Data-Driven Model Free Adaptive Sliding Mode Control for Multi DC Motors Speed Regulation

Tony Blaise Bimenyimana

Abstract

This paper introduces a model free adaptiive sliding mode control (MFASMC) system for multi-DC motors speed regulation. First, the designed control system combines the model-free adaptive control (MFAC) with sliding mode control (SMC) approach. Secondly, the control law is derived by integrating the sliding surface definition with the reaching law. Then, to improve measurement accuracy, the quadruple-frequency data processing method is applied to enhance the precision of encoder measurements and sampling period consistency for DC motors. Additionaly, to accurate speed regulation, the nonlinear multi-agent system with a fixed communication topology is implemented. To further optimize performance, the compact form dynamic linearization (CFDL) technique is incorporated, refinig the control gains and improve the system's adaptability to changing conditions. Finally, simulation results confirm that the presented system effectively tracks the desired trajectory, demonstrating the effectiveness of the proposed design.

Keywords

Model-free adaptive sliding mode control, quadruple-frequency data processing method, compact form dynamic linearization, nonlinear multiagent system, multi-DC motor speed regulation.

I. Introduction

The field of MASs has witnessed significant advancements, with many different applications in several areas such as satellite formations, autonomous vehicles, and distributed robotics. Robust speed regulation is a critical aspect of these systems, especially when dealing with multiple DC motors. Traditional control methods, such as Proportional-Integral-Derivative (PID) and Linear Quadratic Regulator (LQR), often fall short in handling the complex dynamics and uncertainties inherent in multi-agent systems.

This paper proposes a novel control strategy that combines the strengths of Model-Free Adaptive Control (MFAC) and Sliding Mode Control (SMC). MFAC, as demonstrated in [2], offers adaptability to real-time changes, while SMC ensures robustness against uncertainties and disturbances [13]. This synergistic approach aims to enhance the performance, stability, and reliability of multi-DC motor speed regulation within multi-agent systems.

As highlighted in [3], conventional control methods often rely on accurate mathematical models of the system. However, due to the complexity and unpredictability of real-world scenarios, obtaining such models can be challenging. The proposed data-driven control strategy leverages the MFAC framework to circumvent this limitation, offering a model-free solution that is highly adaptable to changes in system dynamics as explored in [4] , [6].

By combining MFAC and SMC, this paper addresses the limitations of both individual methods. MFAC provides adaptability, while SMC ensures robustness. This integrated approach offers a comprehensive solution to the challenges of robust speed regulation in multi-agent systems, as demonstrated in [7] - [11].

Recent studies have explored the application of MFAC and SMC for various control problems in multi-agent systems. In [1], MFAC was used for consensus control in multi-agent systems with unknown dynamics. In [5], SMC was employed for

fault-tolerant control in multi-agent systems. The combination of MFAC and SMC has been shown to be effective in addressing complex control challenges, such as those encountered in multi-DC motor systems.

This paper utilizes the implementation of a fixed topology using Laplacian matrices, which effectively handle the interconnections between multiple agents (DC Motors) and maintain precise desired speed control. Encoder counts are employed as feedback to ensure high-resolution performance monitoring, thereby reinforcing the accuracy and reliability of the control system.

In summary, the proposed data-driven model-free adaptive sliding mode control strategy is expected to outperform traditional methods in terms of accuracy, stability, and robustness based on the MFAC method [2]. By leveraging the advantages of MFAC and SMC, this approach can enhance the performance and reliability of multi-DC motor speed regulation in various applications, such as autonomous vehicles, robotics, and industrial automation.

The following sections will outline the remaining content of this paper: section 2 provides Preliminaries and problem formulation, Section 3 The main results, Section 4 presents simulation results and performance analysis , demonstrating the efficacy of the proposed method under various operating conditions. At the end, Section 5 concludes the paper, summarizing the key findings potential points for future research, including the extension of the proposed approach to other types of multi-agent systems and exploring further enhancements to the control algorithms.

II. Preliminaries and problem formulation

A. Preliminaries

The set of real numbers is denoted by $\mathbb{R}$. For a given matrix $A \in \mathbb{R}^{n\times n}$, $\|A\|$ represents the matrix norm. The notation $\text{diag}(\cdot)$ refers to a diagonal matrix, and I signifies the identity matrix of appropriate dimensions. In the context of multi-agent systems, graph theory serves as an effective tool to model interaction topologies. A brief introduction to directed graphs within algebraic graph theory is provided. Let $G = (V, E, A)$ be the weighted directed graph, where $V = \{1, 2, \ldots, N\}$ represents the set of vertices, $E \subseteq V \times V$ denotes the set of edges, and A is the adjacency matrix. Here, V also indexes the agents. If agent j can receive a message from agent i, then $(i, j) \in E$, where j is the child of i . The neighborhood of agent i is given by $N_i = \{j \in V|(j, i) \in E\}$.

The weighted adjacency matrix $A = (a_{i,j})$ is defined such that $a_{i,i} = 0$, $a_{i,j} = 1$ if $(j, i) \in E$; otherwise, $a_{i,j} = 0$. The Laplacian matrix of G is defined as $L = D - A$, where $D = \text{diag}(d_1^{\text{in}}, d_2^{\text{in}}, \ldots, d_N^{\text{in}})$ and $d_i^{\text{in}} = \sum_{j=1}^{N} a_{i,j}$ is the in-degree of vertex i. The graph is said to be strongly connected if there exists a path between any pair of vertices.

III. Problem Formulation

In this research, the speed regulation problem of DC motors is frequently examined under the assumption that all motors demonstrate identical dynamic characteristics. However, heterogeneity remains a fundamental characteristic of systems incorporating multi-DC motors. Even when motors are of the same type and share similar structural features, the parameters can never be exactly the same. This inherent variability complicates coordinated speed regulation across heterogeneous motors. Consider the multi-DC motor system consisting of N motors, where the interaction topology is represented by G. Assume that each motor i follows the nonlinear dynamics:

$$y_i(k+1) = f_i(y_i(k), u_i(k)), \quad i = 1, 2, \ldots, N \tag{1}$$

where $y_i(k) \in \mathbb{R}$ represents the output (speed of the DC motor), $u_i(k) \in \mathbb{R}$ is the control input (the voltage), and $f_i(\cdot)$ is an unknown nonlinear function, respectively.

In this scenario, multiple agents aim to track the consensus trajectory $y_d(k)$, which is exclusively accessible to the subset of agents. This trsajectory is assumed to be generated by a virtual leader designated as vertex 0. To model this interaction, the construction of the direct graph $G' = (V \cup \{0\}, E', A')$ where V denotes the set of agents, E' represents the edge set defining connections from agents to the virtual leader, and A' constitute a weighted adjacency matrix detailing the above-mentioned connections.

This analysis prosupposes nonlinear dynamics governing the evolution and encompassing dependencies on the respective states and inputs of each agent. The previously-mentioned nonlinear dynamics accomodate various complexities, facilitating an investigation into how agents align with the desired trajectory $y_d(k)$.

Assumption 1: The partial derivative of the nonlinear function $f_i(\cdot)$ with respect to $u_i(k)$ is continuous.

Assumption 2: The model $y_i(k+1) = f_i(y_i(k), u_i(k))$ is generalized lipschitz, meaning that if $\Delta u_i(k) = u_i(k) - u_i(k-1) \neq 0$, then $|\Delta y_i(k+1)| \leq b|\Delta u_i(k)|$ holds for any k, where $\Delta y_i(k+1) = y_i(k+1) - y_i(k)$ and b is a positive constant.

Remark 1: The practical applicability of the aforementioned assumptions to nonlinear systems has been extensively discussed in [1], affirming their suitability for practical multi-agent systems. Assumption 1 establishes a foundational criterion for controller design, ensuring the continuity of the partial derivative of the nonlinear function $f_i(\cdot)$ with respect to $u_i(k)$. Assumption 2 implies that the rate of change in the output of an agent in response to changes in the control input is bounded. This constraint ensures that, from an energetic perspective, finite changes in control input energy correspond to bounded changes in output energy rates, a crucial consideration for system stability and performance.

A. Linearization Technique

This paper delves into the CFDL technique, this method simplifies the dynamic of nonlinear systems into the linear form that is easier to handle and control. The CFDL is particularly useful when the control input $u_i(k) = 0$ holds, enabling the system to be described through the compact dynamic linearization model. The system under consideration is governed by the following equation:

$$\Delta y_i(k+1) = \phi_i(k)\Delta u_i(k) \tag{2}$$

where $|\phi_i(k)| \leq b$ with the variable $\phi_i(k)$ named PPD.

We define the distributed measurement output of $\xi_i(k)$ for ith agents as follow:

$$\xi_i(k) = \sum_{j \in N_i} a_{ij}(y_j(k) - y_i(k)) + d_i(y_d(k) - y_i(k)) \tag{3}$$

In this equation $\xi_i(k)$ represents the distributed measurement output of the ith agent at time step k , N_i denotes the set of the neighboring agents of the ith agent and a_{ij} are elements of the adjancecy matrix representing the weights between agents i and j. For this equation, we assume that $d_i = 0$, meaning the agent i doesn't directly consider the desired trajectory $y_d(k)$ in the distributed measurement output. However, if agent i can receive the disired trajectory, setting d_i to a non-zero value allows it to align the output with the desired trajectory, improving the control performance.

Let $e_i(k) = y_d(k) - y_i(k)$ denote the tracking error. The objective of this paper is to find an good control law using I/O data of the agents (DC motors), such that the outputs of all agents can track the reference trajectory $y_d(k)$ when only some of agents can access the desired trajectory.

Assumption 3: The communication graph $\bar{G}$ is fixed and strognly connected, with at least one follower agent able to access the trajectory of the leader.

Remark 2: This assumption ensures the solvability the tracking problem. An isolated agent, oblivious of the control objective, cannot follow the leader's reference trajectory.

Assumption 4 : The PPD $\phi_i(k) > \varsigma$, $i = 1, 2, 3, \ldots, N$ holds for all k, where ς is an rondomly small positive constant without loss of generality, assume that $\phi_i(k) > \varsigma$.

Remark 3: This indicatesthat the agent output does not decrease with encreasing control input, resembling a linear characteristic. It implies the control direction is known and unchanging Similar assumptions are common in model-based control and are reasonable for practical systems like mobile robots and UAVs.

IV. Main Results

Consider the following PPD criterion function, which is used to evaluate the performance of the parameter $\phi_i(k)$:

$$J(\phi_i(k)) = |\Delta y_i(k) - \hat{\phi}_i(k)\Delta u_i(k-1)|^2 + \mu|\hat{\phi}_i(k) - \hat{\phi}_i(k-1)| \tag{4}$$

In the previous function, $\Delta y_i(k)$ represents the change in the output, $\hat{\phi}_i(k)$ is the estimated parameter at time k and μ the regularization parameter that penalizes the deviation of the parameter estimate from the previous value.

Differentiating equation (4) with respect to PPD parameter $\phi_i(k)$ and make it equal to zero:

$$\frac{\partial J(\phi_i(k))}{\partial \phi_i(k)} = |\Delta y_i(k) - \hat{\phi}_i(k)\Delta u_i(k-1)|^2 + \mu|\hat{\phi}_i(k) - \hat{\phi}_i(k-1)| = 0 \tag{5}$$

And, let's compute the derivative of $J(\hat{\phi}_i(k))$, we obtain:

$$2[\Delta y_i(k) - \phi_i(k-1)] - [\Delta u_i(k-1)] + 2\mu[\phi_i(k) - \hat{\phi}_i(k-1)] = 0 \tag{6}$$

This equation provides a necessary condition for minimizing the criterion function $J(\phi_i(k))$ and based on this optimization criterion, the following update rule is proposed for the distributed MFAC algorithm:

$$\hat{\phi}_i(k) = \hat{\phi}_i(k-1) + \frac{\eta\Delta u_i(k-1)(\Delta y_i(k) - \hat{\phi}_i(k-1)\Delta u_i(k-1))}{\mu + \Delta u_i(k-1)} \tag{7}$$

Here, η is the learning rate that controls the step size of the update. Additionaly, to ensure stability and adherence to constraints, the following condition is applied and helps in maintaining the validity of the parameter and avoiding drastic changes.

$$\hat{\phi}_i(k) = \hat{\phi}_i(1), if|\hat{\phi}_i(k)| \leq \epsilon \ or \ sign(\hat{\phi}_i(k)) \neq sign(\hat{\phi}_i(1)) \tag{8}$$

A. Model Free Adaptiive Controller Design

To design the MFAC alogrithm, the performance function $J(u_i(k))$ is set as:

$$J(u_i(k)) = |\xi_i(k)|^2 + \lambda|u_i(k) - u_i(k-1)|^2 \tag{9}$$

This function used to evaluate the effectiveness of the control input $u_i(k)$ for the ith agent by considering two terms. The first term $|\xi_i(k)|^2$, represents the tracking error and ensures that the output of agent closely follows the desired trajectory. The second term$\lambda|u_i(k) - u_i(k-1)|^2$ where $u_i(k)$ is the current control input, and $u_i(k-1)$ is the previous control input, with λ serving as a weighting factor.

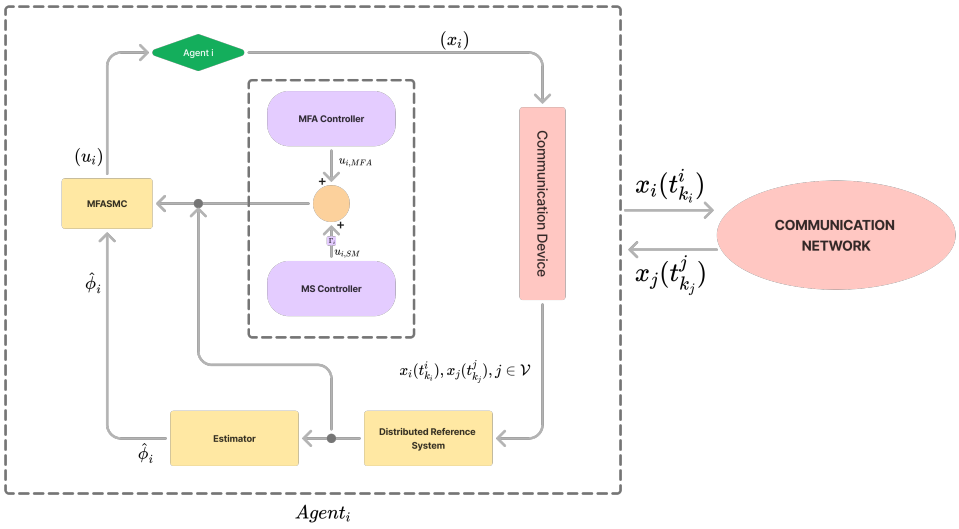

Fig. 1: Block diagram of the data-driven MFASMC control.

Subtituting (2) and (3) into (9), then differentiating (9) with respect to $u_i(k)$, and make it equal to zero, gives:

$$\mathbf{u}_{\mathbf{i}\text{MFA}}(k) = \mathbf{u}_{\mathbf{i}\text{MFA}}(k-1) + \frac{\rho\phi_i(k)}{\lambda + |\phi_i(k)|^2}\xi_i(k) \tag{10}$$

where $\rho \in$ (0,1) is a step-size constant, which is added to make (10) general. Using the parameter estimation algorithm (7) and the control law algorithm (10), the MFAC scheme is constructed.

B. Sliding Mode Controller Design

To design the SMC for system, the sliding mode surface is first defined , guiding the behavior of system to ensure robust and accurate tracking of the desired trajectory.

The sliding mode surface is defined as:

$$S_i(k+1) = S_i(k) + e_i(k+1) + \alpha e_i(k) \tag{11}$$

where $S_i(k)$ represents the sliding surface at the current time step k, α is a positive constant.

To ensure that the system trajectory is driven toward and remains on the sliding surface. The reaching law dictates how quickly the system state converges to the sliding surface and is given by:

$$\Delta S_i(k+1) = -\varepsilon T Sign(k) \tag{12}$$

In that equation, ε is a small positive constant that controls the rate of the convergence, T is the sampling period, and $sign(k)$ indicates the direction in which the system should move to reach the sliding surface.

By combining the sliding surface definition and the reaching law, derivating the control law that ensures the desired tracking performance while maintaining robustness.

The final sliding mode control input $\mathbf{u}_{\mathbf{i}\text{SM}}(k)$ is designed:

$$\mathbf{u_{iSM}}(k) = u_i(k-1) + \frac{y_d(k+1) - y(k) + \alpha e_i(k) + \varepsilon T sign(k)}{\phi_i(k)} \tag{13}$$

To enhance the robustness and adaptibility of the control system, the MFASMC approach is employed. the conrol input of the system is defined as:

$$u_i(k) = \mathbf{u_{iMFA}}(k) + \Gamma_i \mathbf{u_{iSM}} \tag{14}$$

where the parameter Γ is a gain factor that adjusts the contribution of the sliding mode control in the control effort and tunes the convergence rate.

C. Stability Analysis

The stability analysis is conducted in two primary steps. The first step focuses on estabilishing the bounds, the second step ensures that the error remains within acceptable limits over time, leading to a stable system.

Step 1: Establishment of error bounds

The error between the estimated and actual values of the system parameters, denote as $\tilde{\phi}_i(k) = \phi_i(k) - \hat{\phi}_i(k)$, starting from the foundational equation derived from the compact dynamic linearization model in equation (2) along with the PPD estimation equation (7).To ensure the theorem's validity, the detailed proof that demonstrates the correctness of this error bound is provided.

$$\tilde{\phi}_i(k) = \hat{\phi}_i(k+1) + \frac{\eta \Delta u_i(k-1)}{\mu + |\Delta u_i(k-1)|^2}((\Delta y_i(k) - \hat{\phi}_i(k-1)\Delta u_i(k-1))) - \phi_i(k) \tag{15}$$

Then by simplifying the previous equation, the result is:

$$\tilde{\phi}_i(k) = (\hat{\phi}_i(k+1) - \phi_i(k-1)) + \frac{\eta \Delta u_i(k-1)}{\mu + |\Delta u_i(k-1)|^2}((\Delta y_i(k) - \hat{\phi}_i(k-1)\Delta u_i(k-1))) - (\phi_i(k) - \phi_i(k-1)) \tag{16}$$

The control coefficient $\beta_i(k)$ is expressed from the previous equation, plays a critical role in adjusting the control input for each agent at time step k. The following equation is expressed:

$$\tilde{\phi}_i(k) = \tilde{\phi}_i(k-1) + \beta_i(k)(\Delta y_i(k) - \hat{\phi}_i(k-1)\Delta u_i(k-1) - \phi_i(k) - \phi_i(k-1)) \tag{17}$$

$$\tilde{\phi}_i(k) = (1 - \frac{\eta(\Delta u_i(k-1))^2}{\mu + |\Delta u_i(k-1)|^2})\tilde{\phi}_i(k-1) + \phi_i(k-1) - \phi_i(k) \tag{18}$$

$$\tilde{\phi}_i(k) = (1 - \frac{\eta(\Delta u_i(k-1))^2}{\mu + |\Delta u_i(k-1)|^2})\tilde{\phi}_i(k-1) - \Delta\phi_i(k) \tag{19}$$

To demonstrate the boundedness of the error, by taking the absolute value of both sides of the error (19). This is a crucial step, as it allows us to establish an inequality that provides an upper bound on the error term.

Taking the absolute value on both sides and applying the triangle inequality to the right-hand side, it follows that:

$$|\tilde{\phi}_i(k)| \leq \left|1 - \frac{\eta(\Delta u_i(k-1))^2}{\mu + |\Delta u_i(k-1)|^2}\right| |\tilde{\phi}_i(k-1)| + |\Delta\phi_i(k)| \tag{20}$$

Defining:

$$\alpha(k-1) = \frac{\eta(\Delta u_i(k-1))^2}{\mu + |\Delta u_i(k-1)|^2} \tag{21}$$

So equation (23) becomes:

$$|\tilde{\phi}_i(k)| \leq |1-\alpha(k-1)||\tilde{\phi}_i(k-1)| + |\Delta\phi_i(k)| \tag{22}$$

Assumption 4: We assume that the control input increment $\Delta u_i(k)$ is always non-zero, i.e., $\Delta u_i(k) \neq 0$, at every time step k. This assumption ensures that the system continues to receive meaningful input changes, which are necessary for the adaptive control mechanism to function.

Moreover, the parameters involved in the formulation of $\alpha(k-1)$ are bounded as follows: the tuning parameter η is constrained within the range $0 < \eta \leq 1$, ensuring that the influence of $\Delta u_i(k-1)$ is appropriately scaled. The constant μ is assumed to satisfy $\mu \geq 0$, which provides numerical stability by preventing division by extremely small values.

Under these conditions, the term $\alpha(k-1)$, which governs the scaling of the control input, is bounded such that $0 < \alpha(k-1) \leq q_1 < 1$, where q_1 is a positive constant. This constraint on $\alpha(k-1)$ ensures that it remains strictly less than 1, preventing excessive correction in the control input. Consequently, $\alpha(k-1)$ effectively moderates the magnitude of the corrective actions applied to the system, thereby ensuring stability and preventing oscillations or instability that could arise from overly aggressive control updates.

By combining the inequalities and applying an iterative process, represents the change as follows:

$$|\tilde{\phi}_i(k)| \leq |1-q_1||\tilde{\phi}_i(k-1)| + b \tag{23}$$

Continuing this process for previous time steps::

$$|\tilde{\phi}_i(k-1)| \leq |1-q_1||\tilde{\phi}_i(k-2)| + b \tag{24}$$

Back to the initial condition at $k = 0$ and summing the resulting geometric series:

$$\sum_{j=0}^{k-1}(1-q_1)^j = \frac{1-(1-q_1)^k}{q_1}$$

Thus:

$$|\tilde{\phi}_i(k)| \leq (1-q_1)^k|\tilde{\phi}_i(0)| + \frac{b}{q_1}(1-(1-q_1)^k) \tag{25}$$

As $k \to \infty$, the term $(1-q_1)^k$ tends to zero, simplifying the bound to:

$$|\tilde{\phi}_i(k)| \leq \frac{b}{q_1} \tag{26}$$

Remark 4: The inequality in (26) shows that $\tilde{\phi}_i(k)$ is bounded by $\frac{b}{q_1}$. This result implies that the error $\tilde{\phi}_i(k)$ will remain within this bound, even as k approaches infinity. Thus, the system error behavior is effectively controlled and constrained.

Part 2: In this part, delving into the convergence properties of the tracking error $\xi_i(k)$. To understand the convergence, we start with the expression for $\xi_i(k)$ as given in equation (3). We aim to express $\xi_i(k)$ in terms of the individual tracking errors $e_i(k)$ and the errors associated with neighboring systems.

The expression for $\xi_i(k)$ is formulated as follows:

$$\xi_i(k) = \sum_{j \in N_i} (e_i(k) - e_j(k)) + d_i e_i(k) \tag{27}$$

In this equation, $\xi_i(k)$ represents the tracking error of the i-th system at time k, taking into account the deviations from its neighbors j in the set N_i and an additional term $d_i e_i(k)$ that depends on the specific characteristics of the i-th system. The parameter d_i is a diagonal matrix term that adjusts the impact of the individual error $e_i(k)$ on the overall error $\xi_i(k)$.

To facilitate analysis, we define the following vectors that aggregate the tracking errors, outputs, and control inputs for all systems:

$$y(k) = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{pmatrix}, \quad e(k) = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_n \end{pmatrix}, \quad \xi(k) = \begin{pmatrix} \xi_1 \\ \xi_2 \\ \xi_3 \\ \vdots \\ \xi_n \end{pmatrix}, \quad u(k) = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_n \end{pmatrix}$$

Using these definitions, the expression for $\xi_i(k)$ can be compactly rewritten as:

$$\xi_i(k) = (L + D)e(k) \tag{28}$$

Here, L represents the interaction matrix that describes how each system interacts with its neighbors, while D is a diagonal matrix defined by $D = \text{diag}(d_1, d_2, d_3, \ldots, d_n)$. This formulation helps us to analyze how the overall system's tracking error is influenced by the individual tracking errors and the interaction structure.

Next, the control input $u(k)$ is updated according to the following rule:

$$u(k) = u(k-1) + \rho \cdot \text{diag}\left(\frac{\hat{\phi}_1(k)}{\lambda + |\hat{\phi}_1(k)|^2}, \frac{\hat{\phi}_2(k)}{\lambda + |\hat{\phi}_2(k)|^2}, \frac{\hat{\phi}_3(k)}{\lambda + |\hat{\phi}_3(k)|^2}, \ldots, \frac{\hat{\phi}_n(k)}{\lambda + |\hat{\phi}_n(k)|^2} \right) (L + D)e(k)$$

This update rule accounts for the adjustment of control inputs based on the estimation $\hat{\phi}_i(k)$ of the parameters, where λ is a regularization parameter that ensures stability. This formulation leads to:

$$u(k) = u(k-1) + \rho H_1(k)(L + D)e(k) \tag{29}$$

where $H_1(k)$ is a diagonal matrix whose elements are functions of the estimated parameters $\hat{\phi}_i(k)$. The evolution of the output $y(k)$ is described by:

$$y(k+1) = y(k) + \phi(k)\Delta u(k)$$

In this context, $\Delta u(k)$ represents the change in the control input:

$$\Delta u(k) = u(k) - u(k-1)$$

and $H_\phi(k)$ is defined as:

$$H_\phi(k) = \text{diag}(\phi_1(k), \phi_2(k), \phi_3(k), \ldots, \phi_n(k))$$

By substituting the expression for $u(k)$ from equation (29) into the output evolution equation, we obtain:

$$e(k+1) - e(k) = y(k) - y(k+1)$$

Rearranging the terms, we get:

$$e(k+1) = e(k) + y(k) - y(k+1)$$

Substituting the previously derived expression for $y(k+1)$, we have:

$$e(k+1) = e(k) - H_\phi(k)\rho H_1(k)(L+D)e(k)$$

This can be rewritten as:

$$e(k+1) = (I - \rho\Theta(k))e(k) \tag{30}$$

where $\Theta(k)$ is defined by:

$$\Theta(k) = \sum(k)(L+D) = H_\phi(k)H_1(k)(L+D)$$

with:

$$\sum(k) = \text{diag}\left(\frac{\hat{\phi}_1(k)}{\lambda + |\hat{\phi}_1(k)|^2}, \frac{\hat{\phi}_2(k)}{\lambda + |\hat{\phi}_2(k)|^2}, \frac{\hat{\phi}_3(k)}{\lambda + |\hat{\phi}_3(k)|^2}, \ldots, \frac{\hat{\phi}_n(k)}{\lambda + |\hat{\phi}_n(k)|^2}\right)$$

which can be expressed in diagonal form as:

$$\sum(k) = \text{diag}(\mathcal{V}_1(k), \mathcal{V}_2(k), \mathcal{V}_3(k), \ldots, \mathcal{V}_n(k))$$

where each $\mathcal{V}_i(k)$ is given by:

$$\mathcal{V}_i(k) = \frac{\hat{\phi}_i(k)}{\lambda + |\hat{\phi}_i(k)|^2}, \quad i = 1, 2, 3, \ldots, n$$

Thus, we have:

$$\Theta(k) = \text{diag}(\mathcal{V}_1(k), \mathcal{V}_2(k), \mathcal{V}_3(k), \ldots, \mathcal{V}_n(k))(L+D)$$

To ensure convergence of the tracking error, we impose the condition:

$$\|I - \rho\Theta(k)\| < 1 \tag{31}$$

This condition guarantees that the tracking error $e(k)$ will approach zero as $k \to \infty$. Consequently, we have:

$$\lim_{k\to\infty} \|e(k+1)\| = 0$$

This result indicates that the tracking error will converge to zero, demonstrating the effectiveness of the proposed control strategy in achieving desired tracking performance.

V. Quadriple-Frequency Data Processing Method for DC Motors Speed Regulation

To regulate the speed of DC motors accurately, we employ a sophisticated data-driven binary containment control method, which requires precise measurement and processing of encoder data. This section describes the methodology and hardware used for speed regulation, including the calculation of motor speed using the quadriple-frequency data processing (QFDP) method.

A. Hardware Implementation

The DC brushed motors have a rated voltage of 12V, an unloaded speed of 293 ± 21 RPM, and a rated current of 0.36 A. The gear ratio of 20 means that the output speed of the motor is 1/20 of the rotor speed, resulting in higher torque with a higher gear ratio. The Hall encoders used have 13 pulses per revolution (ppr), meaning each full rotation generates 13 pulse signals.

To enhance measurement accuracy, we employed a quadruple-frequency data processing method. This technique quadruples the effective resolution of the encoder by processing the output pulse signals at four times the frequency, thus increasing measurement precision by a factor of four.

B. Data Processing Method

The motor speed is measured in revolutions per second (r/s). The following equations are used to calculate the speed based on encoder measurements and sampling:

1. Calculation of Rounds:

The total number of ”rounds” that the encoder measures is given by:

$$T = \text{Encoder line count} \times \text{Reduction Ratio} \times 4 \tag{32}$$

Here, the Encoder line count is 13, and the Reduction Ratio is 20. The factor of 4 accounts for the quadrature encoding, which effectively quadruples the resolution.

2. Calculation of Number of Rotations:

The number of rotations can be determined using:

$$\text{Number of Rotations} = \frac{m}{\text{Rounds Per T}} \tag{33}$$

Where m is the total count from the encoder, and Rounds Per T is the number of encoder counts per revolution, derived from Equation (33).

3. Calculation of Speed:

The speed of the motor in resolution per second (r/s) is given by:

$$v = \frac{\text{Number of Rotations}}{T} \tag{34}$$

Here, v represents the speed, calculated by dividing the number of rotations by the time T it takes to complete those rotations.

4. Combining the Equations:

Substituting Equation (34) into Equation (35), we obtain:

$$v = \frac{m}{\text{Rounds Per T} \times T} \tag{35}$$

This final equation integrates encoder measurements and the sampling period to provide an accurate speed regulation formula using the QFDP method.

In our system, the sampling period, which is set by a timer, defines how frequently speed measurements are taken. Each sampling interval triggers an interrupt where the controller samples the motor speed and updates control commands accordingly.

The quadruple-frequency method, implemented through software, is crucial for maximizing encoder measurement precision, resulting in more accurate speed control for the motor system.

VI. Simulation Results

Assumption 5: The communication graph $\bar{\mathcal{G}}_l$ is a fixed strongly connected graph and at least one of the follower agents can access the leader's trajectory for all $l = 1, 2, \ldots, M$.

In this simulation results, we perform numerical simulations to illustrate the proposed speed tracking results for fixed communication topology. Consider a network comprising

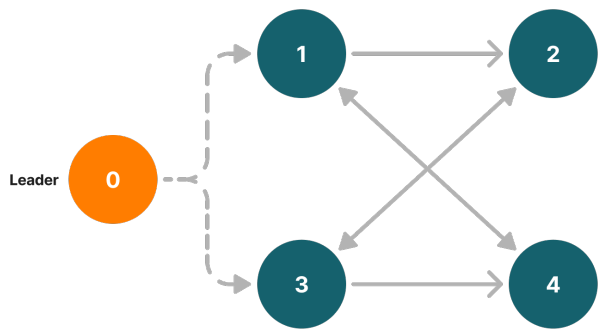

Fig. 2: Communication topology among agents (DC Motors).

Four follower DC motors and the models for each DC motor governed by:

DC Motor 1 (Agent 1): $y_1(k+1) = \frac{m}{rT \times 0.1} \times u_1(k)$

DC Motor 2 (Agent 2): $y_2(k+1) = \frac{m}{rT \times 0.1} \times u_2(k)$

DC Motor 3 (Agent 3): $y_3(k+1) = \frac{m}{rT \times 0.3} \times u_3(k)$

DC Motor 4 (Agent 4): $y_4(k+1) = \frac{m}{rT \times 0.3} \times u_4(k)$

In this context, m represents the total pulse count from the encoder, which measures the rotation of the motor shaft. The variable rT stands for the number of encoder counts per revolution, indicating how many counts correspond to one full rotation of the motor shaft. The variable T denotes time, which can vary and directly influences the motor's speed depending on the duration of each control input. These models are derived from the expression (35).

It is evident that the agents considered are heterogeneous, as their dynamics differ from one another. In this scenario, the dynamics are assumed to be unknown and are only provided here to generate the I/O data for the multi-agent system. No model information is utilized in the distributed MFSMAC algorithm.

As illustrated in Figure 2, the virtual leader is designated as vertex 0. It can be observed that only agents 1 and 3 can receive information from the leader, forming a strongly connected communication graph. Assume that the information exchange among agents is directed and fixed. The Laplacian matrix of the graph is given as follows:

$$L = \begin{bmatrix} 1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & -1 & 2 \end{bmatrix}$$

with $D = \text{diag}(1, 0, 1, 0)$. The reciprocal of the greatest diagonal entry of $L+D$ is 0.5. By selecting the controller parameters as $\rho = 1$, the convergence condition in Theorem 2 and Theorem 3 is satisfied for all $i = 1, 2, 3, 4$. We then consider the following two distinct desired trajectories.

A. Time Invariable Desired Trajectory

where $L + 1$ accounts for the range from 0 to 100. The expression for $y_d(k)$ is:

$$y_d(k) = 0.5 \cdot \sin\left(\frac{k\pi}{30}\right) + 0.3 \cdot \cos\left(\frac{k\pi}{10}\right)$$

for k in the range $0 \leq k \leq 100$.

The initial parameters are chosen as $u_i(1) = 0.1$, $y_i(1) = 0.1$, $\phi_i(0) = 1$ for all agents in this simulation ,$\Gamma_{1,2} = 0.15$ and $\Gamma_{3,4} = 0.45$,$T = 0.1$,$m = 350$, $\eta = 1$,$\mu = 1$. The MFA Controller parameters are given as $\rho = 1$,$\lambda = 50$ and the SM Controller parameters are $\alpha = 1$ with $\epsilon = 10^-5$.

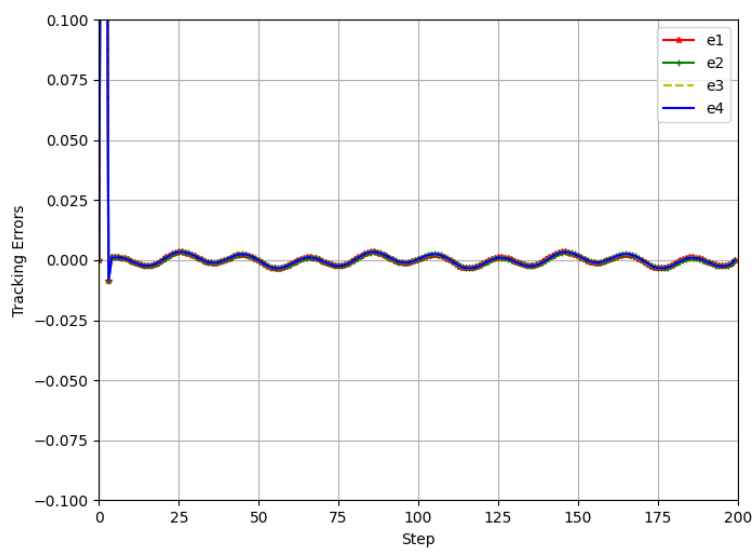

Fig. 3: Tracking Errors for time varying desired trajectory.

As shown in Figure 3, the tracking errors between the actual and desired trajectories for agents a1, a2, a3, and a4 are relatively small and converge to zero over time, indicating satisfactory performance. However, the individual agents exhibit varying levels of tracking error, suggesting that their unique dynamics or initial conditions may influence their performance.

Figure 4 presents a detailed analysis of the tracking performance for all agents. As illustrated, all agents successfully track the time-varying desired trajectory, confirming the effectiveness of the proposed control system. While minor variations in individual trajectories are evident, they generally adhere to the desired path. Factors such as agent dynamics, communication delays, and environmental disturbances could potentially influence the tracking performance.

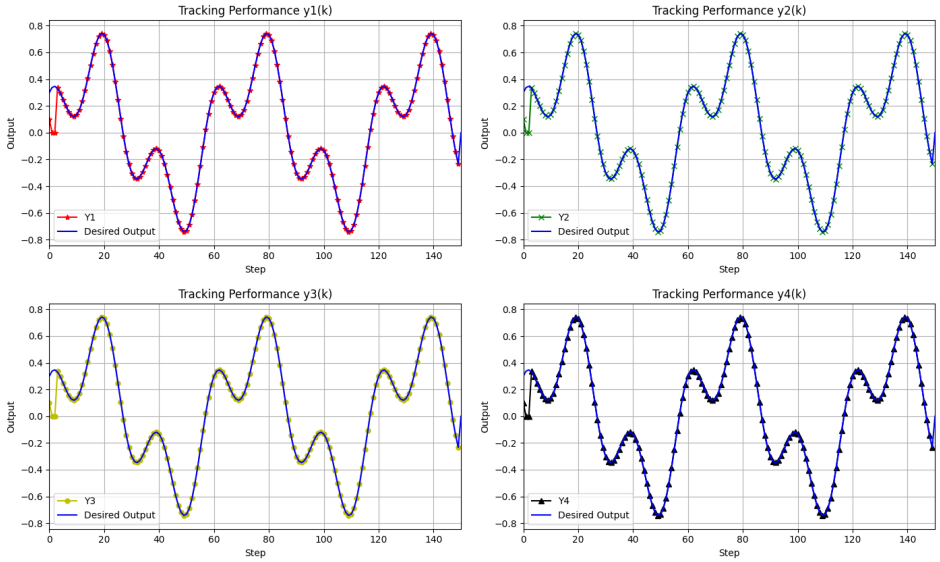

Fig. 4: Tracking performance of all agents for time-varying desired trajectory.

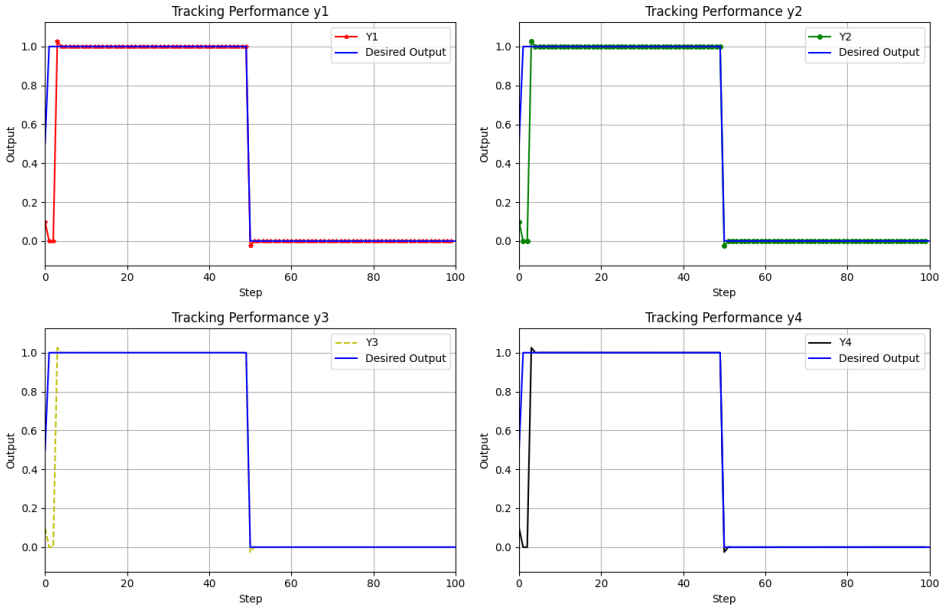

Fig. 5: Tracking performance of all agents for time-invariable desired trajectory.

Figure 5 demonstrates that all agents successfully track the time-invariable desired trajectory, further validating the robustness of the proposed control strategy. Meanwhile, Figure 6 shows the PPD estimation for all agents, highlighting the accuracy of the adaptive estimation process within the control framework.

Overall, the simulation results suggest that the proposed control system is capable of tracking a constant desired trajectory for multiple agents. While there may be initial transient errors, the system eventually reaches a steady-state condition with minimal tracking error. The variations in tracking performance among the agents highlight the potential influence of individual characteristics and external factors.

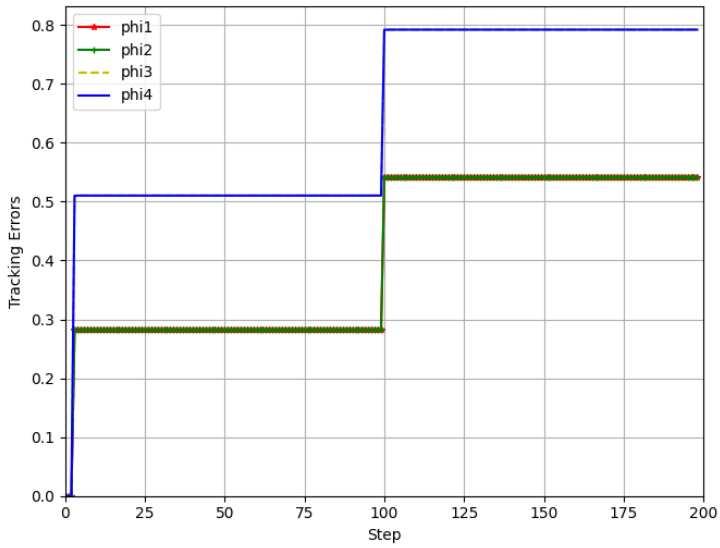

Fig. 6: PPD Estimation of all agents.

VII. Conclusion

This paper introduces a Model-Free Adaptive Sliding Mode Control (MFASMC) system. The proposed approach is designed for the speed regulation of multiple DC motors. Model-Free Adaptive Control (MFAC), combined with Sliding Mode Control (SMC), ensures robust performance despite varying motor characteristics and uncertainties. The system utilizes a fixed topology with Laplacian matrices to effectively handle the interconnections between multiple DC motors and maintain precise speed control. This is demonstrated through the simulation results presented.

References

[1] Bu, XH; Hou, ZS and Zhang, HW. Data-Driven Multiagent Systems Consensus Tracking Using Model Free Adaptive Control. *IEEE Transactions on Neural Networks and Learning Systems*, May 2018, 29(5): 1514-1524.

[2] Z. Hou and S. Jin, ”A novel data-driven control approach for a class of SISO nonlinear systems,” *IEEE Transactions on Control Systems Technology*, vol. 19, no. 6, pp. 1549-1558, 2011.

[3] R. Olfati-Saber, J. A. Fax, and R. M. Murray, ”Consensus and cooperation in networked multi-agent systems,” *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215-233, 2007.

[4] Wang, XF (Wang, Xiaofeng); Li, X (Li, Xing); Wang, JH (Wang, Jianhui); Fang, XK (Fang, Xiaoke); Zhu, XF (Zhu, Xuefeng), ”*Information Sciences*”, vol. 327, pp. 246-257, January 2016.

[5] W. Ren, R. W. Beard, and E. M. Atkins, ”Information consensus in multivehicle cooperative control,” *IEEE Control Systems*, vol. 27, no. 2, pp. 71-82, 2007.

[6] D. Xu, B. Jiang, and P. Shi, ”Adaptive observer based data-driven control for nonlinear discrete-time processes,” *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 4, pp. 1037-1045, 2014.

[7] R. Chi, Z. Hou, S. Jin, D. Wang, and C.-J. Chien, ”Enhanced data-driven optimal terminal ILC using current iteration control knowledge,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 11, pp. 2939-2948, 2015.

[8] H. Zhang and F. L. Lewis, ”Adaptive cooperative tracking control of higher-order nonlinear systems with unknown dynamics,” *Automatica*, vol. 48, no. 7, pp. 1432-1439, 2012.

[9] C. L. P. Chen, G.-X. Wen, Y.-J. Liu, and F.-Y. Wang, ”Adaptive consensus control for a class of nonlinear multiagent time-delay systems using neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1217-1226, 2014.

[10] Zhou, N (Zhou, Ning); Deng, WX (Deng, Wenxiang); Yang, XW (Yang, Xiaowei); Yao, JY (Yao, Jianyong), ”Continuous adaptive integral recursive terminal sliding mode control for DC motors,” *International Journal of Control*, vol. 96, no. 9, pp. 2190-2200, June 2022.

[11] X. Liu, J. Lam, W. Yu, and G. Chen, ”Finite-time consensus of multiagent systems with a switching protocol,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 853–862, Apr. 2016.

[12] Z.-G. Hou, L. Cheng, and M. Tan, ”Decentralized robust adaptive control for the multiagent system consensus problem using neural networks,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 3, pp. 636–647, Jun. 2009.

[13] Z. Hou and W. Huang, ”The model-free learning adaptive control of a class of SISO nonlinear systems,” *Proceedings of the American Control Conference*, Albuquerque, NM, USA, Jun. 1997, pp. 343–344.

[14] H. Su, G. Chen, X. Wang, and Z. Lin, ”Adaptive second-order consensus of networked mobile agents with nonlinear dynamics,” *Automatica*, vol. 47, no. 2, pp. 368–375, Feb. 2011.

[15] D. Xu, B. Jiang, and P. Shi, ”Adaptive observer based data-driven control for nonlinear discrete-time processes,” *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 4, pp. 1037–1045, Oct. 2014.

[16] H. Zhang and F. L. Lewis, ”Adaptive cooperative tracking control of higher-order nonlinear systems with unknown dynamics,” *Automatica*, vol. 48, no. 7, pp. 1432–1439, Jul. 2012.

[17] D. Meng, Y. Jia, J. Du, and J. Zhang, ”On iterative learning algorithms for the formation control of nonlinear multi-agent systems,” *Automatica*, vol. 50, no. 1, pp. 291–295, Jan. 2014.

[18] A. Sharafian, V. Bagheri, and W. Zhang, ”RBF neural network sliding mode consensus of multi-agent systems with unknown dynamical model of leader-follower agents,” *International Journal of Control, Automation and Systems*, vol. 16, no. 2, pp. 749–758, 2018.

[19] X. Ma, F. Sun, H. Li, and B. He, ”Neural-network-based integral sliding-mode tracking control of second-order multi-agent systems with unmatched disturbances and completely unknown dynamics,” *International Journal of Control, Automation and Systems*, vol. 15, no. 4, pp. 1925–1935, 2017.

[20] R. Rahmani, H. Toshani, and S. Mobayen, ”Consensus tracking of multi-agent systems using constrained neural-optimiser-based sliding mode control,” *International Journal of Systems Science*, vol. 51, no. 14, pp. 2653–2674, 2020.

[21] Z. Peng, G. Wen, A. Rahmani, and Y. Yongguang, ”Distributed consensus-based formation control for multiple nonholonomic mobile robots with a specified reference trajectory,” *International Journal of Systems Science*, vol. 46, no. 8, pp. 1447–1457, 2015.