2019 IEEE 15th International Conference on Control and Automation (ICCA)
Edinburgh, Scotland, July 16-19, 2019

Model Free Sliding Mode Control for Inspection Robot Based on Pigeon-Inspired Optimization

Lin Chen, Haibin Duan, *Senior Member*, *IEEE*, Songyi Dian, Son Hoang

Abstract— **In this paper, a model free sliding mode controller is designed for a power line inspection robot system that only the I/O information is available. The compact form dynamic linearization (CFDL) technique is firstly applied to approximate the original nonlinear system via an unknown pseudo-partial derivative (PPD) Jacobean matrix. Then an observer is designed to obtain an update law of PPD matrix. Meanwhile, a sliding mode control framework with index reaching law is studied to ensure the error's asymptotically convergence. Besides, the robust stability of the closed-loop system and the selection of controller's parameters are discussed using the Lyapunov theorem. Furthermore, a pigeon-inspired optimization (PIO) algorithm is used to automatically tune these parameters. Eventually, the simulation is carried out to demonstrate the effectiveness of the proposed control scheme.**

I. Introduction

With the development of robotics, robots are considered to be part of the best practices in power system operations and maintenance [1]. In our previous work [2], a self-balance robot with two suspension arms was designed for power line inspection (PLI). When the robot negotiates obstacles, only one arm hangs on the line, wind load and transmission line's vertical vibration may bring great challenge to its balance. In order to make the robot possess better reliability and mobility. Balance control is the first problem to be solved.

To address the dynamic balance control of the PLI robot, the robotic system subjected to disturbances was considered in [2]. After that, a gain scheduled dynamic surface control based on disturbance observer was proposed in [3]. It effectively solved the problem of "explosion of complexity" and had strong robustness to fast time-varying disturbances. These control algorithms mentioned above take care of the robot's balance control to a great degree, but they are designed on the premise that the balance adjustment process of the robot is reasonably described [4]. However, the controlled system's dynamics are difficult to model precisely in practice [5]. Though a relatively accurate mathematical model is established, its characteristics of nonlinearity, time-varying, and uncertainties increase the difficulty of controller design. Despite the difficulty of realizing well control performance. These problems are also inevitable for the other model-based control methods. To solve these problem, model free control methods such as PID control, model free adaptive control (MFAC) are necessary [6, 7].

The MFAC was originally proposed in [7]. It provides a controller design process applicable to a class of nonlinear systems, whose dynamics are poorly modeled. In recent years, a number of theoretical improvements of MFAC for different types of systems have been presented [6-9]. In this paper, motivated by MFAC, a model free sliding mode control (MFSMC) integrated with the PIO is proposed.

The parameters of the designed controller have significant influence on the control effect [10]. However, the fine tuning of controller parameters is a tedious and time-consuming task that always requires experimenters have rich knowledge of control theory and controlled objects. Employing optimization algorithms to tune parameters has received more and more attention[11]. The PIO is an efficient stochastic global optimization technique inspired by pigeon homing behavior. It was originally introduced in [12]. In this paper, it will be employed to choose proper controller parameters and initial values of PPD for MFSMC.

The rest of the paper is organized as follows, Section II describes the mathematical model of the PLI robot's balance adjustment process. Section III gives controller design process of MFSMC. The implementation of PIO is introduced in Section IV. Section V shows the simulation results. Finally, the conclusion is presented in Section VI.

II. Dynamic Model for the PLI Robot's Balance Adjustment Process

The configuration of the robot is shown in Fig. 1 [2]. The mass of counter-weight box accounts for a large proportion of robot's total mass. The term m_1 is the mass of the robot body, m_2 is the mass of the counter-weight box, l is the length of actuator bar, H is the height of the T-shaped base, h_1 is the distance between the cable and the robot body's center of mass (COM) (given at the point A), h_{20} is the distance from the counter-weight box's COM (given at the point B) to the $Y_1O_1Z_1$ plane at $\theta_1 = 0$, θ_1 is the tilt angle between the robot body and the X_1 axis, θ_2 is the angle between the initial and active positions of the actuator bar. The values of some parameters are shown in Table 1.

TABLE 1. The parameters of the robot illustrated in Fig. 1 [3]

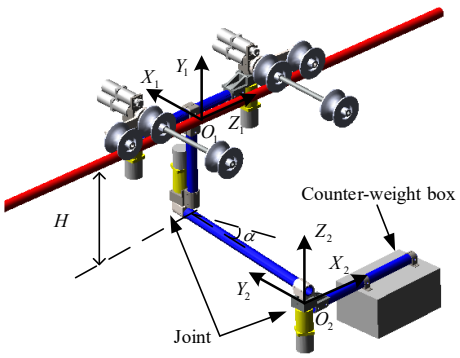
m_1 (kg)	m_2 (kg)	h_1 (m)	h_{20} (m)	l (m)	H (m)
63	27	0.18	0.42	0.5	0.5

Lin Chen is with the School of Automation Science and Electrical Engineering, Beihang University, Beijing 100083, China (e-mail: bulinchen@buaa.edu.cn).

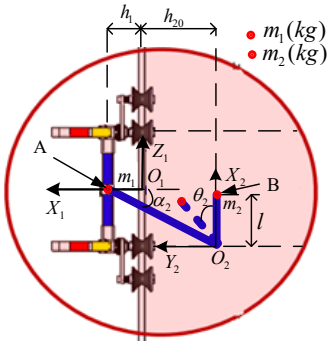
Haibin Duan is with the School of Automation Science and Electrical Engineering, Beihang University, Beijing 100083, China; Peng Cheng Laboratory, Shenzhen, 518000, China (corresponding author to provide phone: 010-82317318; e-mail: hbduan@buaa.edu.cn).

Songyi Dian is with the College of Electrical Engineering and Information Technology, Sichuan University, Chengdu 610065, China (e-mail: scudiansy@scu.edu.cn).

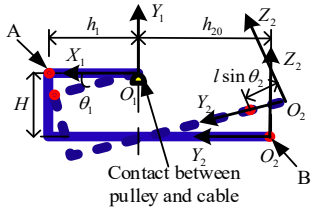
Son Hoang is with the Vietnam National University of Forestry, Hanoi, Vietnam (e-mail: hoangsonbk83@yahoo.com.vn).



(a) 3D model and frame illustration of the PLI robot



(b) Frame illustration in the $X_1O_1Z_1$ plane



(c) Frame illustration in the $X_1O_1Y_1$ plane

Figure 1. Balance adjustment illustration of the PLI robot [2]

Considering the balance adjustment procedure of the PLI robot illustrated in Fig. 1. With Lagrange formulation, the Lagrange equations of motion expressing the process was derived as follows:

$$\begin{aligned} u_1 = &\left[m_1 h_1^2 + m_2 \left(H^2 + (-h_{20} + l\sin\theta_2)^2 \right) \right] \ddot{\theta}_1 \\ &+2m_2 l(-h_{20} + l\sin\theta_2)(\cos\theta_2)\dot{\theta}_1\dot{\theta}_2 \\ &+m_2 gH\sin\theta_1 + m_2 gl\sin\theta_2\cos\theta_1 \end{aligned} \tag{1}$$

$$\begin{aligned} u_2 = &m_2 l^2 \ddot{\theta}_2 + m_2 gl\cos\theta_2\sin\theta_1 \\ &-m_2 l(-h_{20} + l\sin\theta_2)(\cos\theta_2)\dot{\theta}_1^2 \end{aligned} \tag{2}$$

where g is the gravitational acceleration, u_1 and u_2 are the torques associated with variables θ_1 and θ_2, respectively. Let $\mathbf{X} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \theta_1 & \dot{\theta}_1 & \theta_2 & \dot{\theta}_2 \end{bmatrix}^{\mathrm{T}}$ be the state variable vector. Using the first-order Euler discrete method, the discretized model of the robot can be obtained as:

$$\begin{cases} x_1^{(k+1)} = x_1^{(k)} + T_s x_2^{(k)} \\ x_2^{(k+1)} = x_2^{(k)} + \dfrac{T_s \begin{bmatrix} -2m_2 l\left(-h_{20} + l\sin x_3^{(k)}\right)\left(\cos x_3^{(k)}\right) x_2^{(k)} x_4^{(k)} \\ -m_2 g\left(H\sin x_1^{(k)} + l\sin x_3^{(k)}\cos x_1^{(k)}\right) + u_1^{(k)} \end{bmatrix}}{m_1 h_1^2 + m_2\left[H^2 + \left(-h_{20} + l\sin x_3^{(k)}\right)^2\right]} \\ x_3^{(k+1)} = x_3^{(k)} + T_s x_4^{(k)} \\ x_4^{(k+1)} = x_4^{(k)} + \dfrac{T_s \begin{bmatrix} m_2 l\left[-h_{20} + l\sin x_3^{(k)}\right]\left(\cos x_3^{(k)}\right)\left(x_2^{(k)}\right)^2 \\ -m_2 gl\cos x_3^{(k)}\sin x_1^{(k)} + u_2^{(k)} \end{bmatrix}}{m_2 l^2} \end{cases} \tag{3}$$

where T_s is the sampling time.

Remark 1. The mathematical model (3) established in this section is used only to provide online I/O information, it does not participate in the design of controller.

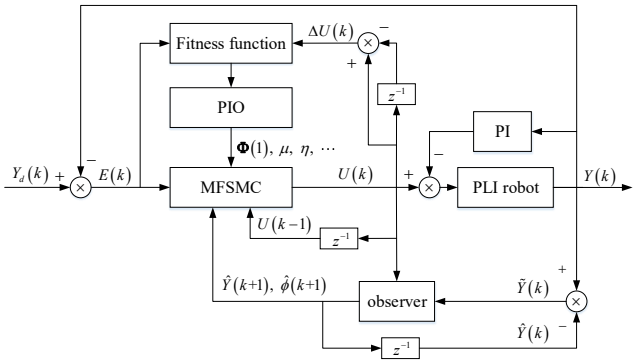


Figure 2. Control block of the PLI robot

Under the constraint that only the system's I/O data are available. A double closed-loop control is proposed. Therein, the PLI robot with inner loop controller is considered as a generalized controlled plant. The MFSMC scheme combined with PIO is designed to make the plant track desired trajectory precisely. The control block is shown as in Fig. 2. Detailed design procedures are given in the following sections.

III. Design of MFSMC and Stability Analysis

A. The CFDL Expression of the Robotic System

A discrete two input two output nonlinear system corresponding to the controlled plant can be described as:

$$\mathbf{Y}(k+1) = f\left(\mathbf{Y}(k),\cdots,\mathbf{Y}(k-dy),\mathbf{U}(k),\cdots,\mathbf{U}(k-du)\right) \tag{4}$$

where $\mathbf{U} = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^{\mathrm{T}}$ and $\mathbf{Y} = \begin{bmatrix} y_1 & y_2 \end{bmatrix}^{\mathrm{T}}$ are the system's input and output vectors. The terms dy and du are unknown order of input and output. The term $f(\square)$ denotes unknown nonlinear function vector.

Before we use the CFDL technique to linearize nonlinear system (4), the following assumptions are imposed [8]:

Assumption 1. The controlled system is generalized Lipshitz.

Assumption 2. The partial derivatives of nonlinear functions with respect to control inputs are continuous.

Remark 2. These assumptions are reasonable and widely used in the related literature (detailed explanation could be founded in [8] and references therein).

Under these assumptions, system (4) can be approximated by the following CFDL equation:

$$\Delta\mathbf{Y}(k+1)=\mathbf{Y}(k+1)-\mathbf{Y}(k)=\mathbf{\Phi}(k)\Delta\mathbf{U}(k) \tag{5}$$

where $\left|\Delta\mathbf{U}(k)\right|\neq 0$, and $\mathbf{\Phi}(k)$ is the PPD Jacobean matrix, whose composition is shown as:

$$\mathbf{\Phi}(k)=\begin{bmatrix}\mathbf{\Phi}_1(k)\\ \mathbf{\Phi}_2(k)\end{bmatrix}=\begin{bmatrix}\varphi_{11}(k) & \varphi_{12}(k)\\ \varphi_{21}(k) & \varphi_{22}(k)\end{bmatrix} \tag{6}$$

In (6), $\mathbf{\Phi}(k)$ is assumed to be bounded and whose norm satisfies $\left\|\mathbf{\Phi}(k)\right\|\le B$. Since it is unknown and related to the system's I/O information, as well as the iteration time k. Then in the next subsection, an update law for $\mathbf{\Phi}(k)$ is designed based on an established observer.

B. The Estimation of PPD Jacobean Matrix

For system (5), an observer with the following structure is firstly constructed [13].

$$\hat{\mathbf{Y}}(k+1)=\hat{\mathbf{Y}}(k)+\hat{\mathbf{\Phi}}(k)\Delta\mathbf{U}(k)+c\tilde{\mathbf{Y}}(k) \tag{7}$$

where $\hat{\mathbf{Y}}(k)$ denotes the estimated output vector, $\tilde{\mathbf{Y}}(k)=\mathbf{Y}(k)-\hat{\mathbf{Y}}(k)$ is the error vector between the estimated output and the actual output, $\hat{\mathbf{\Phi}}(k)$ is the estimation of $\mathbf{\Phi}(k)$, c is a design parameter.

Combining (5) and (7), we can obtain that

$$\tilde{\mathbf{Y}}(k+1)=\Delta\mathbf{Y}(k+1)-\hat{\mathbf{\Phi}}(k)\Delta\mathbf{U}(k)+(1-c)\tilde{\mathbf{Y}}(k) \tag{8}$$

Now let $\tilde{\mathbf{\Phi}}(k)=\mathbf{\Phi}(k)-\hat{\mathbf{\Phi}}(k)$ be the estimation error of $\mathbf{\Phi}(k)$ and substitute (5) into (8), (8) can be rewritten as:

$$\tilde{\mathbf{Y}}(k+1)=\tilde{\mathbf{\Phi}}(k)\Delta\mathbf{U}(k)+(1-c)\tilde{\mathbf{Y}}(k) \tag{9}$$

In this paper, the update law for PPD matrix is designed as:

$$\Delta\hat{\mathbf{\Phi}}(k+1)=\left[\tilde{\mathbf{Y}}(k+1)-(1-c)\tilde{\mathbf{Y}}(k)\right]\frac{2\Delta\mathbf{U}^{\mathrm{T}}(k)}{\left(\left\|\Delta\mathbf{U}(k)\right\|^2+\mu\right)} \tag{10}$$

where μ is a design parameter.

Here, similar to [8], a reset rule is introduced into the estimation of PPD matrix $\mathbf{\Phi}(k)$ to make parameter update law have a better ability in tracking time-varying parameter. It is that if $\hat{\mathbf{\Phi}}(k+1)$ satisfies the following relationship:

$$\left\|\hat{\mathbf{\Phi}}(k+1)\right\|\ge\varphi_M \ \ or \ \ \left\|\hat{\mathbf{\Phi}}(k+1)\right\|<\varphi_\varepsilon \tag{11}$$

where φ_M and φ_ε are positive constants. Then

$$\hat{\mathbf{\Phi}}(k+1)=\hat{\mathbf{\Phi}}(1) \tag{12}$$

where $\hat{\mathbf{\Phi}}(1)$ is the initial value of $\mathbf{\Phi}(k)$. From (11) and (12), it can be observed that the introduction of (12) makes the control scheme highly dependent on the initial value $\hat{\mathbf{\Phi}}(1)$. Then in next section the employment of PIO for choosing parameter properly is important for the control scheme.

Theorem 1. Considering the system (4), under the given assumptions, the adaptive observer is designed by (7), the PPD matrix is estimated by (10-12). The estimation errors $\tilde{\mathbf{\Phi}}$ and $\tilde{\mathbf{Y}}$ in the system are asymptotically converge to the origin by choosing proper design parameter.

Proof. It is assumed once again that the PPD matrix $\mathbf{\Phi}(k)$ is varying slowly at a short time interval, that is $\mathbf{\Phi}(k+1)-\mathbf{\Phi}(k)=0$. Then we have

$$\tilde{\mathbf{\Phi}}(k+1)-\tilde{\mathbf{\Phi}}(k)=-\left[\hat{\mathbf{\Phi}}(k+1)-\hat{\mathbf{\Phi}}(k)\right] \tag{13}$$

Substituting (9) and (10) into (13), we can obtain that

$$\tilde{\mathbf{\Phi}}(k+1)=\tilde{\mathbf{\Phi}}(k)-\frac{2\tilde{\mathbf{\Phi}}(k)\Delta\mathbf{U}(k)\Delta\mathbf{U}^{\mathrm{T}}(k)}{\left(\left\|\Delta\mathbf{U}(k)\right\|^2+\mu\right)}=\tilde{\mathbf{\Phi}}(k)\mathbf{R} \tag{14}$$

To make the estimation errors $\tilde{\mathbf{\Phi}}$ and $\tilde{\mathbf{Y}}$ converge to zero, a positive definite Lypunov function is chosen as :

$$V_{ei}(k)=\zeta\tilde{\mathbf{\Phi}}_i(k)\tilde{\mathbf{\Phi}}_i^{\mathrm{T}}(k)+\beta\tilde{y}_i^2(k) \tag{15}$$

where $i=1,2$, ζ and β are positive constants. Then according to (14), the differential equation of (15) can be obtained as:

$$\begin{aligned}&V_{ei}(k+1)-V_{ei}(k)\\&=\zeta\tilde{\mathbf{\Phi}}_i(k)\left(\mathbf{R}\mathbf{R}^{\mathrm{T}}-I\right)\tilde{\mathbf{\Phi}}_i^{\mathrm{T}}(k)\\&+\beta\tilde{y}_i^2(k+1)-\beta\tilde{y}_i^2(k)\end{aligned} \tag{16}$$

Substituting (8) into (16), we have

$$\begin{aligned}&\Delta V_{ei}(k+1)\\&=\tilde{\boldsymbol{\Phi}}_i(k)\left[\zeta\left(\mathbf{R}\mathbf{R}^{\mathrm{T}}-I\right)+\beta\Delta\mathbf{U}(k)\Delta\mathbf{U}^{\mathrm{T}}(k)\right]\tilde{\boldsymbol{\Phi}}_i^{\mathrm{T}}(k)\\&+2\beta\left[\tilde{\boldsymbol{\Phi}}_i(k)\Delta\mathbf{U}(k)\right]\left[(1-c)\tilde{y}_i(k)\right]\\&+\beta\left[(1-c)^2-1\right]\tilde{y}_i^2(k)\end{aligned}\tag{17}$$

Let $\lambda_i=\tilde{\boldsymbol{\Phi}}_i(k)\Delta\mathbf{U}(k)$, then (17) can be rewritten as:

$$\begin{aligned}&\Delta V_{ei}(k+1)\\&=\left[-2\zeta\mu\left(\left\|\Delta\mathbf{U}(k)\right\|^2+\mu\right)^{-2}+\beta\right]\lambda_i^2\\&+2\beta\lambda_i\left[(1-c)\tilde{y}_i(k)\right]+\beta\left[(1-c)^2-1\right]\tilde{y}_i^2(k)\end{aligned}\tag{18}$$

Based on the theory of the average value inequality, equation (18) can be rewritten as

$$\begin{aligned}&\Delta V_{ei}(k+1)\\&\le\left[-2\zeta\mu\left(\left\|\Delta\mathbf{U}(k)\right\|^2+\mu\right)^{-2}+\beta\right]\lambda_i^2\\&+\left(\varepsilon\beta(1-c)\lambda_i\right)^2+\left(\varepsilon^{-1}\tilde{y}_i(k)\right)^2\\&+\beta\left[(1-c)^2-1\right]\tilde{y}_i^2(k)\end{aligned}\tag{19}$$

where ε is a positive constant. From (19), we can get that in order to make $\Delta V_{ei}(k+1)$ negative definite, the corresponding design parameters should be chosen to satisfy the following condition:

$$\begin{cases}(1-c)^2-1+\varepsilon^{-2}\le 0\\ \beta+\varepsilon^2\beta^2(1-c)^2\le 2\zeta\mu^{-1}\end{cases}\tag{20}$$

It can be proved that the estimation errors $\tilde{\boldsymbol{\Phi}}$ and $\tilde{\mathbf{Y}}$ in the system is asymptotically stable by choosing proper design parameters c and μ.

C. Design of MFSMC

Before we move on to the next stage, let's define the following two error variables:

$$e_i(k)=y_{di}(k)-y_i(k)\tag{21}$$

$$e_{oi}(k)=y_{di}(k)-\hat{y}_i(k)\tag{22}$$

where $e_i(k)$, $i=1,2$ is the system tracking error, $e_{oi}(k)$ is the observer error, $y_{di}(k)$ is the desired output for $y_i(k)$.

The controller characterized by a discrete-time integral sliding model framework is designed [14]. The sliding mode surface is chosen as :

$$\begin{cases}\mathbf{S}(k)=k_P\mathbf{E_O}(k)+k_I\boldsymbol{\upsilon}(k)\\ \boldsymbol{\upsilon}(k)=\boldsymbol{\upsilon}(k-1)+T_s\mathbf{E_O}(k)\end{cases}\tag{23}$$

where $\mathbf{S}=[s_1\ \ s_2]^{\mathrm{T}}$, $\mathbf{E_O}=[e_{o1}\ \ e_{o2}]^{\mathrm{T}}$, $\boldsymbol{\upsilon}=[\upsilon_1\ \ \upsilon_2]^{\mathrm{T}}$, the terms k_P and k_I are the proportion gain and integer gain of sliding mode surface, respectively. An index reaching law presented in [15] is chosen as the reaching law:

$$\Xi\left(s_i(k)\right)=T_s\left[-k_s s_i(k)-\gamma\,\mathrm{sgn}\left(s_i(k)\right)\right]\tag{24}$$

where $0<T_sk_s<1$, and $\gamma T_s>0$. The use of (24) guarantees good performance in reaching to the sliding mode surface, and makes the calculation of control law become simple and intuitive. In order to make (24) meet the reaching condition of sliding mode [15].

To obtain the control law, the difference equation of (23) is deduced to show the control variable:

$$\begin{aligned}&\mathbf{S}(k+1)-\mathbf{S}(k)\\&=k_P\mathbf{E_O}(k+1)+k_I\boldsymbol{\upsilon}(k+1)-\left[k_P\mathbf{E_O}(k)+k_I\boldsymbol{\upsilon}(k)\right]\\&=\left(k_P+k_IT_s\right)\begin{bmatrix}\mathbf{Y_d}(k+1)-\hat{\mathbf{Y}}(k)\\ -c\tilde{\mathbf{Y}}(k)-\hat{\boldsymbol{\Phi}}(k)\Delta\mathbf{U}(k)\end{bmatrix}-k_P\mathbf{E_O}(k)\end{aligned}\tag{25}$$

where $\mathbf{Y_d}=[y_{d1}\ \ y_{d2}]^{\mathrm{T}}$. Because (25) must be equal to (24), we can get the following control input:

$$\Delta\mathbf{U}(k)=\frac{\hat{\boldsymbol{\Phi}}(k)}{\left\|\hat{\boldsymbol{\Phi}}(k)\right\|^2+\eta}\begin{bmatrix}\mathbf{Y_d}(k+1)-\hat{\mathbf{Y}}(k)-c\tilde{\mathbf{Y}}(k)\\ -\dfrac{k_P\mathbf{E_O}(k)+\Xi\left(\mathbf{S}(k)\right)}{\left(k_P+k_IT_s\right)}\end{bmatrix}\tag{26}$$

Theorem 2. For the nonlinear robotic system (4), the use of index reaching law (24) and control law (26) can ensure sliding mode surface (23) coverage to a bounded area with a thickness of $\left(2-k_sT_s\right)^{-1}\gamma T_s$. Besides, the closed-loop system's tracking error (23) and observer tracking error (22) are all bounded by $2\gamma T_s\left(2-k_sT_s\right)^{-1}$.

Proof. According to (24) and the reaching condition of sliding mode, we have

$$\left[s_i(k+1)-s_i(k)\right]\mathrm{sgn}\left(s_i(k)\right)=-k_sT_s\left|s_i(k)\right|-\gamma T_s<0\tag{27}$$

$$\left[s_i(k+1)+s_i(k)\right]\mathrm{sgn}\left(s_i(k)\right)=\left(2-k_sT_s\right)\left|s_i(k)\right|-\gamma T_s\tag{28}$$

From (28), it can be observed that the system takes equal amplitude crossing movement on both sides of the switching surface to form a boundary layer with a thickness of $\left(2-k_sT_s\right)^{-1}\gamma T_s$, i.e. $\lim_{k\to\infty}\left|s_i(k)\right|\le\left(2-k_sT_s\right)^{-1}\gamma T_s$. By

appropriately choosing γ and k_s, $s_i(k)$ can be made arbitrarily small.

According to (25), we can obtain that

$$\left|e_{oi}(k+1)\right| \le \frac{2\gamma T_s}{(k_P + k_I T_s)(2 - k_s T_s)} + \frac{k_P \left|e_{oi}(k)\right|}{(k_P + k_I T_s)} \tag{29}$$

Then the observer tacking error is bounded and

$$\lim_{k\to\infty}\left|e_{oi}(k)\right| \le \frac{2\gamma k_I}{(2 - k_s T_s)} \tag{30}$$

From (19), (21) and (22), it can be obtained that the error $e_i(k) = e_{oi}(k) - \tilde{y}_i(k)$ satisfies:

$$\lim_{k\to\infty}\left|e_i(k)\right| \le \frac{2\gamma k_I}{(2 - k_s T_s)} \tag{31}$$

IV. The Parameter Optimization of MFSMC using PIO

According to [12], the PIO algorithm thinks that the pigeons' homing behaviour is mainly rely on two tools: the map and compass operator, the landmark operator.

Map and compass operator: all pigeons use magnetoreception to sense the earth magnetic field, and adjust their direction in accordance with the altitude of sun. The PIO regards each pigeon's position $\mathbf{P}_i = [p_{i1}, p_{i2}, \cdots, p_{iD}]$ as a potential solution to a problem in D-dimensional search space. Besides, every pigeon also has a corresponding velocity that is expressed as $\mathbf{V}_i = [v_{i1}, v_{i2}, \cdots, v_{iD}]$. At each iteration k, the position $\mathbf{P}_i(k)$ with the best fitness is chosen as the global best position $\mathbf{P}_{gbest}$ to update the velocity. The mathematical expression of this process can be given as:

$$\begin{cases} \mathbf{V}_i(k) = \mathbf{V}_i(k-1)e^{-rt} + rand\left(\mathbf{P}_{gbest} - \mathbf{P}_i(k-1)\right) \\ \mathbf{P}_i(k) = \mathbf{P}_i(k-1) + \mathbf{V}_i(k) \end{cases} \tag{32}$$

where r is the map and compass factor, $rand$ generates random number.

Landmark operator: when pigeons fly close to their destination, they will rely more on the landmarks to arrive at the destination. In this stage, pigeons' number will be reduced by halt in each iteration. The pigeons with a better fitness will be kept, and the other half with poor fitness will be discarded. The remaining pigeons' center position $\mathbf{P}_{center}(k)$ is considered as the landmark for the reference flight direction, according to the following equation to modify the position. Eventually, we can obtain the optimal value.

$$\begin{cases} \mathbf{P}_{center}(k) = \sum_{i=1}^{Np} \mathbf{P}_i(k)\, fitness\left(\mathbf{P}_i(k)\right) / \sum_{i=1}^{Np} fitness\left(\mathbf{P}_i(k)\right) \\ \mathbf{P}_i(k) = \mathbf{P}_i(k-1) + rand\left(\mathbf{P}_{center}(k) - \mathbf{P}_i(k-1)\right) \end{cases} \tag{33}$$

where $Np(k) = Np(k-1)/2$.

V. Simulation Results

In this section, simulations are carried out to verify the effectiveness of the MFSMC algorithm based on PIO (it is referred to MFSMC-PIO in the following text) to the PLI robot system. Simultaneously, the simulation compared with PID and MFSMC without PIO is conducted.

Firstly, PIO is used to obtain reasonable controller parameter. Its main parameters are given as: iterations for the map and compass operator $N_{i\max 1} = 50$, iterations for the landmark operator $N_{i\max 2} = 40$, pigeon swarm size $N_p = 30$, map and compass factor $r = 0.3$, and dimension $D = 11$. The objective of the controller parameter optimization is to guarantee the system's tracking error is small enough and ensure the smoothness of the control input. Then the following fitness function are defined.

$$J = \sum_{k=1}^{N}\sum_{i=1}^{2}\left(e_{oi}^2(k) + \Delta u_i^2(k) + \tilde{y}_i^2(k)\right) \tag{34}$$

One group optimal results for the controller parameters is listed in Table 2. Besides, the sampling time is set as $T_s = 0.01$. The gain for PID controller are chosen as $K_{p1} = K_{p2} = 300$, $K_{i1} = 1.3$, $K_{d1} = 3000$, $K_{i2} = 0.4$, $K_{d2} = 4000$.

TABLE 2. THE PARAMETER SETTING FOR MFSMC

Parameter	Min	Max	Initial value	Optimal result
$\phi_{ij}(1)$	0	20	[15 4.2; 5 4.8]	[10.5 2.2; 1.8 10]
k_P	0	50	4.3	0.05
k_I	0	10	2.1	0.6
η	0	1	0.8	0.2
μ	0	1	1.0	0.3
k_s	0	100	48	52
γ	0	10	3.3	1
c	0	2	1.6	0.875

The comparison simulation results of PID, MFSMC and MFSMC-PIO are shown in Figs.4-6, where Fig. 4 and Fig. 5 imply position tracking of system outputs. It can be observed that the system outputs are asymptotically balanced at the desired position under the control of PID, MFSMC and MFSMC_PIO. However, it should be noted that the system's tracking response with PID controller has an oscillation at the beginning. The MFSMC provides poor rapidity and stability for tracking response. Although they finally are able to track accurately. When the parameters are adjusted by PIO, the system's tracking performance has been improved in terms of rapidity and stability. Fig. 6 shows the estimation of PPD Jacobean matrix. From the last two figures, we can get that the PPD matrix for the CFDL model (5) is not unique. Its initial value has a great influence on the control effect.

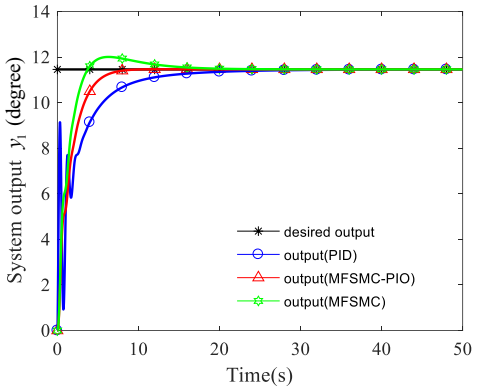


Figure 3. Trajectory of system output y_1 (angle θ_1)

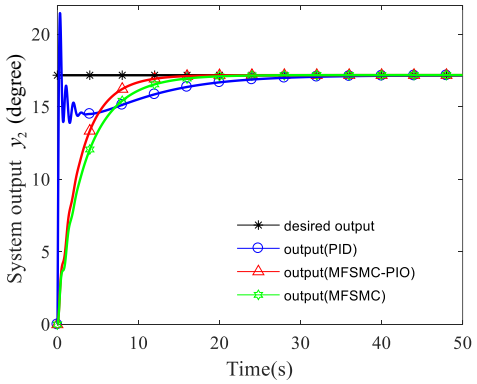


Figure 4. Trajectory of system output y_2 (angle θ_2)

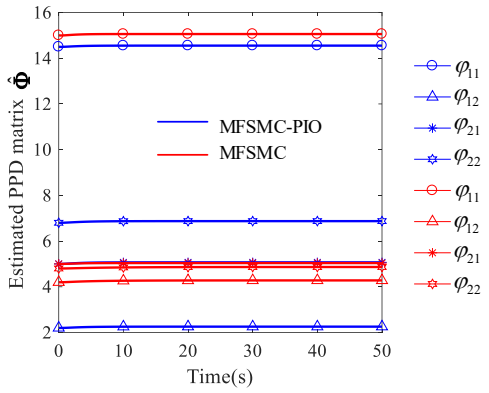


Figure 5. Trajectory of estimated PPD matrix $\hat{\Phi}$

Clearly, the performance of the closed-loop system is significantly improved. The proposed MFSMC-PIO is able to achieve the following goals: (1) It realizes accurate tracking mission under the circumstance of only the system I/O information is known. (2) It enhances both the immediacy of rapidity and stability.

VI. Conclusions

In this paper, a model free sliding mode controller is designed to address the balance adjustment of a power line inspection robot. Under the circumstance of only the I/O information are available. The original nonlinear system is approximated by the CFDL technique via an unknown PPD Jacobean matrix. Then an observer is designed to obtain the update law of PPD matrix. In addition, the employed SMC with index reaching law greatly simplifies the controller design and enhances the robustness. Meanwhile, it is proved that all the error signals in the closed-loop system are bounded using Lyapunov theorem. In addition, the effectiveness of the presented MFSMC-PIO is demonstrated by simulation. Comparisons with PID and MFSMC imply that the proposed algorithm ensures a fast, accurate and stable tracking mission.

Acknowledgment

This work was partially supported by National Natural Science Foundation of China under grants #91648205, #61425008.

References

[1] N. Pouliot, S. Montambault, "LineScout Technology: From inspection to robotic maintenance on live transmission power lines," in *Conf. Rec. 2009 IEEE Int. Conf. Robotics & Automation,* pp. 1034–1040.

[2] S. Y. Dian, L. Chen, S. Hoang, et al. "Dynamic balance control based on an adaptive gain-scheduled backstepping scheme for power-line inspection robots," *IEEE/CAA Journal of Automatica Sinica*, pp. 1–11, 2017.

[3] S. Y. Dian, L. Chen, S. Hoang, et al. "Gain scheduled dynamic surface control for a class of underactuated mechanical systems using neural network disturbance observer," *Neurocomputing*, Vol. 275, 2017.

[4] Y. H. Chang, W. S. Chan, C. W. Chang, T-S fuzzy model based adaptive dynamic surface control for ball and beam system. *IEEE Trans. Industrial Electronics*, Vol. 60, No. 6, pp. 2251–2263, 2013.

[5] J. X. Xu, Z. S. Hou, "Notes on data-driven system approaches," *Acta Automatica Sinica*, Vol. 35, No. 6, pp. 668–675, 2009.

[6] S. S. Han, H. Wang, Y. Tian, "Model-free based adaptive nonsingular fast terminal sliding mode control with time-delay estimation for a 12 DOF multi-functional lower limb exoskeleton," *Advances in Engineering Software*, Vol. 119, pp. 38–47, 2018.

[7] Z. S. Hou, Z. G. Han, "Robust modelless learning adaptive control of nonlinear systems," *Control and Decision*, No. 2, pp. 137–142, 1995. (in Chinese)

[8] Z. S. Hou, S. Liu, T. Tian, "Lazy-learning-based data-driven model-free adaptive predictive control for a class of discrete-time nonlinear systems," *IEEE Trans. Neural Networks & Learning Systems*, Vol. 28, No. 8, pp. 1914–1928, 2016.

[9] D. Z. Xu, F. Liu, B. Jiang, "Improved data driven model free adaptive constrained control for a solid oxide fuel cell," *IET Control Theory & Applications*, Vol. 10, No. 12, pp. 1412–1419, 2016.

[10] E. V. Kumar, G. S. Raaja, J. Jerome, "Adaptive PSO for optimal LQR tracking control of 2 DOF laboratory helicopter," *Applied Soft Computing*, Vol. 41, pp. 77–90, 2016.

[11] Y. M. Deng, H. B. Duan, "Control parameter design for automatic carrier landing system via pigeon-inspired optimization," *Nonlinear Dynamics*, Vol. 85, No. 1, pp. 1–10, 2016.

[12] H. B. Duan, P. X. Qiao, "Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning," *International Journal of Intelligent Computing & Cybernetics*, Vol. 7, No. 1, pp. 24–37, 2014.

[13] B. Kurkcu, C. Kasnakoglu, M. O. Efe, "Disturbance/uncertainty estimator based integral sliding mode control," *IEEE Trans. Automatic Control*, No. 99, pp.1–1, 2018.

[14] Y. Xiong, Y. Li, "Modified internal model control scheme for the drive part with elastic joints in robotic system," *Journal of Intelligent & Robotic Systems*, Vol. 79, No. 3-4, pp. 475–485, 2015.

[15] K. Abidi, J. X. Xu, X. Yu, "On the discrete-time integral sliding-mode control," *IEEE Trans. on Automatic Control*, Vol. 52, No. 4, pp. 709–715, 2007.