Mugla Journal of Science and Technology

ADVANCED ADAPTIVE CONTROL STRATEGIES APPLICATION FOR SPEED REGULATION OF A 12V SMALL DC GEARED MOTOR

Gökhan ÇETİN*, Department of Electrical and Electronics Engineering, Gümüşhane University, Gümüşhane, TÜRKİYE, gokhancetin@gumushane.edu.tr
(https://orcid.org/ORCID: 0000-0002-7960-1217)

Received: 06.08.2025, Accepted: 07.11.2025 Research Article
*Corresponding author DOI: 10.22531/muglajsci.1759444

Abstract

This paper investigates the implementation and performance of adaptive control techniques for a 12V small geared DC motor characterized by modeling errors and input disturbances. This paper discusses the following two primary approaches: Adaptive Radial Basis Function Neural Network (ARBFNN) Controllers and Model Reference Adaptive Control (MRAC). In model uncertainty, MRAC and ARBFNN outperformed the simple Proportional-Integral (PI) controller. The study is further expanded to involve Robust MRAC and Adaptive Sliding Mode Radial Basis Function Neural Network (ASRBFNN) Controllers to counter the compounded effects of model uncertainty and input disturbances. The versions of the robust controllers performed better than the conventional PI controller in cases involving both uncertainties and disturbances. Implementations were done on a 12V geared DC motor testbed with an Arduino microcontroller and MATLAB's System Identification Toolbox. The results from simulations and experimental applications highlight the greater flexibility and disturbance rejection capability of the developed advanced adaptive control schemes, making them perform better than standard PI controllers under challenging conditions.

Keywords: Model-based adaptive control, Adaptive sliding mode control, DC motor, Speed control, Unknown system parameters

12V KÜÇÜK DC DİŞLİ MOTORUNUN HIZ DÜZENLEMESİ İÇİN GELİŞMİŞ UYARLANABİLİR KONTROL STRATEJİLERİ UYGULAMASI

Özet

Bu makale, modelleme hataları ve giriş bozuklukları ile karakterize edilen 12 V küçük dişlili DC motor için uyarlamalı kontrol tekniklerinin uygulanmasını ve performansını araştırmaktadır. Bu makale aşağıdaki iki temel yaklaşımı tartışmaktadır: Uyarlamalı Radyal Baz Fonksiyonlu Sinir Ağı (ARBFNN) Denetleyicileri ve Model Referanslı Uyarlamalı Kontrol (MRAC). Model belirsizliğinde, MRAC ve ARBFNN basit Oransal-İntegral (PI) denetleyiciden daha iyi performans göstermiştir. Çalışma, model belirsizliğinin ve giriş bozukluklarının bileşik etkilerini dengelemek için Gürbüz MRAC ve Uyarlamalı Kayan Modlu Radyal Baz Fonksiyonlu Sinir Ağı (ASRBFNN) Denetleyicilerini içerecek şekilde daha da genişletilmiştir. Gürbüz denetleyicilerin versiyonları, hem belirsizlik hem de bozukluk içeren durumlarda geleneksel PI denetleyicisinden daha iyi performans göstermiştir. Uygulamalar, bir Arduino mikrodenetleyici ve MATLAB'ın Sistem Tanımlama Aracı ile bir 12 V dişlili DC motor test ortamında gerçekleştirilmiştir. Simülasyon ve deneysel uygulamalardan elde edilen sonuçlar, geliştirilen ileri adaptif kontrol şemalarının daha fazla esneklik ve bozulmayı reddetme kabiliyetine sahip olduğunu ve bu sayede zorlu koşullar altında standart PI kontrolörlerinden daha iyi performans gösterdiğini ortaya koymaktadır.

Anahtar Kelimeler: Model tabanlı adaptive control, Adaptif kayan kipli kontrol, DC motor, Hız kontrolü, Bilinmeyen sistem parametreleri

Cite

Çetin, G., (2025). "Advanced Adaptive Control Strategies Application For Speed Regulation of a 12v Small DC Geared Motor", Mugla Journal of Science and Technology, 11(2), 85-95.

1. Introduction

DC motors are found to be used extensively to control robotics, automotive, and industrial systems. DC motors, such as gearmotors, are used extensively in robotics due to their ability to deliver high torque and decent speed and position control. The motors are essential in smooth and controlled motion applications, and thus are a staple in robotic drive and actuator systems. They are most useful in regulating the speed and placement of robot systems with varying loads, providing the desired flexibility and response for intricate maneuvers and operations [1,2]. Proper mathematical models must be used in constructing effective control systems. Various

models can be used in DC motors, even though DC motors are largely nonlinear due to effects like friction, load variation, and electrical behavior [3].

Typically, systems are often approximated during modeling, which subsequently introduces inaccuracies into the model. When a system can be precisely modeled and remains untouched by external disturbances like delay, noise, or fluctuating parameters, conventional controllers with fixed parameters tend to suffice. However, in cases where these disruptions come into play, such as an unknown system model, standard controllers fall short of delivering the desired performance. To counteract the adverse impact of these disruptions and achieve optimal performance, sophisticated control methods like Adaptive control, robust control, and resilient control techniques come into play.

Adaptive control is widely used and an effective control technique for real-time implementation of controlling a plant, which can be both linear and nonlinear, since it is capable of adapting controller parameters to maintain a desired response in scenarios where the parameters of the system are unknown or time-varying [4,5]. Adaptive control can be classified into two primary subcategories: model-based adaptive control and data-driven adaptive control. In model-based adaptive control, e.g., MRAC, the controller and adaptation laws are devised according to the system model. Conversely, in learning-based control, e.g., ARBFNN, which is a subcategory of model-based adaptive control, the controller draws from a system model, but the adaptation mechanism is tailored using data gathered from the system. Data-driven controllers only rely on sensor measurements of the system [6].

The authors in [7] developed an extension theory-based sliding mode controller for brushless DC motors that adaptively adjusts sliding surface gains, achieving faster speed tracking and improved robustness over conventional SMC

Using an actor–critic reinforcement learning agent, the authors in [8] achieved automatic PID gain tuning for DC motor speed control, resulting in superior tracking performance compared to a classical fixed PID controller.

In [9], two adaptive neural architectures were proposed by the authors. These designs were validated using comparative simulations and practical experiments on a turntable servo motor system. The first approach introduces a robust term, dependent on control gain-bound information, to address NN approximation errors. Alternatively, in the second method designed for cases lacking this information, a new NN structure is created. This structure involves updating only a scalar weight to manage unknown nonlinearities, leading to reduced computational expenses.

In [10], an adaptive control method is devised for tracking a DC motor system with a dead zone. This technique effectively incorporates an asymmetric barrier Lyapunov function to ease initial condition constraints. Radial basis function neural networks (RBFNN) approximate unknown functions in the DC system. The study demonstrates the motor's ability to trace a desired path while ensuring signal boundedness.

The main contributions of this paper are as follows:

- Hybrid Adaptive Framework: By combining Model Reference Adaptive Control (MRAC) and Adaptive Radial Basis Function Neural Network (ARBFNN), this article proposes a new hybrid adaptive control framework for a 12V geared DC motor operating under model uncertainty and input disturbance conditions.
- Robust Adaptive Extension: The proposed structure is extended to Robust MRAC, including σ- and e-modifications, as well as Adaptive Sliding-Mode RBFNN (ASRBFNN), to further enhance the robustness against simultaneous parameter variations and external disturbances.
- Real-time Hardware Implementation: Unlike in most of the previous studies that remained in a simulation stage, the proposed controllers are experimentally implemented on a real 12V geared DC motor testbed using an Arduino microcontroller together with the MATLAB System Identification Toolbox.
- Performance Verification: Experimental and simulation results demonstrate that the proposed adaptive and robust controllers significantly outperform conventional PI controllers in terms of settling time, overshoot, and disturbance rejection in practical applications.

The remaining content of this paper is structured as outlined below: Section 2 presents the modeling of a DC motor. In Section 3, the MRAC and Robust MRAC designs are introduced. The design of ARBFNN is detailed in Section 4, and ASRBFNN is detailed in Section 5. The validation of the proposed approaches through simulations and implementation is discussed in Section 6. Section 7 contains concluding remarks.

2. DC Motor Model

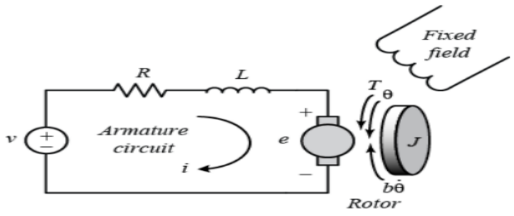

Figure 1. Electric equivalent circuit of the armature and rotor [11].

Dynamics of the DC Motor given in Figure 1 is represented by the equations given below [12],

$$J\ddot{\theta}(t) + b\dot{\theta}(t) = Ki(t) \quad (1)$$

$$L\dot{i}(t) + Ri(t) = V(t) - K\dot{\theta}(t) \quad (2)$$

where, θ is the angular position of the motor, i is the armature current of the motor, and V is the applied input voltage.
The physical parameters of the motors are,
J: moment of inertia of the rotor $(kg \times m^2)$
b: motor viscous friction constant $(N \times m \times s)$
K: electromotive force constant and motor torque constant $(V/rad/sec)$, $(N\frac{m}{Amp})$
R : electric resistance (Ohm)
L : electric inductance (H)

Taking the Laplace transform of Equations (1) and (2)

$$s(Js + b)\theta(s) = KI(s) \quad (3)$$

$$(Ls + R)I(s) = V(s) - Ks\theta(s) \quad (4)$$

$I(s) = \frac{V(s)-Ks\theta(s)}{(Ls+R)}$ is obtained from Equation (4). By replacing the obtained value of $I(s)$ into Equation (3) and taking the inverse Laplace transform, I term is eliminated, and the following differential equation is achieved.

$$JL\ddot{\theta}(t) + (RJ + bL)\ddot{\theta}(t) + bR\dot{\theta}(t) = KV(t) - K^2\dot{\theta}(t) \quad (5)$$

The approximated second-order model of the DC Motor can be obtained below, if the electrical time constant $T_e = \frac{L}{R}$ is much smaller than the mechanical one, the term $JL\ddot{\theta}(t)$ may be neglected. This yields,

$$(RJ + bL)\ddot{\theta}(t) + bR\dot{\theta}(t) = KV(t) - K^2\dot{\theta}(t) \quad (6)$$

Reconstructing Equation (6) we have

$$\ddot{\theta}(t) = -\frac{(K^2 + bR)}{(RJ + bL)}\dot{\theta}(t) + \frac{K}{(RJ + bL)}V(t) \quad (7)$$

Based on the Equations (6) and (7), derivation of the first-order state space equation and the output equation for the system is,

Let $x_1 = \dot{\theta}(t), u = V(t)$

$$\dot{x}_1 = -\frac{(K^2 + bR)}{(RJ + bL)}x_1 + \frac{K}{(RJ + bL)}V(t) \quad (8)$$

which can be written as a state space equation form as,

$$\dot{x} = -\frac{(K^2 + bR)}{(RJ + bL)}x + \frac{K}{(RJ + bL)}u, y = x \quad (9)$$

where $\boldsymbol{x} \in \boldsymbol{\mathcal{R}}$ is the state, $\boldsymbol{u}(t) \in \boldsymbol{\mathcal{R}}$ is the control input, and $\boldsymbol{y} \in \boldsymbol{\mathcal{R}}$ is the measurement.

3. MRAC Design

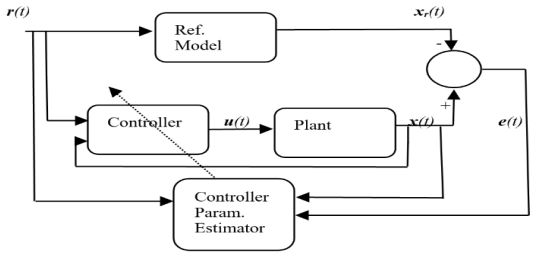

Figure 2. Direct MRAC and Robust MRAC Scheme.

Although the DC motor model introduced in this study is a first-order single-state system, the MRAC design procedure is initially presented in a general Linear Time-Invariant (LTI) form to provide a complete theoretical foundation. This general formulation allows the same adaptive control structure to be directly applicable to higher-order or multi-input systems. In the subsequent subsections, this general MRAC framework is specifically adapted and implemented for the first-order DC motor model defined in Section 2.
Consider a general LTI continuous system,

$$\dot{\boldsymbol{x}}(\boldsymbol{t}) = \boldsymbol{A}\boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{B}\boldsymbol{u}(\boldsymbol{t}) \quad (10)$$

with the measurement equation

$$\boldsymbol{y}(\boldsymbol{t}) = \boldsymbol{C}\boldsymbol{x}(\boldsymbol{t}) \quad (11)$$

where $\boldsymbol{x}(\boldsymbol{t}) \in \mathcal{R}^n$ is the state vector, $\boldsymbol{u}(t) \in \mathcal{R}^p$ is the control input, and $\boldsymbol{y}(t) \in \mathcal{R}^m$ is the measurement vector. $A \in \mathcal{R}^{nxn}$ is the state matrix, $B \in \mathcal{R}^{nxp}$ is the input matrix and $C \in \mathcal{R}^{mxn}$ is the output matrix. Assuming B is known and A is not exactly known.
Consider the LTI continuous-time system reference model.

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}}(\boldsymbol{t}) = \boldsymbol{A}_{\boldsymbol{r}}\boldsymbol{x}_{\boldsymbol{r}}(\boldsymbol{t}) + \boldsymbol{B}_{\boldsymbol{r}}\boldsymbol{r}(\boldsymbol{t}) \quad (12)$$

where $\boldsymbol{x}_{\boldsymbol{r}}(t) \in \mathcal{R}^n$ is the state vector, $\boldsymbol{r}(t) \in \mathcal{R}^p$ is the reference input. $A_{\boldsymbol{r}} \in \mathcal{R}^{nxn}$ is the state matrix, $B_r \in \mathcal{R}^{nxp}$ is the input vector. A_r, B_r are Hurwitz matrix.

Our objective is to design a controller for the DC motor that follows a reference model that gives the desired response. For this purpose, a Direct MRAC design [5] is proposed.
Open loop error dynamic is calculated as;

$$\dot{\boldsymbol{e}}(\boldsymbol{t}) = \dot{\boldsymbol{x}}(\boldsymbol{t}) - \dot{\boldsymbol{x}}_{\boldsymbol{r}}(\boldsymbol{t}) \quad (13)$$

$$\dot{\boldsymbol{e}}(\boldsymbol{t}) = \boldsymbol{A}\boldsymbol{x}(\boldsymbol{t}) + \boldsymbol{B}\boldsymbol{u}(\boldsymbol{t}) - \boldsymbol{A}_{\boldsymbol{r}}\boldsymbol{x}_{\boldsymbol{r}}(\boldsymbol{t}) - \boldsymbol{B}_{\boldsymbol{r}}\boldsymbol{r}(\boldsymbol{t}) \quad (14)$$

Direct MRAC is to be designed as

$$\boldsymbol{u}(\boldsymbol{t}) = \hat{\boldsymbol{k}}_{\boldsymbol{x}}(\boldsymbol{t})\boldsymbol{x}(\boldsymbol{t}) + + \hat{\boldsymbol{k}}_{\boldsymbol{r}}(\boldsymbol{t})\boldsymbol{r}(\boldsymbol{t}) \quad (15)$$

where $\hat{\boldsymbol{k}}_{\boldsymbol{x}}(t) \in \mathcal{R}^{pxn}$, $\hat{\boldsymbol{k}}_{\boldsymbol{r}}(t) \in \mathcal{R}^{pxp}$ are feedback controller gains that are assigned to the controller by a designed online adaptive controller.
Closed loop error dynamics is obtained after replacing Equation (15) into the Equation (14)

$$\dot{\boldsymbol{e}}(\boldsymbol{t}) = \left(\boldsymbol{A} + \boldsymbol{B}\hat{\boldsymbol{k}}_{\boldsymbol{x}}(\boldsymbol{t})\right)\boldsymbol{x}(\boldsymbol{t}) + -\boldsymbol{A}_{\boldsymbol{r}}\boldsymbol{x}_{\boldsymbol{r}}(\boldsymbol{t}) + \boldsymbol{B}\hat{\boldsymbol{k}}_{\boldsymbol{r}}(\boldsymbol{t})\boldsymbol{r}(\boldsymbol{t}) - \boldsymbol{B}_{\boldsymbol{r}}\boldsymbol{r}(\boldsymbol{t}) \quad (16)$$

Assuming matching conditions in the Equation (17) is hold

$$\boldsymbol{A} + \boldsymbol{B}\boldsymbol{k}_{\boldsymbol{x}} = \boldsymbol{A}_{\boldsymbol{r}} \quad \boldsymbol{B}\boldsymbol{k}_{\boldsymbol{r}} = \boldsymbol{B}_{\boldsymbol{r}} \quad (17)$$

Where $\boldsymbol{k}_{\boldsymbol{x}}, \boldsymbol{k}_{\boldsymbol{r}}$ are ideal feedback controller gains, error dynamics becomes,

$$\dot{\boldsymbol{e}}(\boldsymbol{t}) = \boldsymbol{A}_{\boldsymbol{r}}\boldsymbol{e}(\boldsymbol{t}) - \boldsymbol{B}\tilde{\boldsymbol{k}}_{\boldsymbol{x}}(\boldsymbol{t})\boldsymbol{x}(\boldsymbol{t}) - \boldsymbol{B}\tilde{\boldsymbol{k}}_{\boldsymbol{r}}(\boldsymbol{t})\boldsymbol{r}(\boldsymbol{t}) \quad (18)$$

Where $\tilde{\boldsymbol{k}}_{\boldsymbol{x}}(\boldsymbol{t}) \triangleq \boldsymbol{k}_{\boldsymbol{x}} - \hat{\boldsymbol{k}}_{\boldsymbol{x}}(\boldsymbol{t})$ and $\tilde{\boldsymbol{k}}_{\boldsymbol{r}}(\boldsymbol{t}) \triangleq \boldsymbol{k}_{\boldsymbol{r}} - \hat{\boldsymbol{k}}_{\boldsymbol{r}}(\boldsymbol{t})$

A Lyapunov function candidate is constructed to design adaptive controller gains.

$$V = \frac{1}{2}e(t)^T Pe(t) + \frac{1}{2}tr\left(\tilde{k}_x(t)^T \Gamma_x^{-1}\tilde{k}_x(t)\right) + \frac{1}{2}tr(\tilde{k}_r(t)^T \Gamma_r^{-1}\tilde{k}_r(t)) \quad (19)$$

Where $P \in \mathcal{R}^{nxn}$ is a positive definite matrix, which is the solution of the $PA_r + A_r^T P = -Q$, $Q \in \mathcal{R}^{nxn}$ is a positive definite symmetric matrix. Different selection of Q does not affect boundedness or the asymptotic behavior, but affects the transient response. The matrices Γ_x and Γ_r are positive definite matrices that serve as adaptation gains in the adaptive law. They determine the rate at which the adaptive parameters $\hat{k}_x(t)$ and $\hat{k}_r(t)$ are updated. Specifically, Γ_x and Γ_r are typically diagonal matrices, chosen to ensure the stability of the adaptive system and to adjust the responsiveness of the parameter adaptation process [5].

After taking the derivative of the function in Equation (19) and simplifying the resulting equation, we have

$$\begin{aligned}\dot{V} = {} & \frac{1}{2}\dot{e}(t)^T Pe(t) + \frac{1}{2}e(t)^T P\dot{e}(t) \\ & + \frac{1}{2}tr\left(\dot{\tilde{k}}_x(t)^T \Gamma_x^{-1}\tilde{k}_x(t)\right) \\ & + \frac{1}{2}tr\left(\tilde{k}_x(t)^T \Gamma_x^{-1}\dot{\tilde{k}}_x(t)^T\right) \\ & + \frac{1}{2}tr\left(\dot{\tilde{k}}_r(t)^T \Gamma_r^{-1}\tilde{k}_r(t)\right) \\ & + \frac{1}{2}tr\left(\tilde{k}_r(t)^T \Gamma_r^{-1}\dot{\tilde{k}}_r(t)^T\right)\end{aligned} \quad (20)$$

By substituting Equation (18) into Equation (20) and simplifying the resulted equation, we have

$$\begin{aligned}\dot{V} = {} & \frac{1}{2}e(t)^T(PA_r + A_r^T P)e(t) \\ & + e(t)^T P\left(-B\tilde{k}_x(t)x(t) - B\tilde{k}_r(t)r\right) \\ & - tr\left(\tilde{k}_x(t)^T \Gamma_x^{-1}\dot{\hat{k}}_x(t)\right) \\ & - tr(\tilde{k}_r(t)^T \Gamma_r^{-1}\dot{\hat{k}}_r(t))\end{aligned} \quad (21)$$

Where $PA_r + A_r^T P = -Q$

$\dot{\hat{k}}_x(t), \dot{\hat{k}}_r(t)$ are selected as below to cancel out some terms in the Equation (20)

$$\dot{\hat{k}}_x(t) = -\Gamma_x B^T Pe(t)x^T(t) \quad (22)$$

$$\dot{\hat{k}}_r(t) = -\Gamma_r B^T Pe(t)r^T(t) \quad (23)$$

After replacing the Equations (22) and (23) into the Equation (20), we have;

$$\begin{aligned}\dot{V}(t) = {} & -\frac{1}{2}e(t)^T Qe(t) - e(t)^T PB\tilde{k}_x(t)x(t) \\ & - e(t)^T PB\tilde{k}_r(t)r \\ & + tr\left(\tilde{k}_x(t)^T \Gamma_x^{-1}\Gamma B^T Pe(t)x(t)^T\right) \\ & + tr(\tilde{k}_r(t)^T \Gamma_r^{-1}\Gamma B^T Pe(t)r(t)^T)\end{aligned} \quad (24)$$

Taking the transpose of the terms in Equation (24) and using trace properties, we can cancel out the terms except $-\frac{1}{2}e(t)^T Qe(t)$ so the Equation (24) is simplified to,

$$\dot{V}(t) = -\frac{1}{2}e(t)^T Qe(t) \quad (25)$$

Since Q is positive definite symmetric matrix $\dot{V} \leq 0$ and $V \in L_\infty$ so the equilibrium point is Lyapunov stable [5,13].

a) Based on the Equation (19) if $V \in L_\infty$, $\tilde{k}_x(t)$,$\tilde{k}_r(t)$, $e(t) \in L_\infty$

b) Since 'a' holds, based on the Equation (18), $\hat{k}_r(t)$,$\hat{k}_x(t)$, $r(t)$,$x(t) \in L_\infty$

c) Since 'b' holds, Based on the Equation (15), $u(t) \in L_\infty$ thus, Equation (22) and Equation (23) are obtained as adaptive controller gain estimators for the controller in Figure 2 and Robust MRAC adaptive controller gains can be derived as [5,14];

$$\dot{\hat{k}}_x(t) = -\Gamma_x B^T Pe(t)x^T(t) - \Gamma_x \sigma_x ||e||\hat{k}_x(t) \quad (26)$$

$$\dot{\hat{k}}_r(t) = -\Gamma_r B^T Pe(t)r^T(t) - \Gamma_r \sigma_r ||e||\hat{k}_r(t) \quad (27)$$

The addition of e-mode (error-driven adaptation) and σ-mode (model error-driven adaptation) to MRAC enhances its robustness against uncertainties and disturbances. In the e-mode, the control law adjusts parameters based on the output error and the reference model, ensuring effective tracking even in the face of system changes. The σ-mode employs the actual plant output versus the identified model to minimize deviations, making the controller resilient to uncertainties [14-16].

Therefore, the combination of these modes boosts MRAC in the following ways;

The e-mode adjusts parameters via the error signal, while the σ-mode corrects based on model error. Together, they ensure accurate tracking during unexpected system shifts. The σ-mode lessens sensitivity to model inaccuracies, actively adapting to differences between plant behavior and the reference model. Incorporating both modes improves stability and performance under uncertainties, expanding the controller's versatility in real-world scenarios.

4. ARBFNN Controller Design

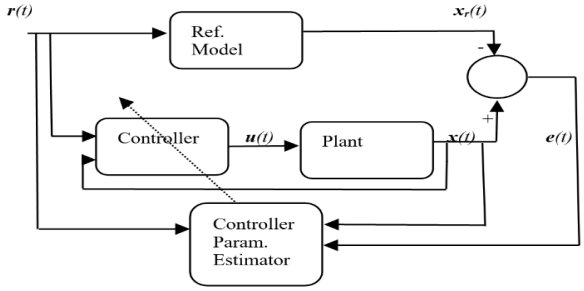

Figure 3. ARBFNN and ASRBFNN controller scheme.

Consider the first-order linear state space equation of the proposed DC motor in Equation (9) and rewrite the equation in general form;

$$\dot{x} = f(x) + gu \quad (28)$$

$$y = x \quad (29)$$

Assuming the desired speed of the motor is r_d,
The error of the output becomes,

$$\boldsymbol{e} = r_d - x \tag{30}$$

The control law is to be designed by isolating $\boldsymbol{u}$ in the Equation (28) and using Equation (33) by disregarding ᛮ

$$\boldsymbol{u} = \frac{-\hat{f}(x) + \dot{r}_d + k_p e}{g} \tag{31}$$

Where $\hat{f}(x)$ is the predicted $f(x)$ by 1-5-1 RBFNN structure, k_p is the proportional gain,

The process of obtaining $\hat{f}(x)$ is as follows [17-19];
Let,

- $j = 1,2,..,5$ and $i = 1$
- $\boldsymbol{x}$ is the input vector,
- c_{ij} is the coordinate value of the center point of the Gaussian function of neuron j for the input i
- $\boldsymbol{b} = b_j$ is the width value of Gaussian function for neuron j,
- The weight values of 1-5-1 RBF structure is $\boldsymbol{W} = w_j$,
- $\boldsymbol{h} = h_j$ represents the radial basis function vector in the hidden layer of 1-5-1 RBF

Note that, c_{ij} and b_j should be chosen based on the scope of input, so that the Gaussian function is mapped effectively [18,19].

$$\hat{f}(\boldsymbol{x}) = \widehat{\boldsymbol{W}}^T h(\boldsymbol{x}) \tag{32}$$

where,

$h_{ij} = \mathrm{e}^{\frac{\|x - c_{ij}\|^2}{b_j^2}}$ and

By replacing Equation (31) into Equation (28), the error dynamics become,

$$\dot{e} = -k_p e + ᛮ \tag{33}$$

Where $ᛮ = \hat{f}(x) - f(x)$
Select k_p based on Equation (33) so that the root of the error dynamic equation is in the left part of the s plane [18,19]. Thus, the error goes to zero as time goes to infinity.
Optimal weight value can be calculated as,

$$\boldsymbol{W}^* = \arg\min_{W \in \Omega}(ᛮ)$$

Then the prediction error regarding the optimal weight becomes

$$\omega = \hat{f}(x|\boldsymbol{W}^*) - f(x)\,, \qquad |\omega| \le \omega_{max}$$

Write the $f(x)$ in terms of ω and replace in Equation (33). Thus, the error dynamics of the system can be rewritten as

$$\dot{e} = -k_p e + (\hat{f}(x) - \hat{f}(x|\boldsymbol{W}^*) + \omega) \tag{34}$$

Replace Equation (32) into Equation (34),

$$\dot{e} = -k_p e + \left(\widehat{\boldsymbol{W}} - \boldsymbol{W}^*\right)^T \boldsymbol{h}(\boldsymbol{x}) + \omega \tag{35}$$

Define Lyapunov function as,

$$V = \frac{1}{2}\boldsymbol{e}^{\boldsymbol{T}} P \boldsymbol{e} + \frac{1}{2\sigma}\left(\widehat{\boldsymbol{W}} - \boldsymbol{W}^*\right)^T \left(\widehat{\boldsymbol{W}} - \boldsymbol{W}^*\right) \tag{36}$$

Where σ is a positive constant and P is a 1x1 positive definite symmetric matrix that can be obtained by solving the following equation

$$-k_p P - P k_P = -Q \tag{37}$$

In which Q is 1x1 positive semidefinite matrix
Taking the derivative of the V ,

$$\dot{V} = \frac{1}{2}\boldsymbol{e}^{\boldsymbol{T}} \boldsymbol{P} \dot{\boldsymbol{e}} + \sigma\left(\widehat{\boldsymbol{W}} - \boldsymbol{W}^*\right)^T \dot{\widehat{\boldsymbol{W}}} \tag{38}$$

Replace Equation (35) and (37) into the Equation (34)

$$\dot{V} = -\frac{1}{2}\,\boldsymbol{e}^{\mathrm{T}}\, Q\, \boldsymbol{e} + (\widehat{\boldsymbol{W}} - \boldsymbol{W}^*)^{\mathrm{T}}\, h(x)\, \boldsymbol{e}^{\mathrm{T}}\, P + \boldsymbol{e}^{\mathrm{T}}\, P\omega + \sigma\, (\widehat{\boldsymbol{W}} - \boldsymbol{W}^*)^{\mathrm{T}} \dot{\widehat{\boldsymbol{W}}} \tag{39}$$

The derivative of the Lyapunov function
We choose adaptive law (see Figure 3) as

$$\dot{\widehat{\boldsymbol{W}}} = -\sigma \boldsymbol{e} P \boldsymbol{h}(\boldsymbol{x}) \tag{40}$$

Substitute $\dot{\widehat{\boldsymbol{W}}}$ into Equation (39) and simplifying the achieved equation, we have,

$$\dot{V} = -\frac{1}{2}\,\boldsymbol{e}^{\mathrm{T}}\, Q\, \boldsymbol{e} + \boldsymbol{e}^{\mathrm{T}} P \omega \tag{41}$$

Since Q is a positive definite matrix, the term $-\frac{1}{2}\,\boldsymbol{e}^{\mathrm{T}}\, Q\, \boldsymbol{e}$ is non-positive. Also, $\boldsymbol{e}^{\mathrm{T}} P\omega$ can be bounded by some small constant value related to the prediction error ω and the positive definite matrix P.
Thus, with the appropriate choice of control parameters, $\dot{V}$ can be made non-positive, ensuring that V does not increase, which is a standard approach in Lyapunov stability theory. This condition $\dot{V} \le 0$ helps in guaranteeing the stability of the system, meaning the system's trajectories will not diverge over time. Therefore, $\dot{V} \le 0$ and $V \in L_\infty$ so the equilibrium point is Lyapunov stable.

5. ASRBFNN Controller Design

Consider the first-order linear state space equation of a general DC motor in Equation (28), and rewrite the equation by adding an input disturbance;

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g\boldsymbol{u} + \boldsymbol{d} \tag{42}$$

$$\boldsymbol{y} = \boldsymbol{x} \tag{43}$$

Where $|d| \le D$ is the input disturbance
Assuming the desired speed of the motor is r_d ,
error of the output becomes,

$$\boldsymbol{e} = x - r_d \tag{44}$$

Sliding mode function for the system selected as,

$$\boldsymbol{s} = z\boldsymbol{e} \tag{45}$$

where z is a positive constant
thus

$$\dot{\boldsymbol{s}} = z(f(\boldsymbol{x}) + g\boldsymbol{u} + \boldsymbol{d} - \dot{r}_d) \quad (46)$$

The control law is to be designed by isolating $\boldsymbol{u}$ in Equation (49) as

$$\boldsymbol{u} = \frac{-\hat{f}(\boldsymbol{x}) + \dot{r}_d - \eta sgn(\boldsymbol{s})}{g} \quad (47)$$

$\eta \geq D.$ and $\hat{f}(\boldsymbol{x})$ is the output neuron of the 1-5-1 RBF can be obtained by using the same process described in sec.4

$$\hat{f}(\boldsymbol{x}) = \boldsymbol{W}^T h(\boldsymbol{x}) \quad (48)$$

let define $f(\boldsymbol{x})$ by using 1-5-1 RBFNN structure as described in sec. 4 regarding to optimal weight with small error $\varepsilon \leq \varepsilon_{max}$ [18] as

$$f(\boldsymbol{x}) = \widehat{\boldsymbol{W}}^{*T} h(\boldsymbol{x}) + \varepsilon \quad (49)$$

Where $h(\boldsymbol{x})$ can be obtained as

$$h_{ij} = \mathrm{e}^{\frac{\|x - c_{ij}\|^2}{2b_j^2}}$$

Substituting Equation (47) into Equation (46) we have

$$\dot{\boldsymbol{s}} = z(\tilde{f}(\boldsymbol{x}) - \eta sgn(\boldsymbol{s}) + \boldsymbol{d}) \quad (50)$$

Where $\tilde{f}(\boldsymbol{x}) = f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}) = (\widehat{\boldsymbol{W}}^{*T} - \boldsymbol{W}^T)h(\boldsymbol{x})$

Define lyapunov function as below by letting $\widetilde{\boldsymbol{W}} = \boldsymbol{W}^* - \widehat{\boldsymbol{W}}$

$$V = \frac{1}{2}\boldsymbol{s}^2 + \frac{1}{2}\tau\, \widetilde{\boldsymbol{W}}^{\boldsymbol{T}}\widetilde{\boldsymbol{W}} \quad (51)$$

where τ is a positive constant

taking derivative of V,we have

$$\dot{V} = \boldsymbol{s}\dot{\boldsymbol{s}} + \tau\widetilde{\boldsymbol{W}}^{\boldsymbol{T}}\dot{\widetilde{\boldsymbol{W}}} = \widetilde{\boldsymbol{W}}^{\boldsymbol{T}}(\boldsymbol{s}h(\boldsymbol{x}) - \tau\dot{\widehat{\boldsymbol{W}}}) + \boldsymbol{s}(\varepsilon + d - \eta sgn(\boldsymbol{s})) \quad (52)$$

We choose adaptive law (see Figure 3) as

$$\dot{\widehat{\boldsymbol{W}}} = \frac{1}{\tau}\boldsymbol{s}\boldsymbol{h}(\boldsymbol{x}) \quad (53)$$

Thus, the first term in Equation (52) is canceled. The first term in the equation is negative, and choosing $\eta \geq \varepsilon_{max} + D + \eta_0, \eta_0 > 0, \dot{V} \leq -\eta_0 sgn(\boldsymbol{s}) \leq 0$ so the equilibrium point is Lyapunov stable.

6. ASRBFNN Controller Design

6.1. System Setup

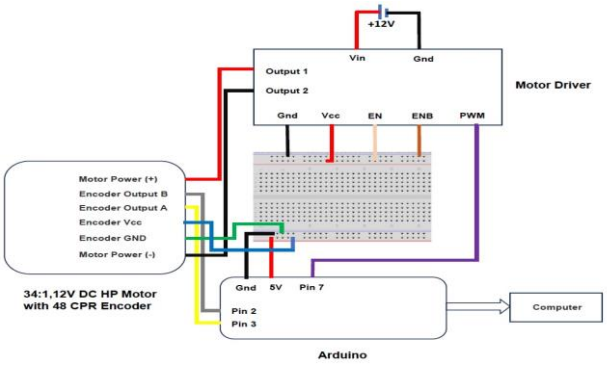

Figure 4. Implementation of DC motor speed control circuit diagram.

Figure 4 illustrates the implementation's connection diagram. For real-time execution of our proposed control methods, we utilize a 12V High Power 34:1 DC motor with a 48 CPR encoder, which is connected to the TB9051FTG motor driver, as depicted in Figure 4. The motor's mathematical model is derived through Matlab [11] and employed in our simulation. Within the diagram, the Arduino assumes the role of controller. Motor speed is determined by reading signals from Encoder outputs via Pin 2 and 3, and it is converted in radians per second (rad/sec). Subsequently, we apply the controller techniques outlined in our paper to generate the control signal. Following this, the calculated control signal is converted to a duty cycle using the equation $255(u\backslash applied\ voltage)$, where the applied voltage is 12V in this instance.
The approximated state-space model for the first-order speed control of the DC motor is depicted in Figure 4 is obtained using the MATLAB System Identification Toolbox. The parameters used are J = 0.009 kg×m^2 , b = 0.01 Nm×s, K = 0.57 V/rad/s, L = 0.006 H, and R = 4.45 Ω. Here K, L, and R values are measured, while the values of J and b are optimized using the MATLAB System Identification Toolbox. The resulting dynamic system of the DC motor is as follows:

$$\dot{\boldsymbol{x}}(t) = -9.2097\boldsymbol{x}(\boldsymbol{t}) + 14.2109\boldsymbol{u}(t)$$
$$\boldsymbol{y}(t) = \boldsymbol{x}(t) \quad (54)$$

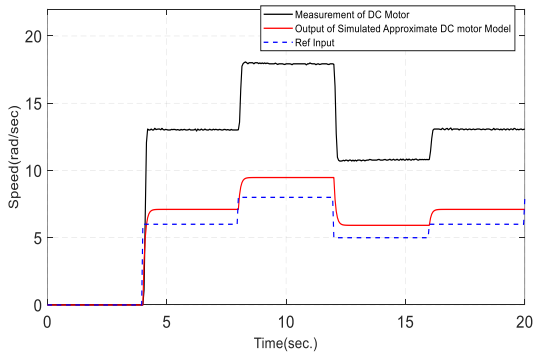

Figure 5. Modeling DC motor (system identification).

Figure 5 shows the real-time speed measurement of the DC motor and response of the estimated state space model given in Equation (54), given stepwise input. The figure indicates that the DC motor model has significant transient and steady-state error.

To enable real-time implementation of speed control for a DC motor using a microcontroller, we adhere to a common practice of selecting the sampling time. The open-loop system's bandwidth is 9.2 rad/s (refer to Figure 6), which translates to a 1.46 Hz sampling frequency. To ensure compatibility with both real-time implementation and simulation, where a model with error is employed, we opt for a sampling frequency that is 10 times faster. Consequently, the maximum selectable sampling time is reduced to 0.0685 seconds (1/14.6). We ultimately choose a sampling time (T) of 0.05 seconds. This choice of sampling time allows for the transformation of the continuous-time controller design into a discrete-time format. Importantly, this selected sampling period is consistently used for simulation purposes to maintain result consistency.

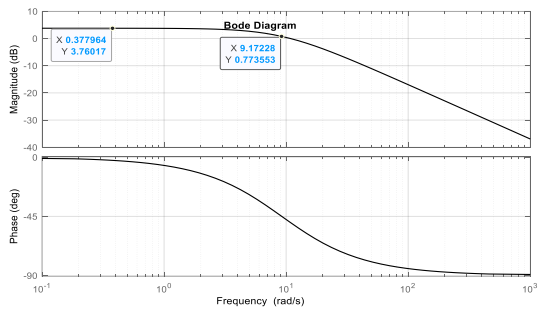

Figure 6. Bode plot of the open-loop system.

6.2. MRAC and Robust MRAC Implementation

Each equation should be written in a separate row and
A conventional full-state feedback controller is designed for the uncertain system in Equation (54) to obtain a reference model for the MRAC design. Designed state and input gains are given below.

$\boldsymbol{k_r} = 0.8444, \boldsymbol{k_x} = 0.1964$

Therefore, the closed-loop system dynamic is obtained as,

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}}(t) = A_r\boldsymbol{x}(t) + B\boldsymbol{k_r}\boldsymbol{r}(t) \quad (55)$$

Where $A_r = (A + B\boldsymbol{k_x})$

$$\dot{\boldsymbol{x}}_{\boldsymbol{r}}(t) = -12\boldsymbol{x}(\boldsymbol{t}) + 11.9997\boldsymbol{r}(t) \quad (56)$$

Where $\boldsymbol{r}(t)$ is the reference input.

Equation (56) is the reference model for the proposed MRAC design. The step response of the reference model is shown in Figure 7. The response has a 0.15 settling time and 0% overshoot with zero steady-state error.

Note that Equation (56) is used for simulation, but it cannot be used for real-time implementation. Instead, a state transition matrix is derived and utilized.

$\varphi(t) = e^{-22t}$ is the state transition matrix for Equation (56),

$x_r[T(k+1)] = \varphi[Tk]x[Tk] + A_r^{-1}(\varphi[Tk] - I_{1\times1})B, k = 0,1,2,\ldots.n, x[0] = 0,$ generates a response of the reference model shown in Figure 7 for a real-time application.

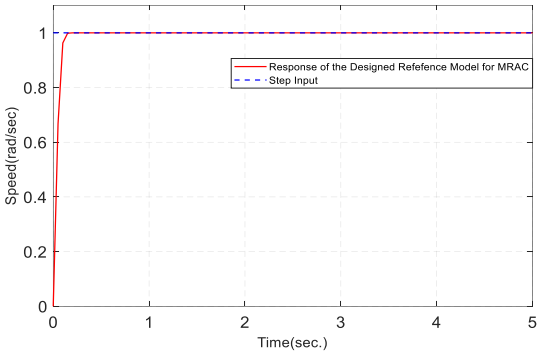

Figure 7. Step response of the reference model.

For the controller gain estimator in Equations (22), (23) $Q = 0.2$ is selected, P is calculated by the solution of $PA_r + A_r^TP = -Q$ given A_r and Q, therefore, $P = 0.0083$ and, Γ is selected as 0.1 .

Thus, the controller gain estimators become,

$$\dot{\hat{\boldsymbol{k}}}_x(t) = -1.42109\boldsymbol{e}(t)\boldsymbol{x}^T(t)$$
$$\dot{\hat{\boldsymbol{k}}}_r(t) = -1.42109\boldsymbol{e}(t)\boldsymbol{r}^T(t) \quad (57)$$

For the Robust controller gain estimator in Equations (26) and (27), $\sigma_x = \sigma_r = 0.1$ are selected and, Γ is selected as 0.4

Thus, the robust controller gain estimators become,

$$\dot{\hat{\boldsymbol{k}}}_x(t) = -5.6844\boldsymbol{e}(t)\boldsymbol{x}^T(t) - 0.1||e||\hat{\boldsymbol{k}}_x(t)$$
$$\dot{\hat{\boldsymbol{k}}}_r(t) = -5.6844\boldsymbol{e}(t)\boldsymbol{r}^T(t) - 0.1||e||\hat{\boldsymbol{k}}_r(t) \quad (58)$$

6.3. ARBFNN and ASRBFNN Implementation

ARBFNN and ASRBFNN controllers are designed by following the design procedures described in sec.4 and sec.5, respectively, for the DC motor approximate system dynamic given in Equation (54)

$f(x) = -9.2097x$ and $g = 14.2109$

1-5-1 RBNN structure is used to predict $f(x)$ for the RBFNN controller, and the selected parameters are as follows,

$k_p = 30, c_{ij} = [6\ 8\ 5\ 6\ 8]$

$b_j = 30, Q = 100\ , P = 1.6667\ , \sigma = 3$

where $j = 1,2,..,5$ and $i = 1$

Based on the selected parameters above, the adaptive law is calculated as

$$\dot{\boldsymbol{W}} = -5.0001e\boldsymbol{h}(\boldsymbol{x}) \quad (59)$$

and the control signal can be obtained by using the Equations (31) and (32).

Considering the system dynamics with input disturbance given Equations (42) and (43) for the ASRBFNN design, the parameters are selected as follows,

$f(x) = -9.2097x\ , g = 14.2109$ and $d(t)$ is shown in Figures 11,12

$k_p = 30, c_{ij} = [6\ 8\ 5\ 6\ 8], z = 3, \tau = 0.022, \eta = 8$

$b_j = 200, \Delta = 0.05$

where $j = 1,2,..,5$ and $i = 1$

to eliminate chattering in the response, sat function is utilized instead of sgn [18].

$$sat(s) = \begin{cases} 1 & s > \Delta \\ ks & |s| \le \Delta \\ -1 & s < -\Delta \end{cases}, \qquad k = \frac{1}{\Delta}$$

Based on the selected parameters above, the adaptive law is calculated as

$$\dot{\hat{\boldsymbol{W}}} = 0.022\boldsymbol{sh}(\boldsymbol{x}) \qquad (60)$$

The control signal can be obtained using Equation (47) by replacing sat function instead of sgn function,

Note that the integral operation should be applied for adaptive laws of the controllers (Equations (57), (58), (59), and (60)) to calculate $u(t)$. Trapezoid rule [20] is employed for taking an integral in a real-time application. Figure 8 shows the prediction of $f(x)$ by RBFNN. The figure indicates that RBFNN closely approximates the system dynamics since g is known.

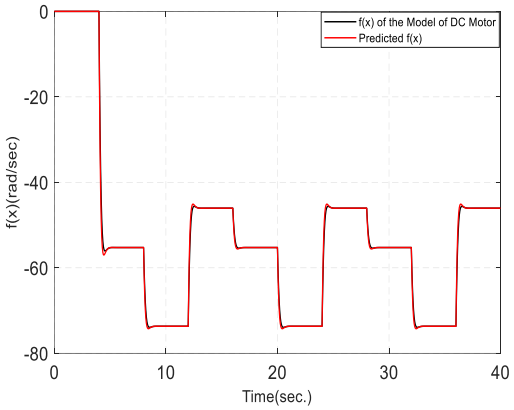

Figure 8. Predicted $\boldsymbol{f}(\boldsymbol{x})$ by RBFNN ($\hat{\boldsymbol{f}}(\boldsymbol{x})$).

Figure 9 presents the responses of the ARBFNN, MRAC, and PI controllers. The PI controller, tuned for the approximated model in Equation (54). It achieves superior simulation performance with zero overshoot and the shortest settling time for the approximated model. Notably, the ARBFNN controller outperforms the MRAC, demonstrating significantly reduced overshoot and a quicker settling time. Although the MRAC response exhibits overshoot and initially reaches the steady state with a delay, it quickly adapts at the start of the second step input. However, its overall settling time remains longer compared to the ARBFNN controller. While all controllers show successful performance for the approximated system, the real-time implementation reveals different system dynamics, as shown in Figure 5. In this context, adaptive controllers are expected to perform better than the conventional PI controller.

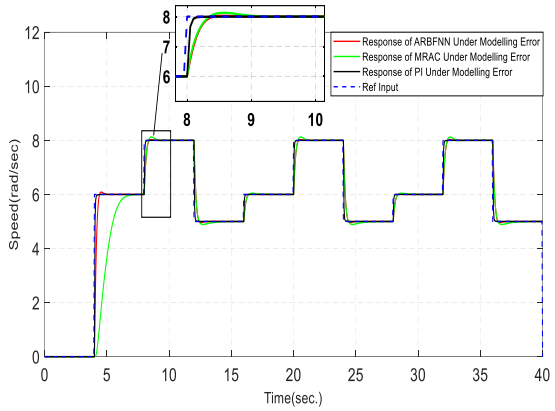

Figure 9. Controller responses under modeling error.

Figure 10 illustrates that the PI controller, designed for an approximated model, proves ineffective during real-time implementation due to modeling errors. In contrast, both the ARBFNN controller and MRAC, which were designed based on the approximated model in Eq. (54), deliver satisfactory outcomes, closely aligning with the simulation results. The figure demonstrates that the ARBFNN exhibits superior performance compared to the MRAC.

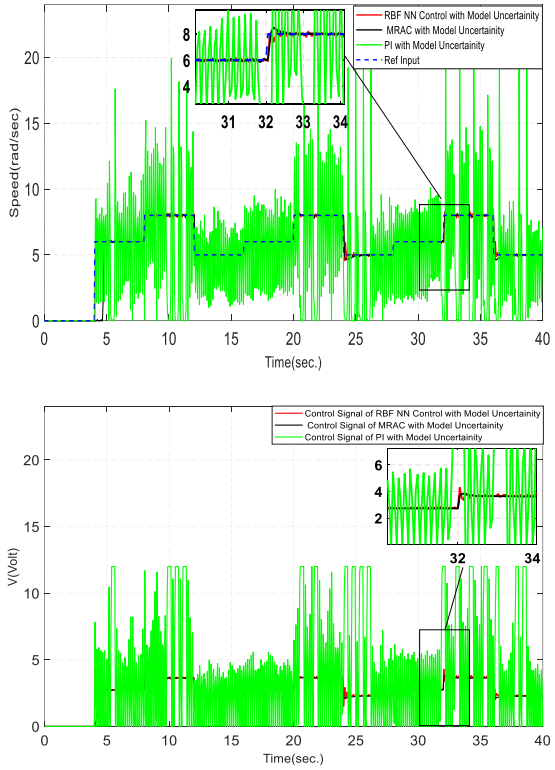

Figure 10. Responses of controllers under modeling error for real-time implementation.

Figure 11 shows the responses of the ASRBFNN and Robust MRAC controllers in the context of input disturbance, despite the continued presence of an inaccurate system model. This scenario is referred to as the "input disturbance case" for simplicity. Notably, the simulation excludes the PI controller to focus on the effects of input disturbance, as the PI controller, designed

for the approximated model, has already been shown to be ineffective for real-time applications. Including the PI controller would not provide meaningful insights into its performance under input disturbance conditions.

As evidenced in the illustration, the ASRBFNN controller excels over the Robust MRAC when facing input disturbances. The figure indicates that the ASRBFNN controller swiftly adapts to control variations and rapidly attenuates disturbances. Conversely, the Robust MRAC demonstrates a sluggish initial adaptation, eventually accommodating control changes and delivering acceptable disturbance rejection.

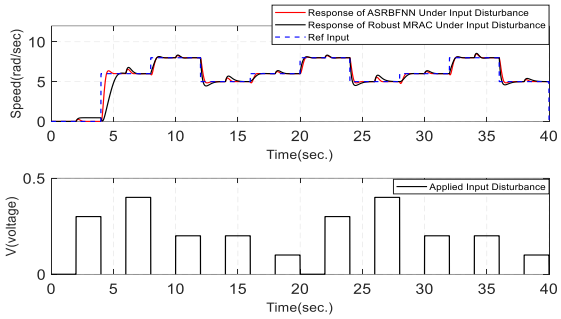

Figure 11. Responses of the controllers under modeling error and input disturbances.

Figure 12 illustrates the real-time application responses of the ASRBFNN and Robust MRAC controllers, both designed using the approximated model, demonstrating their effectiveness in rejecting disturbances in real time. The PI controller, finely tuned for the DC motor through real-time trial and error, performs flawlessly in the absence of input disturbances. This adjustment was necessary because the PI controller designed for the approximated model was ineffective in real-time applications, requiring precise tuning for the exact physical motor to assess its disturbance rejection capability. However, when input disturbances are introduced, the PI controller fails to reject them effectively.

In this context, the figure clearly shows that the ASRBFNN controller outperforms the Robust MRAC, especially when dealing with input disturbances. The ASRBFNN controller quickly adapts to control variations and efficiently suppresses disturbances, exhibiting minimal overshoot. In contrast, the Robust MRAC has a slower initial adjustment, eventually accommodating control fluctuations and providing an acceptable outcome, though with a more pronounced overshoot, for disturbance rejection. The figure underscores the consistency between simulation results and real-time implementation, highlighting the ASRBFNN controller's superior performance in managing disturbances

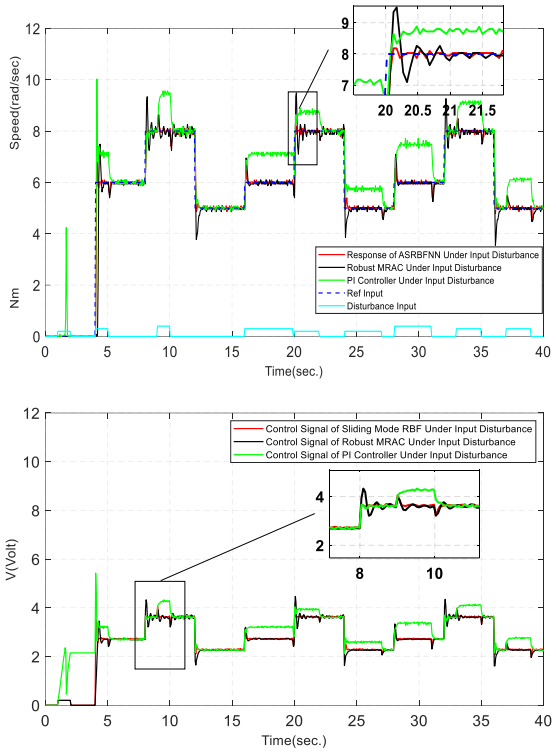

Figure 12. responses of the controllers under modeling error and input disturbances for real-time implementation.

6.4. Adaptive Step-Wise Performance Evaluation

This section presents a thorough, adaptive, step-wise performance analysis for all the proposed controllers. Both MRAC and ARBFNN were tested under model uncertainty conditions, whereas Robust MRAC and ASRBFNN were tested under conditions of model uncertainty with input disturbance. Although all the results have been summarized in unified tables, each pair of controllers is interpreted within its respective category for fair comparison.

All controllers were subjected to a multilevel reference input ($0 \rightarrow 6 \rightarrow 8 \rightarrow 5 \rightarrow 6 \rightarrow 8 \rightarrow 5 \rightarrow 6 \rightarrow 8 \rightarrow 5$), as in Figures 10 and 12, to observe their dynamic behavior for both rising (up-step) and falling (down-step) transitions. Each transition was automatically detected, and the following time-domain indicators were computed for every step window:

- Overshoot (OS %): Maximum deviation above the reference normalized by step amplitude.
- Settling Times (STa, STf): Time to enter and stay within a ± 5 % band of the reference; STa = adaptive, STf = filtered.
- Mean Absolute Error (MAE): Average absolute deviation between output and reference.
- Ripple: Steady-state oscillation amplitude (peak-to-peak).

These metrics were separately calculated for up- and down-steps to reveal controller asymmetries during acceleration and deceleration.

Table 1. Step-Wise Performance Metrics for Rising Reference Transitions (up-step)

	Mean OS	Median STa	Median STf	Mean MAE	Mean Ripple
ARBF NN	19.69	0.305	0.28	0.14	0.282
ASRBF NN	30.44	—	0.175	0.127	0.168
MRAC	18.19	0.385	0.485	0.33	0.160
Robust MRAC	72.39	—	0.615	0.253	0.183

During rising transitions:
Model-uncertainty group: *ARBFNN vs MRAC, ARBFNN* achieved a smaller MAE and faster settling, demonstrating smoother adaptation. MRAC showed comparable overshoot but higher steady-state error.
Disturbance group: *ASRBFNN vs Robust MRAC, ASRBFNN* yielded the lowest error and ripple, while Robust MRAC exhibited excessive overshoot due to aggressive adaptation.

Table 2. Step-Wise Performance Metrics for Falling Reference Transitions (down-step)

	Mean OS	Median STa	Median STf	Mean MAE	Mean Ripple
ARBF NN	10.67	0.36	0.36	0.139	0.23
ASRBF NN	10.56	3.13	0.21	0.127	0.153
MRAC	12.33	0.72	1.13	0.168	0.08
Robust MRAC	42.33	3.28	3.38	0.171	0.157

During falling transitions:
Model-uncertainty group:*ARBFNN* again outperformed *MRAC*, showing lower overshoot and faster recovery with minimal ripple.
Disturbance group: *ASRBFNN* preserved the smallest MAE and ripple, maintaining high stability during rapid decelerations, whereas *Robust MRAC* had longer settling due to over-adaptation.
All in all, ARBFNN and MRAC for model uncertainty within each category, and ASRBFNN and Robust MRAC in the case of input disturbance. Over both rising and falling transitions, ASRBFNN yielded the most consistent and robust tracking with the least oscillation. The experiment results confirm that it has better damping, precision, and disturbance-rejection capability under dynamically varying conditions.

7. Conclusion

In conclusion, this study evaluated adaptive control strategies for a DC motor facing modeling inaccuracies and input disturbances. The research encompassed various approaches, including MRAC and ARBFNN Controllers, as well as Robust MRAC and ASRBFNN Controllers to address combined challenges.
The results from extensive simulations and practical implementations highlighted the limitations of traditional controller designs in handling uncertain system models. Both ARBFNN and MRAC methods demonstrated impressive adaptability to dynamic changes and modeling uncertainties. ASRBFNN and Robust MRAC Controllers also showcased their effectiveness in mitigating disturbances.
The findings indicated that traditional PI controllers were inadequate, whereas adaptive controllers performed better. This underscores the importance of adaptive control techniques in managing the complexities of DC motor control.
Going beyond conventional transient analysis, an adaptive, stepwise performance evaluation was conducted to assess each controller's behavior under multiple rising and falling reference transitions. In model-uncertainty conditions, the results showed that the ARBFNN outperforms MRAC, demonstrating better dynamic adaptability and disturbance-rejection capability, with a smaller mean absolute error and faster settling time. However, ASRBFNN achieved the best balance between robustness, damping, and tracking accuracy in model-uncertainty + disturbance. Aggressive adaptation caused a larger overshoot in robust MRAC, but the system stayed stable within reasonable bounds. Overall, ASRBFNN was the most reliable for both up- and down-step transitions.
From the viewpoint of computational burden, MRAC and Robust MRAC controllers show the least complexity since their adaptation laws consist of simple matrix multiplications and parameter updates in each iteration. The ARBFNN and ASRBFNN controllers, however, include extra calculations with respect to Gaussian activation functions and online weight adaptation, which slightly increases the execution time. The ASRBFNN controller has the highest computation cost among all proposed methods because of the involvement of both the sliding mode term and the adaptive weight update in it. However, the total execution time for all controllers remains inside the sampling period (0.05s) for an Arduino Uno implementation and thus shows that the proposed schemes are fit for real-time applications without violating hardware computational burdens.
Future work could focus on applying these adaptive control strategies to robotic systems, particularly industrial robotic arms, where DC motors are commonly used. Evaluating the performance of adaptive controllers in such systems is crucial. In the future, the adaptive controllers developed in this work will be combined with artificial intelligence techniques such as machine learning based optimization and reinforcement learning frameworks, for autonomous tuning of adaptation gains that reduce computational overhead and further enhance robustness in real-time control environments.

8. References

[1] Khan, H., Khatoon, S., and Gaur, P., “Comparison of Various Controller Design for the Speed Control of DC Motors Used in Two-Wheeled Mobile Robots”,

International Journal of Information Technology, 13(2), 713-720, 2021.

[2] Liang, X. et al., "Output Feedback Asymptotic Tracking Control for Uncertain DC Motors", *International Journal of Control, Automation and Systems*, 21(8), 2748-2759, 2023.

[3] Mahajan, N. P., and Deshpande, S. B., "Study of Nonlinear Behavior of DC Motor Using Modeling and Simulation", *International Journal of Scientific and Research Publications*, 3(3), 1-6, 2013.

[4] Shekhar, A., and Sharma, A., "Review of Model Reference Adaptive Control", *International Conference on Information, Communication, Engineering and Technology (ICICET)*, 2018, 1-5.

[5] Ioannou, P., and Fidan, B., *Adaptive Control Tutorial*, Society for Industrial and Applied Mathematics, Philadelphia, 2006.

[6] Benosman, M., "Model-Based vs Data-Driven Adaptive Control: An Overview", *International Journal of Adaptive Control and Signal Processing*, 32(5), 753-776, 2018.

[7] Chao, K.H., Hsieh, C.T., and Chen, X.J., "A Robust Controller Based on Extension Sliding Mode Theory for Brushless DC Motor Drives", *Electronics*, *13*(20), 4028, 2024.

[8] Alejandro-Sanjines, U. et al., "Adaptive PI Controller Based on a Reinforcement Learning Algorithm for Speed Control of a DC Motor", *Biomimetics*, 8(5), 434, 2023.

[9] Na, J., Ren, X., and Zheng, D., "Adaptive Control for Nonlinear Pure-Feedback Systems With High-Order Sliding Mode Observer", *IEEE Transactions on Neural Networks and Learning Systems*, 24(3), 370-382, 2013.

[10] Liu, L., Liu, Y., and Chen, C. L. P., "Adaptive Neural Network Control for a DC Motor System with Dead-Zone", *Nonlinear Dynamics*, 72, 141-147, 2012.

[11] MathWorks, *MATLAB and Statistics Toolbox Release 2012b*, MathWorks Inc., Natick, Massachusetts, 2012.

[12] Wu, W., "DC Motor Parameter Identification Using Speed Step Responses", *Modelling and Simulation in Engineering*, 2012, 1937-1941.

[13] Narendra, K. S., and Annaswamy, A. M., *Stable Adaptive Systems*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[14] Nguyen, N. T., *Model-Reference Adaptive Control*, Springer, 2018.

[15] Volyanskyy, K. Y., Haddad, W. M., and Calise, A. J., "A New Neuroadaptive Controller Architecture for Nonlinear Uncertain Dynamical Systems: Beyond σ- and e-Modifications", *47th IEEE Conference on Decision and Control*, 2008, 80-85.

[16] Rao, M. P. R. V., and Leckie, T. J., "Robust Adaptive Control: Improved e-Modification", *IFAC Proceedings*, 31(22), 127-132, 1998.

[17] Jiang, Y. et al., "A Brief Review of Neural Networks Based Learning and Control and Their Applications for Robots", *Complexity*, 2017(1), 1-14, 2017

[18] Liu, J., Intelligent Controller Design and MATLAB Simulation, Springer, 2018.

[19] Liu, J., Radial Basis Function Neural Network Control for Mechanical Systems, Springer, 2013.

[20] Chapra, S. and Canale, R., *Numerical Methods for Engineers*, McGraw-Hill Education, New York, 2014.