mode can be expressed by the following reduced-order system:

$$\dot{y} = \Lambda y + W(I - B_d(VB_p)^{-1}V)B_d \dot{d}_q,$$ (20)

which is still affected by the mismatched disturbance $d_q(t)$. Although all the eigenvalues in $A$ possess a negative real part, as given in (C1), the mismatched disturbance $d_q(t)$ always affects the system performance through the term $W(I - B_d(VB_p)^{-1}V)B_d \dot{d}_q(t)$. To mitigate this effect, the following criterion is suggested:

$$\min_{W, V} W(I - B_d(VB_p)^{-1}V)B_d.$$ (21)

Since $W$ and $V$ are left eigenvectors corresponding to the assigned eigenvalues \{ $\lambda_{i_1}, ..., \lambda_{i_{n-p}}$ \} and \{ $\omega_{j_1}, ..., \omega_{j_p}$ \}, the criterion is equivalent to

$$\min_{\lambda_{i_1}, ..., \lambda_{i_{n-p}}, \omega_{j_1}, ..., \omega_{j_p}} W(I - B_d(VB_p)^{-1}V)B_d.$$ (22)

The GA is applied to search for the minimal value using eigenvalues as genes. The design procedure is summarized in the following steps:

**Step 1.** Set up the region for each desired eigenvalue, i.e.,

$$\lambda_{i_{max}} > \lambda_{i_{min}} > i_{1}, ..., i_{n-p},$$

$$\omega_{j_{max}} > \omega_{j_{min}} > j_{1}, ..., j_{p}.$$ (23)

**Step 2.** Find all $n$ eigenvalues based on criterion (23).

**Step 3.** Check conditions (C1)-(C4) for the eigenvalues found in Step 2. If true, go to the next step. Otherwise, go back to Step 1.

**Step 4.** Use the pole-assignment method to obtain the feedback gain matrix $K$ and the matrix $V$ (or $C$).

**Step 5.** Check condition (CS). If $VB_p$ is invertible, then the design of control law (9) is complete. Otherwise, go back to Step 1.

The last three steps, Step 3 to Step 5, are quite
straightforward, while the first two steps, Step 1 and Step 2, have to be carefully implemented in the controller design. In Step 1, the region for each desired eigenvalue in (24) should be appropriately chosen according to the system specifications. The most important task is to find the $n$ eigenvalues from criterion (22) in Step 2. The GA presented in our previous work [11] is applied to search for the $n$ eigenvalues in Step 2. Our method is described in the following.

Our genetic algorithm is called the real-coded rank-based genetic algorithm (RCRBGA). The flowchart shown in Fig. 1 contains an initialization process and an evolution process, both of which are iterative. First, the RCRBGA assigns to each chromosome an age of 0 and a lifetime. The lifetime Goodlife (= 4) is assigned to well-located chromosomes, i.e., those alive for 4 generations, while the others are assigned Poorlife (= 1), i.e., waived in the next generation. Furthermore, all the chromosomes are ranked according to their evaluation values and marked with a rank-based fitness from 1 to $N$, where $N$ is the current population size. Rank-based fitness is used to avoid problems that occur when the search prematurely converges or stagnates.

The iterative evolution process starts with an ending condition. If the ending condition is satisfied, then the evolution process terminates, and the best solution is obtained. If the ending condition is not satisfied, the RCRBGA continues and adds age 1 to all the population. Further, the algorithm checks how many chromosomes, say $k$, have completed their lifetimes; they will be deleted in the next generation. Hence, using the rank-based reproduction, real parametric crossover and mutation must generate $k$ offspring.

Next, an age of 0 and a lifetime are assigned to each new chromosome, where Goodlife (= 4) is assigned to well-located chromosomes and Poorlife (= 1) is assigned to the others. Again, the RCRBGA ranks all the chromosomes according to their evaluation values and assigns a rank-based fitness value to each one. The last step in each cycle is to reset the age of the best (rank-1) chromosome to 0. This implies that the current best chromosome is ‘Ageless’ unless some other better chromosomes are found. This is a kind of elitism used to always keep the best chromosome. More details about this algorithm can be found in [11].

**Remark.** Using transformations (16) and (19), (15) can be changed into

$$
\dot{x} = (A - BK)x + (I - B_p(VB_p)^{-1}V)B_q d_q
$$

$$
= M^{-1}\begin{bmatrix}
\Lambda & 0 \\
0 & 0
\end{bmatrix}Mx + \begin{bmatrix}
(I - B_p(VB_p)^{-1}V)B_q d_q \\
0
\end{bmatrix}
$$

for $s = 0$. (24)

If the system has no mismatched disturbance, i.e., $d_q = 0$, then (24) becomes

$$
\dot{x} = (A - BK)x = M^{-1}\begin{bmatrix}
\Lambda & 0 \\
0 & 0
\end{bmatrix}Mx
$$

for $s = 0$ and $d_q = 0$. (25)

Since $\Lambda$ is a diagonal matrix with the elements {$\lambda_1, \ldots, \lambda_{n-p}$} we can conclude that system (21) has $p$ zero eigenvalues and $n-p$ nonzero eigenvalues {$\lambda_1, \ldots, \lambda_{n-p}$} when $s = 0$. In other words, the order of the system in sliding mode is reduced to $n-p$.

In the next section, we will present a numeric example to show how the proposed controller design is implemented.

**IV. EXAMPLE**

Consider the following specific example of system (3):

$$
A = \begin{bmatrix}
-0.05 & 0 & -1 & 0.23 \\
-0.73 & -1.33 & 0.36 & 0 \\
0.01 & 0.10 & -0.33 & 0 \\
0 & 1 & 0 & 0
\end{bmatrix},
B = \begin{bmatrix}
0.04 & 0 \\
1.27 & -20.31 \\
-2.06 & 1.33 \\
0 & 0
\end{bmatrix}
$$
\[ B_p = \begin{bmatrix} 0 \\ -20.31 \\ 1.33 \\ 0 \end{bmatrix}, \quad B_q = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \] (26)

\[ d_p(x, t) = 0.5 \sin(2t), \quad d_q(t) = 0.5 \cos(t). \] (27)

In addition, the disturbances \( d_p(x, t) \) and \( d_q(t) \) are bounded as

\[ \left\| d_p(x, t) \right\| \leq \delta_p \left\| x \right\|, \quad \left\| d_q(t) \right\| \leq \delta_q. \] (28)

Obviously, since \( p = 1 \), only one control input is designed by means of the sliding-mode theory. Thus, the four eigenvalues are assigned as \( \lambda_1, \lambda_2, \lambda_3, \) and \( \omega_1 \). The controller design follows the five steps described in Section 3. For the first step, assume that the system specifications require

\[ \lambda_1 = -1.4, \quad \lambda_2 = -1.5, \quad \lambda_3 = -3, \quad \omega_1 = -4. \] (29)

However, these eigenvalues allow \( \pm 0.5 \) variation. This means that

\[ -0.9 > \lambda_1 > -1.9, \quad -1 > \lambda_2 > -2 \]
\[ -2.5 > \lambda_3 > -3.5, \quad -3.5 > \omega_1 > -4.5. \] (30)

For the purpose of comparison, we will consider two cases.

**Case 1.** If the eigenvalues (29) are used without any further searching, then the matrices \( K, W, \) and \( C \) are found to be

\[ K = \begin{bmatrix} 2.7201 & -0.1030 & -2.4999 & 0.3260 \\ 0.2172 & -0.1626 & -0.1807 & -0.1993 \end{bmatrix}. \]

\[ C = V = \begin{bmatrix} 2.1029 & -0.1487 & -1.5107 & 0.0738 \end{bmatrix}. \]

\[ W = \begin{bmatrix} 2.4557 & 0.4317 & -0.5909 & 1.3902 \\ 0.2821 & 2.0896 & -0.2396 & 3.1617 \\ 0.6417 & -1.1156 & -0.1141 & -3.3394 \end{bmatrix}. \]

\[ W(I - B_p(VB_p)^{-1}V)B_q = 111.0766. \] (31)

**Case 2.** We adopt the GA in order to satisfy criterion (23). The four eigenvalues are used as the genes, and the GA parameters use a population of 100 and 100 generations to search the eigenvalues in the regions given in (30). If the developed GA [11] is applied to the design, then the following data are obtained:

\[ \lambda_1 = -1.8296, \quad \lambda_2 = -1.0001, \quad \lambda_3 = -2.5097, \quad \omega_1 = -3.5064. \] (32)

The matrices \( K, W, \) and \( C \) are, then,

\[ K = \begin{bmatrix} 1.8915 & 0.4145 & -1.8920 & 1.2914 \\ 0.0639 & -0.1360 & -0.0943 & -0.1690 \end{bmatrix}. \]

\[ W = \begin{bmatrix} 1.6480 & -0.5508 & -0.6415 & -1.2373 \\ -1.2677 & 1.0590 & 1.2992 & 0.7374 \\ -2.7700 & -1.0110 & 0.7498 & -3.7158 \end{bmatrix}. \]

\[ C = V = \begin{bmatrix} -2.4583 & -1.7045 & 1.3393 & -3.4954 \end{bmatrix}. \]

\[ W(I - B_p(VB_p)^{-1}V)B_q = 5.2136. \] (33)

Evidently, the value \[ W(I - B_p(VB_p)^{-1}V)B_q \] in (33) is much less than that in (31). This, in some sense, fulfills criterion (21). Figures 1-3 show the system response for two cases under the initial condition \( x(0) = [0 \quad -1 \quad 0 \quad 0]^T \) and control law (9) with \( \varepsilon = 0.01 \) and \( \sigma = 0.1 \). Fig. 2 shows the trajectory of \( s \). Figure 3 gives the motion of the system state \( x \), and the control input is shown in Fig. 4. It is clear from the figures that the performance in case 2 is superior to that in case 1, in terms of robustness and steady-state error.

Fig. 2. The trajectory of \( s \) for two cases.
Fig. 3. (a) The motion of $x_1$ for two cases, (b). The motion of $x_2$ for two cases, (c). The motion of $x_3$ for two cases, (d) The motion of $x_4$ for two cases.

Fig. 4. (a) The input of $u_1$ for two cases, (b). The input of $u_2$ for two cases.

V. CONCLUSIONS

A new sliding controller that incorporates the GA has been developed to deal with matched-to-part disturbance and mismatched disturbance. Most significantly, only some of the system inputs are designed using the sliding-mode theory. By applying the pole-assignment method to the overall system, the sliding vector and the
system’s behavior during sliding motion can be obtained from the closed-loop eigenstructure. Based on an important criterion, the GA is applied to reduce the effect of mismatched disturbance. In addition, we have summarized five steps for controller design, which make the design more straightforward. Finally, a numeric example has been simulated to demonstrate the performance of the developed algorithm.

REFERENCES