Neural Network-Based DF-MRAC of Nonlinear Aerospace Systems: A Comparative Study of RBF, WNN, and ELM Models

Mayron Pantoja Cardoso
Dept. of Electrical Engineering and Industrial Informatics
Federal University of Technology – Paraná (UTFPR)
Curitiba, Brazil
mayroncardoso@alunos.utfpr.edu.br

Aline Silva Lima
Dept. of Electrical Engineering and Industrial Informatics
Federal University of Technology – Paraná (UTFPR)
Curitiba, Brazil
alinesilvalima@alunos.utfpr.edu.br

Lúcia Valeria Ramos de Arruda
Dept. of Electrical Engineering and Industrial Informatics
Federal University of Technology – Paraná (UTFPR)
Curitiba, Brazil
lvrarruda@utfpr.edu.br

***Abstract*—This paper investigates the application of Derivative-Free Model Reference Adaptive Control (DF-MRAC) to the attitude control dynamics of finless rockets utilizing Thrust Vector Control (TVC). The proposed approach integrates an adaptive model that applies different neural network architectures for system uncertainty approximation to ensure precise trajectory tracking. Specifically, this work evaluates and compares the performance of Radial Basis Function (RBF) networks, Wavelet Neural Networks (WNN), and Extreme Learning Machines (ELM) in estimating uncertainties within the DF-MRAC framework. Simulation results demonstrate the effectiveness of the DF-MRAC approach in managing rapid dynamic changes and disturbances while maintaining stability and alignment with reference trajectories, and provide a comparative analysis of the RBF, WNN, and ELM networks in this context. The results highlight the potential of DF-MRAC for control in aerospace systems under uncertain conditions and offer insights into the suitability of different neural network architectures for uncertainty estimation.**

***Index Terms*—Derivative-Free Adaptive Control (DF-MRAC), Thrust Vector Control (TVC), Rocket Attitude Control, Neural Networks, Radial Basis Functions (RBF), Wavelet Neural Networks (WNN), Extreme Learning Machines (ELM).**

I. Introduction

The control of rockets is a critical component of aerospace engineering, directly influencing the stability, trajectory, and mission success of these vehicles [1]. For systems that lack passive stability, such as finless rockets, advanced control strategies are essential to maintain their attitude. Thrust Vector Control (TVC) involves gimbaling the thrust produced by the engine, enabling precise adjustments to the vehicle's trajectory and stabilization [2], [3].

Wind and other disturbances introduce uncertainties and dynamic variations into the control system, pointing out the need for adaptive control strategies. Model Reference Adaptive Control (MRAC) has proven to be effective in addressing such challenges by dynamically adjusting control parameters to ensure the system follows a desired reference model [4]. However, conventional MRAC methods often rely on derivatives of the adaptive law, which can limit their performance in systems with rapid dynamic changes and noise [5].

To address these limitations, Derivative-Free Model Reference Adaptive Control (DF-MRAC) provides a promising alternative. By eliminating dependence on derivatives of adaptive law, this approach enhances adaptability to fast model variations. The application of DF-MRAC to TVC systems offers the potential to effectively manage uncertainties and disturbances, ensuring performance under a range of operating conditions [6].

On the other hand, Artificial Neural Networks (ANNs) have emerged as efficient tools for modeling complex dynamic systems and enhancing adaptive controllers. This is especially true for rocket attitude control, as shown at [7], which has strong nonlinearities, parametric uncertainties, and external disturbances. ANNs can approximate uncertain dynamics and provide adaptive compensation without requiring explicit derivatives, as in classical control approaches.

This study applies DF-MRAC to the attitude control problem of finless launch vehicles utilizing TVC, in which a neural network model estimates and manages uncertainties and disturbances. The investigation develops a comparative analysis of RBF, WNN, and ELM neural network architectures as uncertainty estimators within the DF-MRAC framework. This analysis evaluates their efficacy for precise trajectory tracking under uncertain conditions, quantified by Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Coefficient of Determination (R^2), revealing that neural network approximates are useful, however their performance is sensitive to hyperparameter configurations, underscoring the

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

importance of careful parameter calibration.

The paper is structured as follows: It starts with an introduction to Thrust Vector Control and the Launch Vehicle Model. The Adaptive Control section formulates the Model Reference Adaptive Control and Derivative-Free MRAC architectures, including Neural Network-Based Modeling for DF-MRAC. A Simulation Study is then presented, with results detailing Adaptive Control and Uncertainty Approximation, State Tracking, and Control Inputs. The paper concludes by summarizing the findings and discussing potential future directions.

II. Thrust Vector Control

The attitude and direction control of space vehicles is achieved through mechanisms such as thrust vector control (TVC), engine thrust adjustment (throttling), or a combination of both. These methods guarantee the stabilization and direction of the vehicle during flight. In finless rockets, TVC is crucial because they lack passive stability to sustain their trajectory. The gimbaled engine system allows the rocket to perform small rotations, adjusting the thrust angle and thus the trajectory [2]. The engine nozzles are connected to the vehicle by spherical pivots, allowing gimbaling along two axes. This movement generates forces perpendicular to the velocity vector, adjusts the trajectory, and stabilizes. Each engine has actuators that control the inclination in pitch(δ_y) and yaw (δ_z) axes, adjusting the nozzle orientation according to the vehicle's reference angles. The gimbal angle typically varies between $\pm 5^\circ$ and $\pm 10^\circ$ [3]. In addition to gimbaling, the vehicle's speed can be adjusted by controlling the total thrust (T_0). The combination of gimbaling, trimming, and thrust control allows precise adjustments to the trajectory and orientation of the space vehicle. These methods are adapted to different flight conditions and specific missions, such as launches, in-orbit maneuvers, and landings. In this work, we propose the use of TVC embedded into a DF-MRAC architecture for attitude control of a launch vehicle, a type of finless rocket. The free-body diagram in Fig. 1, modeling the launch vehicle, refers to δ_y solely as δ.

III. Launch Vehicle Model

For analysis and design of a pitch-axis flight control system of a launch vehicle, the free-body diagram created with an inertial reference frame (X, Y, Z) with its origin at the vehicle's center of gravity is assumed with its X-axis along the vertical axis and its Z-axis along the horizontal direction. Body-fixed (x, y, z) axes with origin at the center of mass (x_{cg}) are shown in Fig. 1. The dynamics for small inclination angles simplify the analysis and make it possible to consider the system as Linear Time-Invariant [1].

From the free-body diagram, we have:

- x_{cp} is center-of-pressure and m is the vehicle mass;
- θ is the small pitch attitude from a vertical inertial reference axis;
- P is the gravitational force, aligned with the center of the earth;

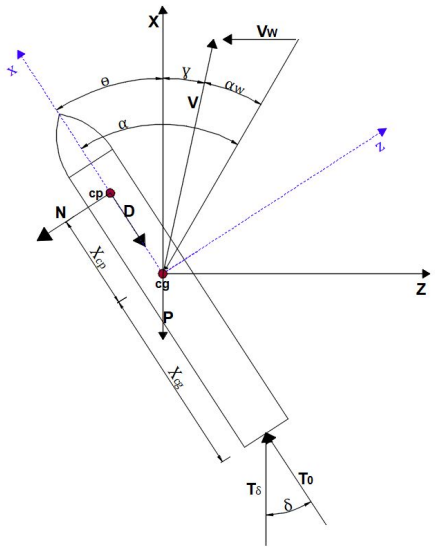

Fig. 1. Free-body diagram of pitch-axis model of the launcher

- V is the vehicle velocity;
- V_w is the wind disturbance velocity
- T_0 is the sustainer thrust for δ = 0;
- T_δ is the gimbaled control thrust;
- δ is the gimbal deflection angle;
- D is the aerodynamic axial force (drag);
- N is the aerodynamic normal force (lift) acting on the center of pressure;
- $\dot{Z}$ is the inertial drift velocity;
- α_w = V_w/V is the wind-induced angle of attack.

Consider an uncertain dynamic system described by:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + B[u(t) + \Delta(x(t))], \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{1}$$

where:

- $x(t) \in \mathbb{R}^n$: State vector of the system.
- $u(t) \in \mathbb{R}^m$: Control input.
- $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$: Known matrices.
- $\Delta(x(t))$: System uncertainty, representing disturbances or unknown dynamics.

According to the dynamic model of attitude control proposed by [1], the launch vehicle in Fig. 1 can be modeled as the state equations (1), resulting in the matrices of A and B equation (2), state $x(t) = [\theta\ \dot{\theta}\ \dot{Z}]^T$, control $u(t) = [\delta\ \alpha_w]^T$, matrices $C = I_{3x3}$ and $D = 0$:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ M_\alpha & 0 & \frac{M_\alpha}{V} \\ \frac{(D-F-N_\alpha)}{m} & 0 & \frac{N_\alpha}{mV} \end{bmatrix}; B = \begin{bmatrix} 0 & 0 \\ M_\delta & M_\alpha \\ \frac{T_\delta}{m} & \frac{N_\alpha}{m} \end{bmatrix} \tag{2}$$

For the uncertain dynamic system model given by the equation (1), the property of controllability is assumed, considering that the pair (A, B) is controllable. This property ensures that the input $u(t)$ has sufficient access to the state space to

stabilize all unstable modes of the plant, as defined by the controllability matrix C_o:

$$C_o = \begin{bmatrix} B & AB & A^2B & \cdots & A^{n-1}B \end{bmatrix}. \tag{3}$$

The pair (A, B) is controllable if $\text{rank}(C_o) = n$, where n is the state-space dimension [8]. This condition is essential to ensure that the control system can be configured to track the reference model as developed in this work, even in scenarios with uncertainties.

IV. Adaptive Control

A. Model Reference Adaptive Control

The goal of MRAC is to design a control law $u(t)$ that makes the system follow a predefined reference model:

$$\dot{x}_m(t) = A_m x_m(t) + B_m r(t), \tag{4}$$

where $x_m(t) \in \mathbb{R}^n$ is the state of the reference model, $r(t) \in \mathbb{R}^r$ is a continuous and bounded reference input, and $A_m \in \mathbb{R}^{n\times n}$, $B_m \in \mathbb{R}^{n\times r}$ are designed matrices, with A_m being Hurwitz (all real parts of the eigenvalues of A_m are negative).

The matrices A_m and B_m are defined in the context of a control strategy with state feedback and feedforward as follows:

$$A_m = A - BK_1, \quad B_m = BK_2, \tag{5}$$

where $K_1 \in \mathbb{R}^{m\times n}$ is the state feedback gain obtained through Linear Quadratic Regulator (LQR) design. The optimal solution to minimize the quadratic performance index is found by solving the algebraic Riccati equation:

$$SA + A^T S - SBR^{-1}B^T S + Q = 0, \tag{6}$$

where S is the unique, stabilizing, positive definite solution, guaranteed by the controllability of the pair (A, B). The gain K_1 is calculated as $K_1 = R^{-1}B^T S$. The matrix $K_2 \in \mathbb{R}^{m\times r}$ is the feedforward gain that adjusts the system so that the output $y(t)$ follows the reference $r(t)$ in steady-state.

In the steady-state, where $\dot{x} = 0$, the output $y(\infty)$ must track the reference r, requiring $y(\infty) = Cx(\infty) = r$. Solving for $x(\infty)$, the gain K_2 is determined as:

$$K_2 = -\left(C(A - BK_1)^{-1}B\right)^{-1}. \tag{7}$$

The uncertainty $\Delta(x(t))$ can be linearly parameterized as:

$$\Delta(x(t)) = W^T \beta(x(t)) + \epsilon(x(t)), \tag{8}$$

where $W \in \mathbb{R}^{s\times m}$ is a constant but unknown weight matrix, $\beta : \mathbb{R}^n \to \mathbb{R}^s$ is a known vector of basis functions in the form $\beta(x) = \begin{bmatrix}\beta_1(x) & \beta_2(x) & \dots & \beta_s(x)\end{bmatrix}^T \in \mathbb{R}^s$, and $\epsilon(x(t)) \in \mathbb{R}^{n\times m}$ is a residual approximation error, assumed to be unknown but bounded by $\|\epsilon(x(t))\| \le \epsilon^*$.

The total control law is composed of two terms:

$$u(t) = u_n(t) - u_{ad}(t), \tag{9}$$

where $u_n(t) = -K_1 x(t) + K_2 r(t)$ is the nominal control term designed to handle the known dynamics, and $u_{ad}(t) = \hat{W}^T(t)\beta(x(t))$ is the adaptive feedback component that compensates for the uncertainty $\Delta(x(t))$.

To approximate the system uncertainty $\Delta(x(t))$, the adaptive control term $u_{ad}(t)$ computes the weights $\hat{W}(t) \in \mathbb{R}^{s\times m}$, using the following update rule:

$$\dot{\hat{W}}(t) = \gamma\beta(x(t))e^T(t)PB, \tag{10}$$

where $\gamma > 0$ is a fixed gain that adjusts the adaptation rate, $e(t) = x(t) - x_m(t)$ is the tracking error between the system state and the reference model, and $P \in \mathbb{R}^{n\times n}$ is a positive definite matrix solution of the Lyapunov equation:

$$0 = A_m^T P + PA_m + Q_L, \tag{11}$$

with $Q_L = Q_L^T > 0$.

This formulation, based on the approach developed at [4], ensures the boundedness and stability of the adaptive system as demonstrated in [5].

The system dynamics can be rewritten as:

$$\dot{x}(t) = A_m x(t) + B_m r(t) + B\tilde{W}^T\beta(x(t)) + B\epsilon(x(t)). \tag{12}$$

where $\tilde{W}(t) = W - \hat{W}(t)$ represents the error between the real and estimated weights.

The works in [4] and [5] use weight update laws, typically formulated as ordinary differential equations that govern the evolution of the estimated weights. A fundamental premise shared by these works is the assumption of the existence of a constant, albeit unknown, set of ideal weights that parameterize the system's uncertainty.

Our work uses neural network models to replace uncertainty estimation with a first-order differential equation. The weights in equations 8 and 10 correspond to the weights learned in the neural network models. Moreover, the beta functions in these equations correspond to the neuron's activation function from a three-layer feed-forward neural network model, as will be discussed in the next section V.

Using a neural network model to estimate uncertainty allows us to formulate a Derivative-Free Model Reference Adaptive Control strategy, as explained in the following.

B. Derivative-Free MRAC Architecture

Derivative-Free Model Reference Adaptive Control (DF-MRAC) distinguishes itself from standard MRAC by two primary characteristics: it relaxes the assumption of constant unknown weights in favor of time-varying weights, and its weight update law is not expressed as an ordinary differential equation, which motivates the "Derivative-Free" nomenclature. This characteristic allows for enhanced adaptation in systems with fast and variable dynamics. The model is applicable to systems subject to abrupt changes, such as structural failures or reconfigurations, as evidenced in practical applications in aircraft control [6], [9].

The proposed adaptive control architecture, shown in Fig. 2, illustrates the interactions between the reference system, the uncertain system, and the DF-MRAC adaptation mechanism.

The continuous weight adaptation law expressed in equation (10) describes the continuous rate of variation of the estimated weights, ensuring the adaptive system's stability under established conditions.

As presented in [6], the weight dynamics $\hat{W}(t)$ in the nonlinear uncertain dynamic system are defined by the following derivative-free update law:

$$\hat{W}(t) = \Omega_1 \hat{W}(t-\tau) + \hat{\Omega}_2(t), \tag{13}$$

where $\tau > 0$, $\Omega_1 \in \mathbb{R}^{s\times s}$, and $\hat{\Omega}_2 : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{s\times m}$. The conditions to ensure that the tracking error $e(t)$ and the estimation error $\hat{W}(t)$ are uniformly ultimately bounded (UUB) are given by:

$$\begin{aligned} 0 \leq & \Omega_1^T \Omega_1 < \kappa_1 I, \quad 0 < \kappa_1 < 1, \\ \hat{\Omega}_2(t) = & \kappa_2 \beta(x(t)) e^T(t) P B, \quad \kappa_2 > 0, \end{aligned} \tag{14}$$

with $P \in \mathbb{R}^{n\times n}$ satisfying (11) for any $Q > 0$. Under these conditions, it is guaranteed that $e(t)$ and $\hat{W}(t)$ remain uniformly bounded, ensuring the stability of the adaptive system. Considering τ_s as the step size, $\Omega_1 = I$, $\kappa_2 = \gamma\tau_s$, and $\tau = \tau_s$, the weight update law presented in Equation (13) can be expressed as:

$$\hat{W}(t) = \gamma\tau_s \beta(x(t)) e^T(t) P B + \hat{W}(t-\tau_s). \tag{15}$$

However, the choice of $\Omega_1 = I$ is generally not permitted due to additional stability requirements. Nevertheless, [6] demonstrates that this choice is valid, consequently, the final formula in equation (15) updates the adaptive weights as a combination of the incremental term, $\gamma\tau_s\beta(x(t))e^T(t)PB$, and the previous weight value, $\hat{W}(t-\tau_s)$, preserving the continuity of the adaptation process.

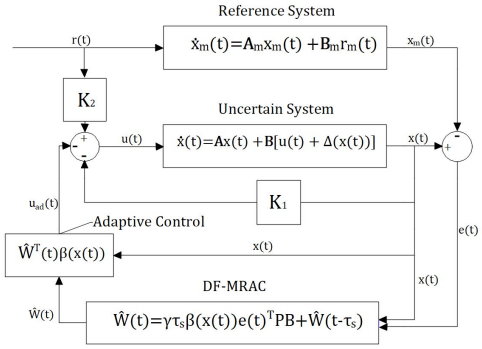

Fig. 2. Block diagram of the DF-MRAC adaptive control system

V. Neural Network-Based Modeling for DF-MRAC

In this work, we integrate a Single-Hidden-Layer Feedforward Network (SHLFN) architecture into the Derivative-Free Model Reference Adaptive Control (DF-MRAC) scheme to estimate uncertainty in the attitude control of a thrust-vectoring rocket. Fig. 3 illustrates this architecture, showing the mapping from the input state $x(t)$ through the hidden layer activation functions $\beta(x)$ to the output $u_{ad}(t)$. The weights $\hat{W}(t)$ are applied to the outputs of the activation functions to generate the control signal, resulting in a weight-free derivative update law.

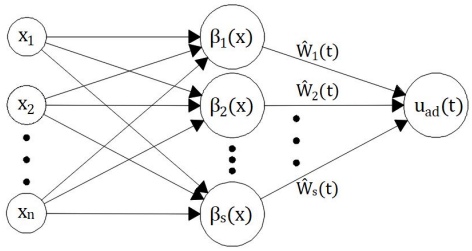

Fig. 3. Neural network-based uncertainty approximator architecture in DF-MRAC

A. Selected Neural Network Architectures

To evaluate the DF-MRAC enhanced by neural networks, a set of distinct activation functions $\beta(x)$ are selected for this study, resulting in three neural network models: Radial Basis Function Networks (RBF), Wavelet Neural Networks (WNN), and Extreme Learning Machines (ELM). These models were chosen for their complementary characteristics and potential applicability to adaptive control of nonlinear systems.

1) Radial Basis Function (RBF) Networks: RBF networks are known for their universal approximation capability and fast convergence. Each neuron in the hidden layer has a radial activation function centered around a point c_i. Two activation variants are considered in this work:

$$\text{Gaussian:} \quad \beta(x) = \exp(-d\|x - c_i\|^2) \tag{16}$$

$$\text{InvMultiquadric:} \quad \beta(x) = (1 + d\|x - c_i\|^2)^{-1/2} \tag{17}$$

where c_i denotes the center of the neuron, uniformly distributed along the input domain with symmetric spacing around zero and d is given by $d = (\Delta c)^{-2}$, with Δc representing the distance between adjacent centers.

RBF networks, especially those based on Gaussian and Inverse Multiquadric activations, have been effectively applied in adaptive control of uncertain nonlinear systems [10]. Additionally, RBF-based adaptive controllers have been successfully applied to fault-tolerant spacecraft attitude control [7] and trajectory tracking in robotic systems [11].

2) Wavelet Neural Networks (WNN): WNNs integrate the time-frequency localization properties of wavelets with the structure of neural networks, enhancing their ability to approximate highly dynamic and nonlinear systems. Two wavelet activations are applied:

$$\text{Mexican Hat:} \quad \beta(x) = (1 - x^2)\exp\left(-\frac{x^2}{2}\right) \tag{18}$$

$$\text{Morlet:} \quad \beta(x) = \cos(5x)\exp\left(-\frac{x^2}{2}\right) \tag{19}$$

Wavelet networks have demonstrated effectiveness in control tasks requiring dynamic adaptation. On [12] Morlet wavelets for quadrotor attitude control were used, achieving robust tracking. Furthermore, [13] combined WNN approximators with sliding-mode control for precise trajectory tracking of robot manipulators.

3) Extreme Learning Machines (ELM): ELM is a rapid training method for single-hidden-layer feedforward networks using ridge functions as activation, where input weights and biases are randomly assigned and fixed, while only the output weights are learned analytically. The following ridge activation functions are considered in this work:

$$\text{Logsig:}\quad \beta(x) = \frac{1}{1+e^{-x}} \tag{20}$$

$$\text{Tansig:}\quad \beta(x) = \frac{2}{1+e^{-2x}} - 1 \tag{21}$$

$$\text{Exponential:}\quad \beta(x) = \exp(-x^2) \tag{22}$$

ELMs have been successfully used for fast online learning in adaptive control, notably in quadrotor UAV control [14] and event-triggered adaptive schemes for nonlinear systems [15].

VI. Simulation Study

The simulation study uses MATLAB to implement the launch vehicle model in Section III. Implementation details also include configuring the model and initializing it with an LQR controller for the DF-MRAC. The derivative-free adaptive structure was implemented to monitor and adjust the parameters of the rocket system. Specifically, optimized control parameters were used to ensure stability during ascent. Table I brings the operational parameters for the launch vehicle model used in this study.

TABLE I
REFERENCE PARAMETERS AT T = 03 MIN

Parameters	Values	Unit
M_α	133,6152	s^{-2}
M_δ	5,7233	s^{-2}
x_{cg}	8,68	m
x_{cp}	8,18	m
m	13248	kg
I_y	320407,7	kgm^2
N_α	5233,66	kN/rad
T_0	224,18	kN
T_δ	224,18	kN
V	1944,44	m/s
D	740,56	kN

Considering Table (I) and equation (2), the matrices A and B are computed for the launch vehicle model. The system's controllability is verified by the controllability matrix C_o in equation (3). Given:

$$C_o = \begin{bmatrix} 0 & 0 & 5.72 & 133.61 & 0.15 & -27.14 \\ 5.72 & 133.61 & 0.15 & -27.14 & 764.68 & 1.7859e04 \\ 2.27 & -395.05 & -0.46 & 80.26 & -2.0508e03 & -4.7897e04 \end{bmatrix}$$

its rank is $\text{rank}(C_o) = 3$, equal to the number of states in the model, confirming controllability.

A. DF-MRAC Implementation

The nominal control law u_n was implemented using the state feedback gain K_1 and the feedforward gain K_2, computed as:

$$K_1 = \begin{bmatrix} 5.4843 & 3.2373 & 1.0960 \\ 2.2377 & 1.1015 & -2.9658 \end{bmatrix}, \quad K_2 = \begin{bmatrix} 3.5719 & 1.0960 \\ 1.3197 & -2.9663 \end{bmatrix}.$$

The weighting matrices Q and R were chosen to balance state tracking performance and control effort. The Lyapunov matrix P, calculated to satisfy the Lyapunov equation associated with the system's stability, is:

$$P = \begin{bmatrix} 1.0277 & 0.0194 & 0.0065 \\ 0.0194 & 0.0222 & 0.0074 \\ 0.0065 & 0.0074 & 0.0029 \end{bmatrix}.$$

The simulation was carried out over a time interval of $t = 20\,\text{s}$ with a time step $\tau_s = 0.001$. The reference signal $r(t)$ is a step function, changing values at intervals of 2 seconds. Initial state conditions were set to zero to represent an unperturbed system.

B. Experimental Setup for Different Neural Network Models

The experimental setup for the neural network models is detailed in the flowchart in Fig. 4, which illustrates the performance evaluation process.

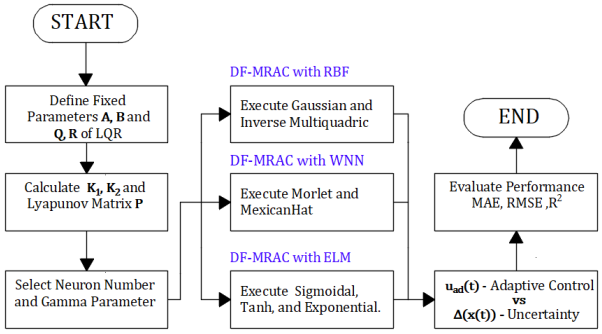

Fig. 4. Experimental Setup Flowchart

To evaluate the effectiveness of different neural network architectures in modeling system uncertainty, a grid search was conducted over neuron counts (1–100) and adaptation rates (1–1000) for each activation function. The detailed procedure is described in Algorithm 1.

Algorithm 1 Neural Network Model Selection

```
Initialization:
Define system dynamics, physical parameters, and control matrices (Q, R).
Compute controller gains (K1, K2), reference model (Am, Bm), and Lyapunov matrix (P).
Set simulation time and initialize result storage.
Parameter Sweep:
for model ∈ {RBF, WNN, ELM} do
    for activation ∈ model's set do
        for n = 1 to 100 do
            for γ = 1 to 1000 do
                Initialize weights and states.
                for each timestep do
                    Update r(t), β, Δx.
                    Compute control actions: un, uad, u.
                    Update states, weights.
                Store MAE, RMSE, R2.
Best Configuration Selection:
for each activation function do
    Select configuration with lowest mean RMSE.
    Record optimal parameters and metrics.
```

The non-linear uncertainty, $\Delta x(t) \in \mathbb{R}^{2\times 1}$, to be modeled by adaptive control follows the following formulation:

$$\Delta(x(t)) = \begin{bmatrix} sin(\theta) + cos(\dot{\theta}) + sin(\dot{Z}) \\ cos(\theta) + sin(\dot{\theta}) + cos(\dot{Z}) \end{bmatrix} \quad (23)$$

C. Evaluation Criteria and Performance Metrics

These metrics are presented in the following equations, where j is the number of observations, $\Delta(x(t))$ represents the actual observed values, and $u_{ad}(t)$ represents the values provided by the models.

The **Root Mean Square Error (RMSE)** measures the deviation of the data, calculated as the square root of the average of the squared errors. It penalizes larger errors, being particularly sensitive to large deviations:

$$RMSE = \sqrt{\frac{1}{j}\sum_{i=1}^{j}(\Delta(x(t)) - u_{ad}(t))^2} \quad (24)$$

The **Mean Absolute Error (MAE)** is defined as the average of the absolute differences between the observed and predicted values, as expressed by:

$$MAE = \frac{1}{j}\sum_{i=1}^{j}|\Delta(x(t)) - u_{ad}(t)| \quad (25)$$

Finally, the **Coefficient of Determination (R^2 Score)** assesses the degree to which the variability of the actual data is explained by the model, serving as a metric that expresses the overall fit of the model to the data, where $\Delta(\bar{x}(t))$ is the mean of the actual observed values:

$$R^2 = 1 - \frac{\sum_{i=1}^{j}(\Delta(x(t)) - u_{ad}(t))^2}{\sum_{i=1}^{j}(\Delta(x(t)) - \Delta(\bar{x}(t)))^2} \quad (26)$$

The evaluate of Δ_1 vs u_{ad_1} e Δ_2 vs u_{ad_2}, will be named as Output 1 and Output 2, on Table II. The average RMSE between Output 1 and Output 2 will be calculate as:

$$AVG_{RMSE} = \frac{RMSE_1 + RMSE_2}{2} \quad (27)$$

VII. Results

A. Adaptive Control and Uncertainty Approximation

The Table II summarizes the performance of the optimized configurations for each evaluated Artificial Neural Network (ANN) model. The configuration for each activation function was selected based on the lowest AVG_{RMSE}. The table presents, for each ANN and activation function combination, the number of neurons in the hidden layer (N) and the adaptation rate parameter (γ) of the selected configuration. Additionally, the metrics Mean Absolute Error (MAE), RMSE, and the coefficient of determination (R^2) are provided for Output 1 and Output 2.

TABLE II
EVALUATE PERFORMANCE OF EACH ANN

ANN	Function	N	γ	Output 1			Output 2		
				MAE	$RMSE$	R^2	MAE	$RMSE$	R^2
RBF	**Gaussian**	100	1000	0.0060	0.0386	0.8696	$8.6735{\cdot}10^{-4}$	0.0214	0.9329
	Inv. Multiq.	100	550	0.0068	0.0429	0.8389	$5.8062{\cdot}10^{-4}$	0.0185	0.9497
WNN	**Morlet**	25	1000	0.0056	0.0389	0.8677	0.0012	0.0229	0.9229
	MexicanHat	100	273	0.0067	0.0428	0.8396	$5.7093{\cdot}10^{-4}$	0.0185	0.9494
ELM	**Sigmoid**	12	921	0.0070	0.0427	0.8402	$5.9122{\cdot}10^{-4}$	0.0178	0.9535
	Tanh	31	708	0.0054	0.0379	1	$6.2974{\cdot}10^{-4}$	0.0176	0.9544
	Exponential	15	720	0.0058	0.0395	0.8632	$4.9952{\cdot}10^{-4}$	0.0168	0.9584

From this table, the best number of neurons for RBF and WNN (MexicanHat) was the maximum tested (100). ELM and WNN (Morlet) showed a smaller best N, varying from 12 to 31, indicating dependence on the specific neural network configuration used. The influence of the best gamma factor (γ) also varied. RBF (Gaussian) and WNN (Morlet) reached the maximum tested (1000). The other configurations, including RBF (Inv. Multiq.), WNN (MexicanHat), and ELM, had best γ values ranging from intermediate to high (between 550 and 921), showing variation across models.

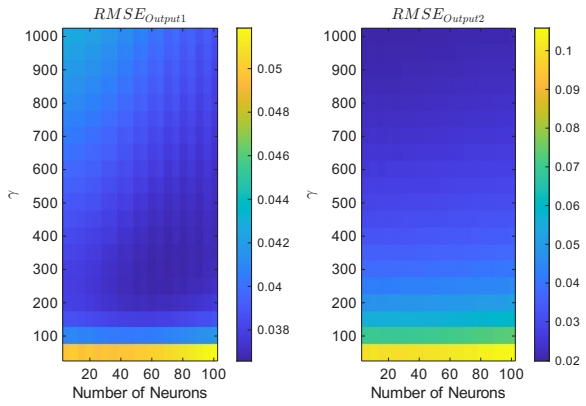

Fig. 5. RMSE (Output 1 and 2), for RBF with Gaussian function

For a more effective visual representation of the considerable data volume (700.000 results in total), a heatmap plot was selected. This choice is justified by its superior ability to illustrate data variation, preventing the information occlusion commonly observed in 3D graphics.

The RBF networks, Fig. 5 and 6, reveals distinct behaviors. Gaussian RBFs demonstrate robust performance with low RMSE across most configurations, exhibiting sensitivity primarily under extremely low adaptation rates. Inverse Multiquadric RBFs show optimal performance at lower adaptation rates, displaying slightly increased sensitivity to high γ values.

The WNNs present varied performance profiles. Morlet WNNs, Fig. 8, exhibit error localization, indicating parameter sensitivity, particularly high RMSE with concurrent high neuron counts and adaptation rates. In contrast, Mexican Hat, Fig. 7, maintain low RMSE across the parameter space, except for configurations combining low neuron number and low adaptation rates.

The ELMs show performance gradients linked to parameter levels, Fig. 10, 11 and 9. Sigmoid and Hyperbolic Tangent ELMs achieve minimal RMSE with low neuron number and low adaptation rates, with errors increasing as these parameters rise. Exponential ELMs generally yield low RMSE but present unpredictable localized high-error regions, suggesting higher sensitivity to specific hyperparameter combinations, especially at elevated neuron counts and adaptation rates γ.

The adaptive control components u_{ad_1} and u_{ad_2}, corresponding to the first and second components of the adaptive control law, are shown in Fig. 12. The solid green line

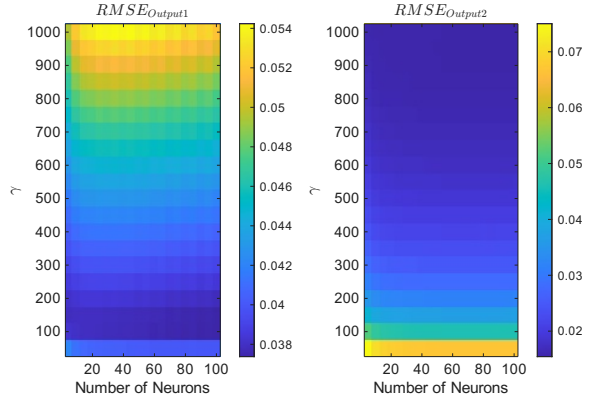

Fig. 6. RMSE (Output 1 and 2), for RBF with Inv. Multiq. function

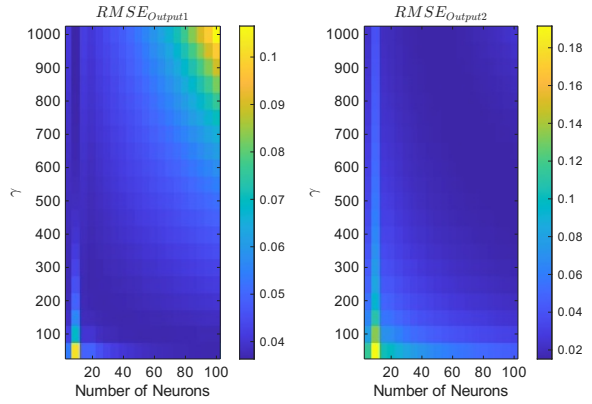

Fig. 7. RMSE (Output 1 and 2), for WNN with MexicanHat function

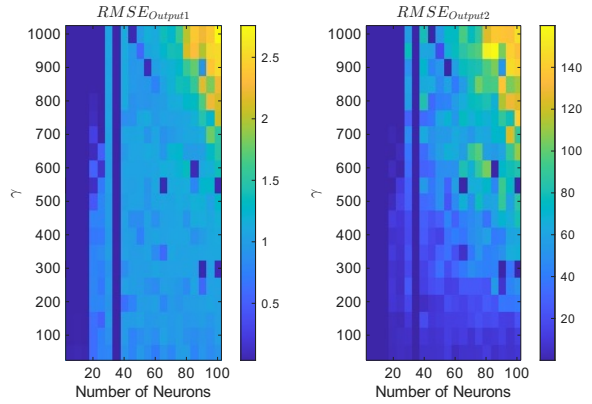

Fig. 8. RMSE (Output 1 and 2), for WNN with Morlet function

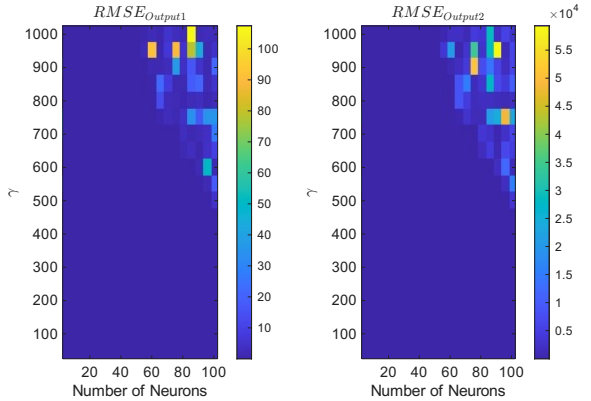

Fig. 9. RMSE (Output 1 and 2), for ELM with Exponential function

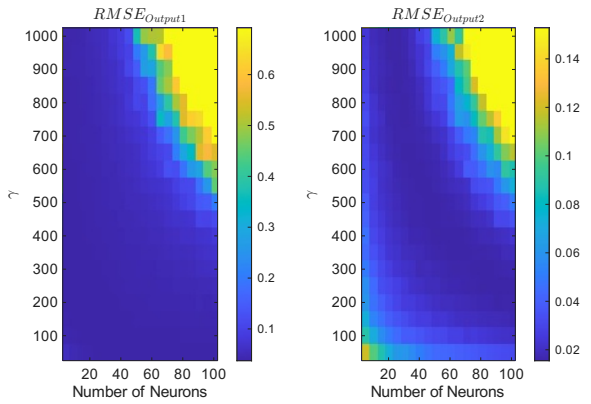

Fig. 10. RMSE (Output 1 and 2), for ELM with Sigmoidal function

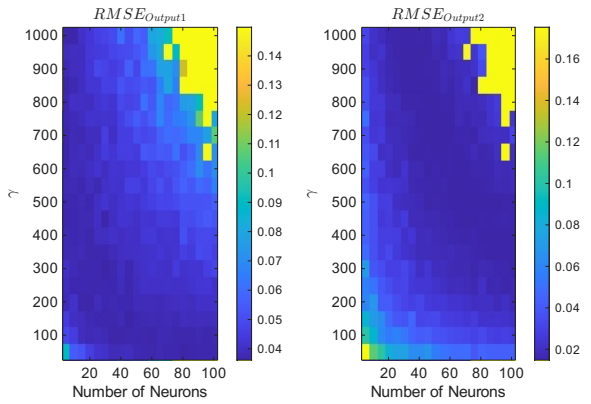

Fig. 11. RMSE (Output 1 and 2), for ELM with TanH function

represents the actual uncertainty $\Delta(x)$, while the dashed red line corresponds to the estimated uncertainty $\hat{W}^T\beta(x)$.

B. State Tracking

The Fig. 13 presents the time responses of the system states: pitch angle θ, pitch rate $\dot{\theta}$, and vertical drift rate $\dot{Z}$. The figure displays the actual state (black line) relative to the reference model (cyan line) and the pitch angle reference trajectory ($r(t)$, dashed red line) in the case of θ. The θ response tracks $r(t)$, while the $\dot{\theta}$ response follows the reference model dynamics with reduced transient oscillations. Alignment with the reference model is maintained for $\dot{Z}$, indicating low drift.

C. Control Inputs

The control input δ, representing the gimbal deflection angle, is shown in Fig. 14. The system effectively adjusts δ to ensure proper trajectory tracking while minimizing overshoot and oscillations. Fig. 14 also presents the input α_w, the wind-induced angle of attack. The system maintains smooth control behavior, adapting to disturbances and ensuring stability.

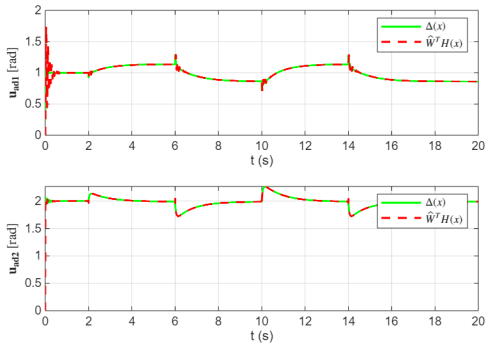

Fig. 12. Adaptive control (u_{ad_1} and u_{ad_2}) by tuning the uncertainty through ELM model with activation function = $tanh$, $N = 31$ and $\gamma = 708$.

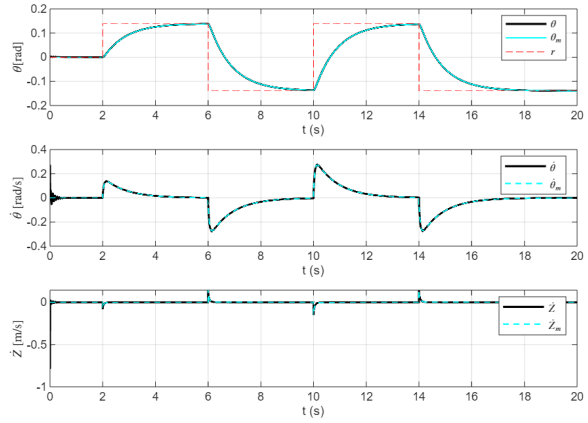

Fig. 13. θ Pitch angle track, θ_m of reference model and the reference signal r. Pitch rate $\dot{\theta}$ and $\dot{\theta}_m$ of reference model. Inertial drift velocity $\dot{Z}$ and $\dot{Z}_m$. For ELM model with activation function = $tanh$, $N = 31$ and $\gamma = 708$.

VIII. Conclusion

This study developed a DF-MRAC to finless launch vehicle attitude control using TVC and neural networks to estimate uncertainty. The comparative analysis of neural networks revealed that while all tested architectures proved viable, different hyperparameter configurations (number of neurons in the hidden layer N and adaptation rate γ) exhibited distinct sensitivities. Specifically, ELM networks with a Hyperbolic Tangent activation function and RBF networks with a Gaussian function achieved noteworthy performances with optimized configurations, displaying low RMSE and MAE values, and high R^2. It was also noted that some architectures, such as Morlet WNN and Exponential ELM, can present regions of higher error under specific combinations of N and γ, indicating the importance of careful parameter calibration.Future work will involve finalizing comparative studies with classical methods of uncertainty estimation. Furthermore, investigation into online hyperparameter optimization, such as the neuron's center parameters in the RBF, or hybrid neural methods for enhanced uncertainty and disturbance could be pursued.

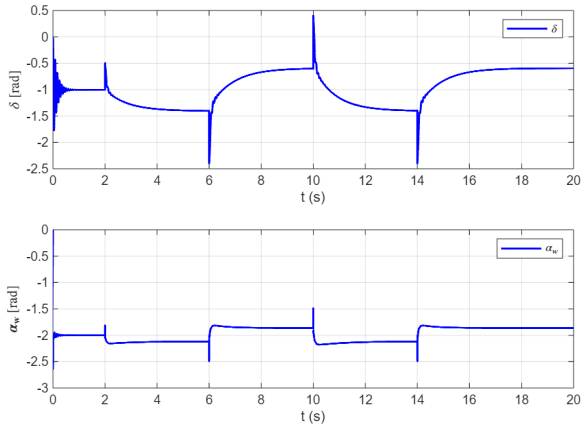

Fig. 14. Gimbal deflection angle δ and wind-induced angle of attack α. For ELM model with activation function = $tanh$, $N = 31$ and $\gamma = 708$.

References

[1] B. Wie, W. Du, and M. Whorton, "Analysis and design of launch vehicle flight control systems," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 6291.

[2] L. Sopegno, P. Livreri, M. Stefanovic, and K. P. Valavanis, "Linear quadratic regulator: A simple thrust vector control system for rockets," in *2022 30th Mediterranean Conference on Control and Automation (MED)*. IEEE, 2022, pp. 591–597.

[3] E. T. Falangas, *Performance evaluation and design of flight vehicle control systems*. John Wiley & Sons, 2015.

[4] E. Arabi, D. Panagou, T. Yucelen, and N. T. Nguyen, "Model reference adaptive control of uncertain dynamical systems subject to high-order actuator dynamics with performance guarantees," in *AIAA Scitech 2020 Forum*, 2020, p. 0591.

[5] B. C. Gruenwald, T. Yucelen, and J. A. Muse, "Direct uncertainty minimization framework for system performance improvement in model reference adaptive control," *Machines*, vol. 5, no. 1, p. 9, 2017.

[6] T. Yucelen and A. J. Calise, "Derivative-free model reference adaptive control," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 4, pp. 933–950, 2011.

[7] Z. Hou and X. Lan, "Adaptive sliding mode and rbf neural network based fault tolerant attitude control for spacecraft with unknown uncertainties and disturbances," *Advances in Space Research*, vol. 74, no. 4, pp. 1680–1692, 2024.

[8] N. T. Nguyen and N. T. Nguyen, *Model-reference adaptive control*. Springer, 2018.

[9] R. Chandramohan, T. Yucelen, A. Calise, G. Chowdhary, and E. Johnson, "Experimental evaluation of derivative free model reference adaptive control," in *AIAA Guidance, Navigation, and Control Conference*, 2010, p. 7862.

[10] E. Arabi, T. Yucelen, B. C. Gruenwald, M. Fravolini, S. Balakrishnan, and N. T. Nguyen, "A neuroadaptive architecture for model reference control of uncertain dynamical systems with performance guarantees," *Systems & Control Letters*, vol. 125, pp. 37–44, 2019.

[11] M.-D. Duong, V.-T. Nguyen, and Q.-T. Dao, "Adaptive control using radial basis function neural networks for pneumatic artificial muscle systems." *International Journal of Online & Biomedical Engineering*, vol. 20, no. 12, 2024.

[12] F. Jurado and S. Lopez, "A wavelet neural control scheme for a quadrotor unmanned aerial vehicle," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 376, no. 2126, p. 20170248, 2018.

[13] V. Panwar, "Wavelet neural network-based h_∞ trajectory tracking for robot manipulators using fast terminal sliding mode control," *Robotica*, vol. 35, no. 7, pp. 1488–1503, 2017.

[14] Y. Zhang, Z. Fang, and H. Li, "Extreme learning machine assisted adaptive control of a quadrotor helicopter," *Mathematical Problems in Engineering*, vol. 2015, no. 1, p. 905184, 2015.

[15] X.-Z. Jin, M.-M. Gao, W.-W. Che, and H. Wang, "Adaptive extreme learning machine-based event-triggered control for perturbed euler–lagrange systems," *International Journal of Robust and Nonlinear Control*, vol. 33, no. 7, pp. 4245–4261, 2023.