Nonlinear Control System with Reinforcement Learning and Neural Networks Based Lyapunov Functions

R. Rego and F. Araújo

***Abstract*—This paper deals with the problem of finding the control Lyapunov function (CLF) that keeps the system stable. Two feedforward artificial neural networks are proposed as a function approximator to generate a control Lyapunov function for a nonlinear dynamical system without any local approximation of their dynamics. Finding a CLF is not an easy task. Then, to determine the control Lyapunov function, this paper proposes the use of reinforcement learning with two artificial neural networks based on the Lyapunov stability theory. The proposed control is applied in two nonlinear dynamical systems. To analyze the stability of the systems, the region of attraction (ROA) of an asymptotically stable equilibrium point was used. The simulations show the good performance of the proposed technique and confirmed that reinforcement learning and neural networks are an excellent mathematical tool to deal with control design problems and may lead to better solutions to nonlinear problems in the design of stable controls without linear approximations.**

***Index Terms*—control Lyapunov function, neural network, reinforcement learing, intelligent control.**

I. Introdução

AS redes neurais (*Neural Network* - NN) são comumente conhecidas como aproximadores universais, e podem ser efetivamente aplicadas para lidar com a modelagem de incertezas para fins de controle robusto e aplicações de identificação e controle de sistemas [1]. A modelagem de uma rede neural foi proposta pela primeira vez por McCulloch e Pitts em meados da década de 1940 [2]. Já os primeiros modelos dinâmicos de redes neurais foram desenvolvidos no início dos anos 80 baseados no trabalho de David Rumelhart [3]. Uma importante rede dinâmica é a rede de Hopfield proposta por [4]. Além disso, as redes estáticas ou dinâmicas provaram sua utilidade na área de controle nas duas modelagens [5], [6], [7].

As NNs são excelentes ferramentas matemáticas para lidar com problemas lineares e não lineares, além disso, elas podem ser efetivamente aplicadas no controle de sistemas não lineares [7], [8]. Realizar o controle de um sistema não linear não é uma tarefa fácil, diversas técnicas já foram propostas, tais como, a realização de uma linearização do problema, aproximações polinomiais, reguladores quadráticos lineares e controladores preditivos [7], [6], [9].

No desenvolvimento do projeto de controle de um sistema dinâmico é necessário obter um ganho que garanta a estabilidade do sistema. A estabilidade de um sistema dinâmico não linear é uma questão matemática difícil de lidar. Geralmente quando deseja-se analisar a estabilidade deste sistema faz-se o uso do teorema de estabilidade de Lyapunov [10], [11]. O método de Lyapunov é amplamente utilizado para a análise de estabilidade de sistemas lineares e não lineares, invariantes e variantes no tempo [5], [12], [13], [14], [10]. Este método pode ser integrado na aprendizagem por reforço e rede neural para obtenção de um controlador estável para um dado sistema não linear.

Ao longo dos anos, a aprendizagem por reforço (*Reinforcement Learning* - RL) tem sido aplicada nos campos de inteligência artificial e aprendizado de máquina para resolver problemas ótimos de tomada de decisão e realização de controle inteligente de sistemas [15], [16], [17], [18], [19]. A abordagem do aprendizado por reforço baseada em redes neurais vem sendo utilizada para obtenção de controle ótimo [19]. As redes neurais e *deep learning* podem encontrar controladores comprovadamente estáveis de maneira direta e combater a não linearidade presente nos sistemas [20], [21], [22]. Dessa maneira é possível perceber que os trabalhos que utilizam NNs para controlar os sistemas de forma inteligente aumentaram nos últimos anos [23], [24], [25], [26].

Além disso, as redes neurais com funções de Lyapunov vêm sendo utilizadas para controle de sistemas não lineares sem nenhuma aproximação local de sua dinâmica [27], [20], [28], [29], [30], [31], [32]. No artigo [28] os autores estudaram o problema da análise de estabilidade assintótica para redes neurais com atrasos distribuídos baseado na função de Lyapunov-Krasovskii. Já em [29] foi proposto um novo tipo de rede com múltiplas funções de Lyapunov.

No artigo [30] os autores desenvolveram uma estratégia de design de controlador baseada em rede neural com função de base radial adaptativa que utiliza funções integrais de Lyapunov para uma classe de sistemas não lineares sujeitos a perturbações. Já [31] considerou o problema de controle adaptativo com rede neural para sistema não linear incerto comutado e as funções múltiplas de Lyapunov são utilizadas para desenvolver um procedimento de projeto recursivo do tipo *backstepping*. Os trabalhos [33] e [34] mostraram que os algoritmos baseados na teoria da função Lyapunov são eficazes na redução do tempo computacional e da carga computacional.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

R. C. B. Rego, Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Universidade Federal do Rio Grande do Norte, Brasil, rosana.rego@ufersa.edu.br

Fábio Meneghetti U. de Araújo, Departamento de Engenharia de Computação e Automação, Universidade Federal do Rio Grande do Norte, Brasil, meneghet@dca.ufrn.br

No artigo [20] foi proposto um método para obtenção da lei de controle baseada na aprendizagem de funções de Lyapunov com rede neural, com garantia comprovada de estabilidade. Em [35] os autores apresentaram uma técnica para aprender o controle com funções de Lyapunov, que são usadas por vez para sintetizar controladores em sistemas dinâmicos lineares e que podem estabilizar o sistema.

Dessa forma, neste trabalho é proposto:

(1) um algoritmo com aprendizado por reforço e duas redes neurais para gerar a função de controle de Lyapunov.
(2) uma análise da estabilidade do sistema por meio da região de atração, assim como descrito em [36], [20], [37].

A abordagem com RL é utilizada para aprender um sinal de controle inicial para o sistema; a rede com o método de Lyapunov é utilizada para gerar a função de Lyapunov, com um limite garantido na função de custo. A técnica descrita é baseada nos trabalhos [35], [20], [19], [32], entretanto o algoritmo proposto apresenta resultados mais eficientes quando comparado com [19] em termos de garantia de estabilidade do sistema.

O trabalho foi dividido da seguinte forma: Na seção 2 a formulação do problema a ser resolvido é apresentada e alguns conceitos de estabilidade de Lyapunov são relembrados. Na seção 3 a estratégia de controle é apresentada, bem como o algoritmo proposto. Na seção 4, os exemplos numéricos são apresentados. E finalmente, as considerações finais são comentadas na seção 5.

II. Formulação do problema

Noções preliminares, incluindo a teoria de estabilização para sistemas dinâmicos não lineares serão recordadas. Primeiro, é definido o modelo do sistema estudado ao longo deste artigo.

Definição 1: (Sistema controlado) Um sistema dinâmico n-dimensional controlado, é descrito por

$$\dot{x}(t) = f(x,u), \quad x(0) = x_0 \tag{1}$$

em que $f : \mathcal{D} \longrightarrow \mathbb{R}^n$ é um campo vetorial contínuo de Lipschitz, e $\mathcal{D} \subseteq \mathbb{R}^n$ é um conjunto aberto com $0 \in \mathcal{D}$ que define o espaço de estado do sistema, isto é, cada $x(t) \in \mathcal{D}$ é um vetor de estado. E o ganho é definido como uma função continua $u : \mathbb{R}^n \longrightarrow \mathbb{R}^m$.

Considerando o sistema (1), com $u(t) \in \mathfrak{U} \subseteq \mathbb{R}^m$. Em que, u é a entrada do sistema que pode ser manipulada para obter a resposta desejada minimizando a função,

$$J = \int_0^\infty V(x)dt. \tag{2}$$

$$V(z) = min\,\{J|\dot{x} = f(x,u), x(0) = z, u(t) \in \mathfrak{U}\} \tag{3}$$

Pode-se resolver esse problema via programação dinâmica, e então obter a lei de controle,

$$u^*(t) = \underset{w \in \mathfrak{U}}{argmin}\, \dot{V}(x,w), \tag{4}$$

mas, pode ser muito difícil de encontrar V e, portanto, u^*. Os projetos de controles não lineares podem ser realizáveis por meio das funções de Lyapunov [10]. Logo, suponha que existe uma função $V : \mathbb{R}^n \longrightarrow \mathbb{R}$, tal que

$$V(z) \geq 0\, \forall z \tag{5}$$

$$\forall z,\ \underset{w \in \mathcal{U}}{min} \dot{V}(z,w) \leq -V(z) \tag{6}$$

Então o controle em malha fechada pode ser dado por,

$$u(t) = g(x(t)), \tag{7}$$

com,

$$g(z) = \underset{w \in \mathfrak{U}}{argmin}\, \dot{V}(z,w), \tag{8}$$

resultando em $J \leq V(x(0))$, neste caso, V é chamada de função de controle de Lyapunov para o problema.

Encontrar a função V não é uma tarefa fácil, dessa forma, as redes neurais podem ser utilizadas para achar a função de Lyapunov que garanta a estabilidade de um dado sistema como já descrito em [20], [21], [22], [35], [37].

Antes de descrever a implementação das redes neurais, é necessário recordar algumas definições do conceito de estabilidade de sistemas, como descrito nas Definições 2, 3, 4 e 6.

Definição 2: (Estabilidade) Assumindo que o sistema (1) admite a origem como o ponto de equilíbrio. Esse ponto de equilíbrio é estável desde que cada constante $\epsilon > 0$, existe uma constante $\delta(\epsilon) > 0$, tal que para cada estado inicial $x_0 \in \mathcal{X} \cap \{x \in \mathbb{R} : x \leq \delta(\epsilon)\}$, e cada instante inicial de tempo $t_0 \geq 0$, uma solução única $x(t, t_0, x_0)$ satisfaz $|x(t, t_0, x_0)| \leq \epsilon\, \forall t \geq t_0$. Caso contrário, chamamos o equilíbrio instável [10].

Definição 3: (Estabilidade Assintótica) [10] O sistema (1) é assintoticamente estável na origem, se for estável e também

$$lim_{t \longrightarrow \infty} ||x(t)|| = 0\, \forall\, ||x(0)|| < \delta. \tag{9}$$

Definição 4: (Funções de Lyapunov para estabilidade assintótica) Considerando o sistema (1) com o ponto de equilíbrio na origem, isto é, $f(0) = 0$. Suponha que exista uma função continuamente diferenciável $V : \mathcal{D} \longrightarrow \mathbb{R}$ que satisfaz a condição,

$$V(0) = 0,\ \text{e, } \forall x \in \mathcal{D}\ \{0\}, V(x) > 0 \text{ e } \bigtriangledown_f V(x) < 0. \tag{10}$$

Em que $\bigtriangledown_f = \sum_{i=1}^n \frac{\partial V}{\partial x_i}[f]_i(x)$ é a derivada de Lie, que mede a taxa de variação de V. Portanto, o sistema é assintoticamente estável na origem e V é chamado de função de Lyapunov [38].

A análise da estabilidade assintótica não é suficiente para garantir a segurança em sistemas, dessa forma, a análise da região de atração (*Region of Attraction* - ROA) de um ponto de equilíbrio assintoticamente estável é de suma importância [37].

Definição 5: (Região de atração) Se $x = 0$ é um ponto de equilíbrio assintoticamente estável para o sistema (1), então a região de atração é definida por, $R_\delta = \{x \in \mathbb{R}^n | V(x) \leq \delta\}$ [37].

Uma região de atração define um conjunto invariável que é garantido para conter todas as trajetórias possíveis do sistema, e dessa forma garantir a estabilidade. O ponto de equilíbrio de uma ROA é um conjunto de todos os pontos, de forma que qualquer trajetória iniciada no estado inicial seja atraída para

o ponto de equilíbrio. No entanto, encontrar a ROA exata analiticamente pode ser difícil ou até impossível. Geralmente, as funções de Lyapunov são usadas para encontrar subestimativas da região de atração [37].

Definição 6: (Estabilidade global assintótica) O sistema (1) é globalmente assintoticamente estável se existe a função de Lyapunov $V(x)$ definida sobre $\mathbb{R}_+ \times \mathbb{R}^n$ com valores em $\mathbb{R}$, tal que para todo $x \in \mathbb{R}^n$ e todo t positivo

$$\alpha_1(\|x\|) \leq V(x) \leq \alpha_2(\|x\|), \tag{11}$$

$$\dot{V}(x) \leq -\alpha_3(\|x\|), \tag{12}$$

em que α_1, α_2 e α_3 são funções definidas sobre $\mathbb{R}_+$ com valores em $\mathbb{R}_+$, contínuas, estritamente crescentes, não limitadas e nulas na origem [39].

Encontrar a função de Lyapunov V que garanta a existência de uma lei de controle pode estabilizar todas as trajetórias para o equilíbrio [40], [35]. Portanto, as definições 2, 3, 4, 5 e 6 precisam ser incorporadas na função custo da rede neural, como já foi discutido em [20].

III. Técnica de controle

Nessa seção é descrito como uma rede neural pode aprender a função V e um sinal de controle $\mathcal{U}$, de forma que as condições de Lyapunov possam ser rigorosamente verificadas para garantir a estabilidade do sistema.

A. Aprendizagem por reforço

Considerando o projeto de um controlador que tenta controlar os estados $\mathcal{X} = [x_1, x_2, \cdots x_m]^T \in \mathbb{R}^m$ de uma planta executando ações $\mathcal{U} = [u_1, u_2, \cdots u_k]^T \in \mathbb{R}^k$ que dependem do estado. Quando o controlador executa uma ação $\mathcal{U}$ na planta nos estados $\mathcal{X}$, recebe uma recompensa $\mathcal{R}(\mathcal{X},\mathcal{U})$ que depende em geral tanto da ação de controle e do estado. Como resultado da ação realizada pelo controlador o sistema controlado transita para o próximo estado. A política $\pi^*(\mathcal{X})$ ou sequência de ação ideal é aquela que maximiza a recompensa esperada com desconto acumulado dada por,

$$\mathcal{W}^{\pi}(\mathcal{X}) = E[\mathcal{R}(\mathcal{X}_0) + \cdots + \gamma^n \mathcal{R}(\mathcal{X}_n)], \tag{13}$$

com $n = 1, \cdots, N$, em que N é a quantidade de amostras.

A recompensa com desconto acumulada a partir do estado $\mathcal{X}_0$ é dada por $\mathcal{W}^{\pi}(\mathcal{X})$, uma vez que depende desse estado inicial e também na sequência de ações executadas em cada estado. γ é uma constante que garante convergência da sequência infinita [19].

Assim, a implementação do aprendizado por reforço começa com a definição do processo de decisão de Markov (*Markov Decision Process* - MDP). O processo de decisão de Markov para o sistema é definido pela 5-tupla $(\mathcal{X},\mathcal{U},\mathcal{P},\gamma,\mathcal{R})$, onde $\mathcal{X}$ é o vetor de estados do sistema, $\mathcal{U}$ é o conjunto de todos as ações de controle possíveis, $\mathcal{P}$ é a probabilidade de transição dos estados.

A função de compensação de acordo com [19], pode ser definida como,

$$\mathcal{R}(\mathcal{X}) = -\beta||\mathcal{X}_r - \mathcal{X}||. \tag{14}$$

O objetivo do RL é escolher ações ao longo do tempo para maximizar o valor de $\mathcal{W}^{\pi}(\mathcal{X})$. Esta função define o valor esperado da soma das recompensas com o desconto que o controlador receberá ao executar uma política fixa, começando do estado $\mathcal{X}_0$ até atingir o valor desejado, isto é, a referência definida.

A função de valor ótimo descrita a seguir, é o valor alcançado quando a política ideal/ótima é seguida pelo controlador,

$$\mathcal{W}^*(\mathcal{X}) = max_{\pi}\ \mathcal{W}^{\pi}(\mathcal{X}) = \mathcal{R}(\mathcal{X}) + max_{\mathcal{U}\in\mathfrak{U}}\ \gamma\mathcal{W}^{\pi}(\mathcal{X}'). \tag{15}$$

A política que otimiza a função (15) em qualquer estado é conhecida como a política ótima, definida por,

$$\pi^*(\mathcal{X}) = argmax_{\mathcal{U}\in\mathfrak{U}}\ \mathcal{W}^*(\mathcal{X}'). \tag{16}$$

O aprendizado por reforço descrito aqui é baseado em [19], em que os autores utilizam RL com uma rede de função de base radial (*Radial Basis Function* - RBF) para o controle de nível de um sistema de tanques. Porém, neste trabalho, o aprendizado por reforço é proposto junto com uma rede *feedforward* multicamadas com função de ativação *tanh*, em que esta é utilizada para aprender diretamente a função (15), e o sinal de controle inicial para o sistema não linear. Além disso, uma outra rede neural *feedforward* é utilizada para geração da função de Lyapunov para o sistema, como é descrito na seção seguinte.

B. Geração de funções de Lyapunov por redes neurais

Para a geração das funções de Lyapunov, é proposto uma rede neural perceptron multicamadas (Multi Layer Perceptron - MLP) *feedforward* com uma camada oculta, como mostra a Figura 1.

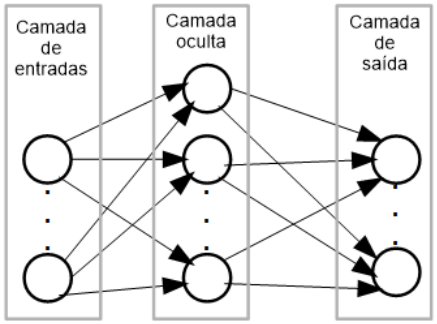


Fig. 1. Representação da rede MLP *feedforward*.

A função a ser encontrada é baseada na parametrização. Mais especificamente, uma classe de função $V_c(\mathcal{X})$ é parametrizada por um conjunto de parâmetros desconhecidos c. Essa parametrização é geralmente especificada como uma combinação linear de funções básicas da forma,

$$V_c(\mathcal{X}) := \sum c_i g_i(\mathcal{X}). \tag{17}$$

As funções g_i geralmente abrangem todos os monômios possíveis até um limite de grau pré-especificado.

Para encontrar V_c uma rede neural será utilizada, em que a entrada será qualquer vetor de estado do sistema (1). De acordo com a Figura 2, considerando a rede com k camadas a função de Lyapunov pode ser descrita por,

$$V_{c,k+1}(\mathcal{X}) = \varphi(W_{k+1}V_{c,k}(\mathcal{X}) + B_{k+1}), \tag{18}$$

em que $z = V_{c,k}(\mathcal{X}) = \varphi(W_k\mathcal{X} + B_k)$, W_k é o vetor de pesos correspondente a camada k e B_k o vetor de bias. $\varphi(.)$ é a função de ativação.

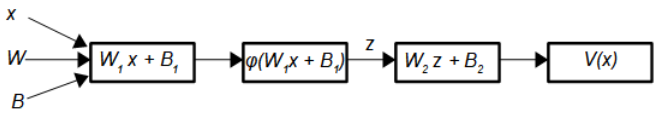


Fig. 2. Abstração de uma rede neural com uma camada oculta para geração da função de Lyapunov.

c será utilizado para denotar o vetor de parâmetro da função de Lypaunov V_c candidata. E $\mathcal{U}$ denota a função de controle do sistema. O processo de aprendizado atualiza os parâmetros c e $\mathcal{U}$ para melhorar a probabilidade de satisfazer as condições de Lyapunov, que será formulada como uma função custo chamado risco de Lyapunov. Conceitualmente, o problema geral do projeto de controle baseado nas definições de Lyapunov consiste em minimizar a função custo [20],

$$\inf_{c,\mathcal{U}} \sup_{\mathcal{X}\in\mathcal{D}} \left(max(0, -V_c(\mathcal{X})) + max(0, \bigtriangledown_f V_c(\mathcal{X})) + V_c^2(0)\right). \tag{19}$$

Considere uma função Lyapunov candidata V_c para um sistema dinâmico controlado (1). A função risco Lyapunov é definida por,

$$\begin{aligned} L_\rho(c,\mathcal{U}) = \mathrm{E}_{x\sim\rho(\mathcal{D})}(max(0, -V_c(\mathcal{X})) \\ +max(0, \bigtriangledown_f V_c(\mathcal{X})) + V_c^2(0)), \end{aligned} \tag{20}$$

em que $\mathcal{X}$ é uma variável aleatória no espaço de estados do sistema com uma distribuição ρ [20].

Observe que V_c e f são altamente não-lineares, portanto, $L(c,\mathcal{U})$ será complexa. Mas, as redes *feedforward* multicamadas e o algoritmo gradiente descendente estocástico podem produzir rapidamente as funções Lyapunov generalizáveis com boas propriedades geométricas como mostrado em [20], [12].

Para cada controle $\mathcal{U}$ e função de Lyapunov V_c obtida, implementou-se um verificador para encontrar estados que violem as condições de estabilidade de Lyapunov. A restrição do verificador é a seguinte fórmula lógica [20],

$$\Upsilon_\epsilon(\mathcal{X}) := \left(||\mathcal{X}||_2^2 \geq \epsilon\right) \bigwedge \left(V(\mathcal{X}) \leq 0 \bigvee \bigtriangledown_f V(\mathcal{X}) \geq 0\right) \tag{21}$$

em que $\mathcal{X}$ é delimitado no espaço de estado $\mathcal{D}$ do sistema. O parâmetro de erro numérico ϵ é explicitamente introduzido para controlar a sensibilidade numérica próxima à origem. Todas as trajetórias devem convergir assintoticamente para a origem.

A Figura 3 mostra a visão geral da rede e do verificador implementados para gerar a função de Lyapunov que estabiliza o sistema.

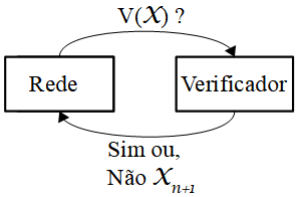


Fig. 3. Visão geral da rede e do verificador utilizado para aprender a função de Lyapunov.

O verificador resolve um problema de viabilidade de restrições não lineares, onde tenta encontrar um estado que viola o critério de estabilidade de Lyapunov, dada a atual função de controle de Lyapunov. Se não for encontrado nenhum estado que viole as condições de estabilidade de Lyapunov, é garantido que a função encontrada estabiliza o sistema.

C. Algoritmo

A estrutura geral de aprendizado é iterativa, eliminando um conjunto de candidatos em cada iteração. O algoritmo proposto é mostrado na Tabela I.

Tabela I
ALGORITMO PROPOSTO COM APRENDIZADO POR REFORÇO E REDES NEURAIS PARA GERAR A FUNÇÃO DE LYAPUNOV.

Algoritmo RL com NN-Lyapunov
Entradas: sistema, γ, α, β, N, i_{max}
$\mathcal{X}$,$\mathcal{U}$:= $random()$
$\mathcal{X}_r$:= referência
para i:=1 até i_{max} faça
para j:=1 até N faça
$\mathcal{W}(\mathcal{X}) \leftarrow \mathcal{R}(\mathcal{X}) + max_{\mathcal{U}\in\mathfrak{U}}\ \gamma\mathcal{W}^\pi(\mathcal{X}')$
fimpara
fimpara
para i:=1 até N faça
$\pi^*(\mathcal{X}) \leftarrow argmax_{\mathcal{U}\in\mathfrak{U}}\ \mathcal{W}^*(\mathcal{X}')$
fimpara
$\bar{\pi}^*(\mathcal{X}) \leftarrow RedeNeural(\mathcal{X}, \mathcal{X}_r, \pi^*)$
//Use $\bar{\pi}^*(\mathcal{X})$ para inciar o treinamento da NN-Lyapunov
enquanto count $< i_{max}$ faça
enquanto V_c não válida faça
V_c,$\mathcal{U} \leftarrow RedeNeural(\mathcal{X}, \pi^*(\mathcal{X}))$ //Calcula V_c e $\mathcal{U}$
$\bigtriangledown_f V_c(\mathcal{X}) \leftarrow calculaLieDerivada()$
$L_\rho(c,\mathcal{U}) \leftarrow \mathrm{E}_{\mathcal{X}\sim\rho(\mathcal{D})}(max(0, -V_c(\mathcal{X}))$
$+max(0, \bigtriangledown_f V_c(\mathcal{X})) + V_c^2(0))$
$c \leftarrow c + \alpha \bigtriangledown_c L(c,\mathcal{U})$ //Atualiza c
$\mathcal{U} \leftarrow \mathcal{U} + \alpha \bigtriangledown_\mathcal{U} L(c,\mathcal{U})$ //Atualiza $\mathcal{U}$
$\mathcal{X} \leftarrow sistema(\mathcal{U}, V_c)$ //Atualiza $\mathcal{X}$
para i:= 1 até n faça
$\Upsilon_\epsilon \leftarrow \Upsilon_\epsilon + \mathcal{X}_i^2$
fimpara
se $\Upsilon_\epsilon \geq \epsilon\, E\, (V_c \leq 0\, OU\ \bigtriangledown_f V_c \geq 0)$
Escreva(Função V_c válida) //A função satisfaz a teoria
Pare //Fim da busca
fimse
count:= count +1
fimenquanto
fimenquanto

A Figura 4 mostra a estrutura das redes neurais implementadas. Em que o algoritmo faz o uso de duas redes neurais *feedforward*. Para ambas redes, fez-se o uso das funções de ativação $tanh$.

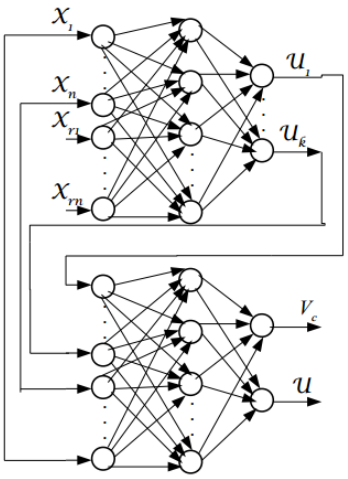


Fig. 4. Estrutura de redes proposta.

IV. Simulação Numérica

Para testar o algoritmo foi utilizado o modelo de duas plantas não lineares: o pêndulo invertido e o sistema barra bola. A simulação numérica foi realizada em python no ambiente *JupyterLab*. A técnica proposta neste trabalho (RL-NN) é comparada apenas com a técnica de aprendizado por reforço (RL) proposta em [19]. Para ambos os sistemas na implementação da rede 1, fez-se o uso de 10 neurônios na camada oculta, em que as entradas são os estados do sistema e a saída é o controle ótimo inicial. Já na implementação da rede 2, fez-se o uso de 6 neurônios na camada oculta, em que as entradas desta rede são o sinal de controle obtido na rede 1 e os estados do sistema, e a saída será a função de Lyapunov. Atribuiu-se uma taxa de aprendizado de 0.01 para ambas redes.

a) Pêndulo Invertido: É um problema de manter o pêndulo invertido na posição vertical. A dinâmica deste sistema é dada por,

$$\ddot{\Theta} = \frac{m\,g\,l\,sen(\Theta) + u - \varepsilon\dot{\Theta}}{m\,l^2}. \tag{22}$$

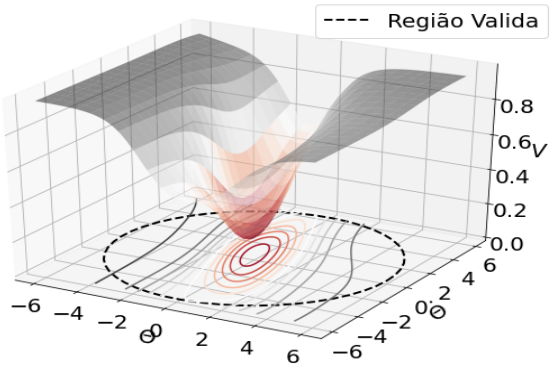
Em que Θ representa a posição angular do pêndulo invertido e $\dot{\Theta}$ representa a velocidade angular. O sinal de controle é denotado por u. Para simulação foi considerado $g = 9.81m/s^2$, $m = 1.5 \times 10^{-4}kg$, $\varepsilon = 0.1$ e $l = 0.5m$. $\mathcal{X}_0 = [\Theta_0\ \dot{\Theta}_0]^T = [0.5\ 0.5]^T$. Os parâmetros das redes foram: $\gamma = 0.99$, $\beta = 0.1$, $i_{max} = 1000$.

O ganho obtido via RL-NN e RL foi respectivamente,

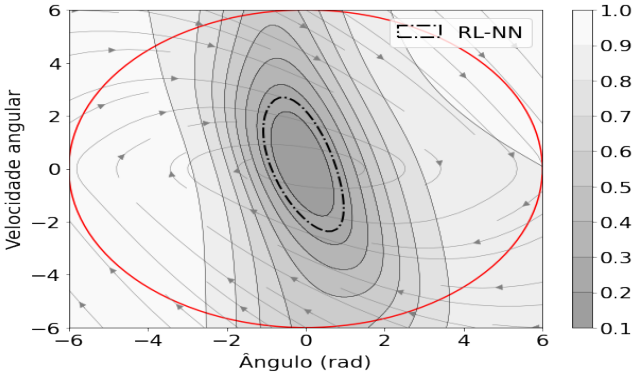
$$K_{RL-NN} = [-12.8405\ \ -7.9785]^T, \tag{23}$$

$$K_{RL} = [-6\ \ -6.43]^T. \tag{24}$$

Na Figura 5 (a) tem-se a função de Lyapunov obtida com a aplicação do algoritmo RL-NN. Na Figura 5 (b) tem-se a região de atração, que é definida por $R_\delta = \{V_c(\mathcal{X}) \leq \delta\} \subseteq \mathcal{D}$. A elipse vermelha é a região válida definida, isto é, $||\mathcal{X}|| \leq 6$. A região preta tracejada é a região de atração obtida com a função de Lypaunov gerada pela rede neural. Dessa forma, por meio da região de atração é possível verificar que a estabilidade do sistema será garantida com a lei de controle encontrada.



(a) Função de Lyapunov



(b) Região de atração

Fig. 5. Pêndulo invertido: (a) função V_c obtida via algoritmo 1; (b) região de atração obtida com a função V_c.

Na Figura 6 são mostrados os estados do sistema com o controle em malha fechada. É possível observar que o controle proposto RL-NN apresenta uma resposta mais rápida, quando comparado com o controle apenas com RL.

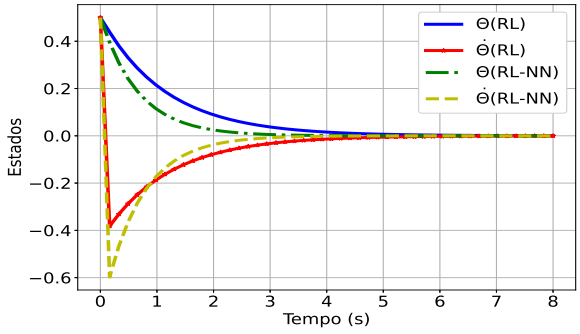


Fig. 6. Estados com a técnica RL, e com a técnica RL-NN proposta.

Já na Figura 7 tem-se os sinais de controle obtidos via aplicação de ambas as técnicas. É possível observar que a simulação com o algoritmo proposto obteve melhores resultados, além de garantir a estabilidade do sistema.

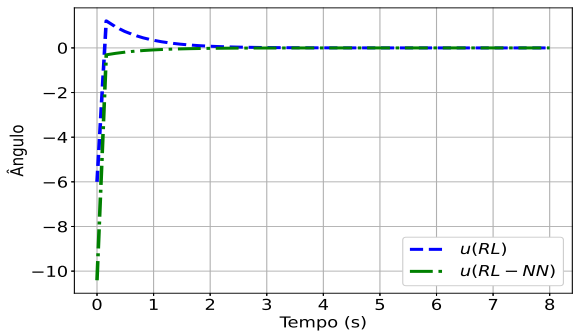


Fig. 7. Sinal de controle obtido com a técnica RL, e com a técnica RL-NN proposta.

b) Barra bola: A dinâmica do sistema barra bola (*ball-beam*), considerando apenas a equação não linear do movimento da bola e da barra, é dada por,

$$\ddot{x} = \frac{m_b\, g\, sen(\phi)\, r_b^2 r}{L(m_b\, r_b^2 + J_b)}. \tag{25}$$

Em que, $x = x_1$ representa a posição da bola na barra, e $\dot{x} = x_2$ representa a velocidade, o sinal de controle ϕ é o ângulo do motor. L é o comprimento da barra, m é a massa da bola, r_b é o raio da bola, g é a gravidade e J o momento de inércia de uma esfera sólida. Para simulação considerou-se os valores $L = 1m$, $r = 0.01m$, $r_b = 5 \times 10^{-4}m$, $m = 1.5 \times 10^{-4}kg$, $g = 9.81m/s^2$, $J_b = \frac{2mr_b^2}{5}$. Na simulação foi setado a posição desejada da bola como $x_r = 0.50m$. Os parâmetros das redes foram: $\gamma = 0.99$, $\beta = 0.1$, $i_{max} = 4000$.

O ganho obtido via RL-NN e RL foi respectivamente,

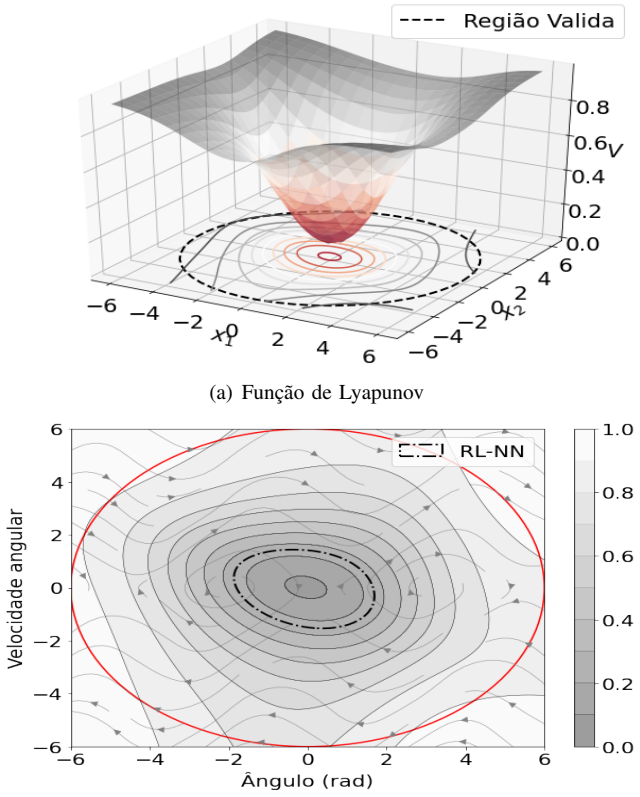
$$K_{RL-NN} = [1.2473\ \ 4.3248]^T, \tag{26}$$

$$K_{RL} = [0.9824\ \ 1]^T. \tag{27}$$

A Figura 8 (a) mostra a função de Lyapunov obtida via aplicação do algoritmo 1 proposto. Na Figura 8 (b) tem-se a região de atração do sistema barra bola, observar-se que a região de atração obtida com a função de Lyapunov gerada pela rede está dentro da região válida delimitada, dessa forma, a estabilidade do sistema é garantida.

Na Figura 9 (a) tem-se a saída do sistema barra bola com as técnicas aplicadas. Já na Figura 9 (b) tem-se os sinais de controle. Com ambas as técnicas o sistema seguiu a referência definida, entretanto, é possível verificar que a técnica RL apresentou uma resposta mais rápida. Na Figura 9 (c) tem-se a velocidade (estado x_2) do sistema. Observando a Figura 9 (a,b,c), percebe-se que o controle obtido via técnica RL-NN fornece uma resposta mais lenta, consequentemente mais suave. Em contraste, a técnica RL fornece uma resposta mais rápida para este sistema, entretanto o controle ótimo obtido via RL não garante a estabilidade do sistema em todas as trajetórias possíveis.

Além disso, o artigo fez o uso das duas plantas de dinâmicas distintas (pêndulo invertido e barra bola) justamente para mostrar que para diferentes dinâmicas as redes neurais conseguem



(a) Função de Lyapunov

(b) Região de atração

Fig. 8. Sistema Barra bola: (a) função V_c obtida via algoritmo 1; (b) região de atração obtida com a função V_c.

obter uma função de controle de Lyapunov que garante a estabilidade do sistema.

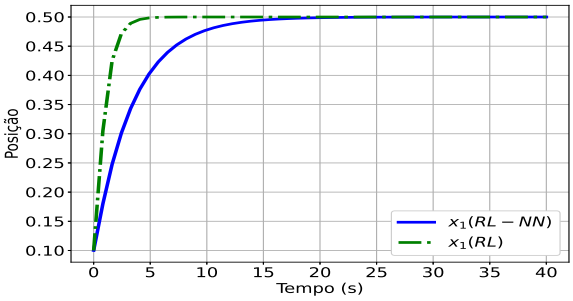
As estatísticas de execução do algoritmo proposto, para ambas as plantas não lineares, são mostradas na Tabela II, em que tem-se a quantidade de iterações realizadas até a rede encontrar o sinal de controle e a função de Lyapunov que estabiliza o sistema. O tempo de execução gasto e a quantidade de amostras utilizadas para ambos os sistemas. Para medir o tempo, fez-se o uso da função $default\_timer()$ pertencente a biblioteca $timeit$.

Tabela II
ESTATÍSTICAS DE TEMPO DE EXECUÇÃO DO ALGORITMO 1

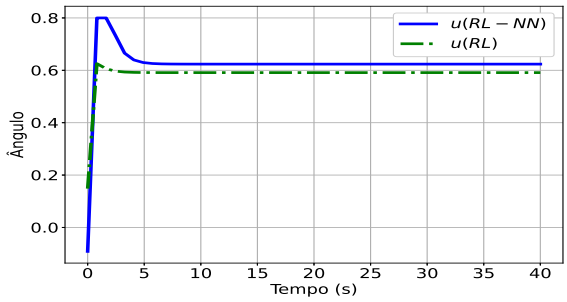
Sistema	Iterações	Tempo	Número de amostras
Pêndulo invertido	760	61.5736 s	500
Barra bola	680	312.5518 s	500

V. CONCLUSÃO

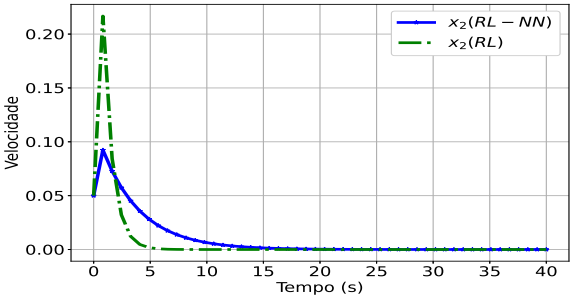
Foi demonstrado que o algoritmo proposto encontra uma função numérica de Lyapunov e uma estimativa da região de atração de um dado sistema, garantindo a estabilidade do mesmo. Para demonstração fez-se o uso de dois sistemas não lineares. Por meio da região de atração foi possível verificar que as condições de Lyapunov são garantidas em todos os conjuntos de estados dentro do círculo vermelho nos gráficos. Diferente dos trabalhos [20], [19], o algoritmo proposto faz o



(a) Posição da bola



(b) Sinal de controle



(c) Velocidade

Fig. 9. Sistema Barra bola

uso aprendizado por reforço para encontrar uma política ótima de controle e posteriormente a função de Lyapunov para um dado sistema.

O algoritmo proposto é interrompido assim que a função de Lyapunov é descoberta. No entanto, nenhuma reivindicação é feita quanto à otimização da função. Uma extensão importante deste trabalho seria encontrar uma função de Lyapunov para qual os controladores resultantes otimizem algumas métricas de desempenho do sistema, tais como resposta não saturada e rápida. O desafio é especificar essas performances como funções dos coeficientes da função de Lyapunov. Ainda, como trabalhos futuros, os autores pretendem fazer o uso do algoritmo *Q-Learning* e realizar o estudo dos aspectos de robustez da técnica, além de comparar com outras técnicas existentes.

Porém, através das simulações é possível prever que as redes neurais e *deep learning* poderão levar a melhores soluções para os problemas não lineares no projeto de controles estáveis sem aproximações lineares.

REFERÊNCIAS

[1] J. Liu, *Radial Basis Function (RBF) neural network control for mechanical systems: design, analysis and Matlab simulation*. Springer Science & Business Media, 2013.

[2] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[3] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[4] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[5] H. Simon, *Neural networks: a comprehensive foundation*. Prentice hall, 1999.

[6] O. Nelles, *Nonlinear system identification: from classical approaches to neural networks and fuzzy models*. Springer Science & Business Media, 2013.

[7] K. Patan, *Robust and Fault-Tolerant Control: Neural-Network-Based Solutions*. Springer, 2019, vol. 197.

[8] M. T. Hagan, H. B. Demuth, and O. D. Jesús, "An introduction to the use of neural networks in control systems," *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, vol. 12, no. 11, pp. 959–985, 2002.

[9] R. C. B. Rego and M. V. S. Costa, "Output feedback robust control with anti-windup applied to the 3ssc boost converter," *IEEE Latin America Transactions*, vol. 18, no. 05, pp. 874–880, 2020.

[10] M. Malisoff and F. Mazenc, *Constructions of strict Lyapunov functions*. Springer Science & Business Media, 2009.

[11] H. Ravanbakhsh and S. Sankaranarayanan, "Counterexample guided synthesis of switched controllers for reach-while-stay properties," *arXiv preprint arXiv:1505.01180*, 2015.

[12] S. M. Richards, F. Berkenkamp, and A. Krause, "The lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems," in *Proceedings of the 2nd Conference on Robot Learning (CoRL)*, vol. 87. PMLR, 2018, pp. 466–476.

[13] M. SaleheenAftab and F. Aftab, "A study on lyapunov function backpropagation algorithm for suitability as neuro-adaptive inverse controller," in *2018 IEEE 21st International Multi-Topic Conference (INMIC)*. IEEE, 2018, pp. 1–4.

[14] H.-H. Lian, S.-P. Xiao, H. Yan, F. Yang, and H.-B. Zeng, "Dissipativity analysis for neural networks with time-varying delays via a delay-product-type lyapunov functional approach," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[15] H. Li, Y. Wu, and M. Chen, "Adaptive fault-tolerant tracking control for discrete-time multiagent systems via reinforcement learning algorithm," *IEEE Transactions on Cybernetics*, pp. 1–12, 2020.

[16] Y. Liu, S. Li, S. Tong, and C. L. P. Chen, "Adaptive reinforcement learning control based on neural approximation for nonlinear discrete-time systems with unknown nonaffine dead-zone input," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 1, pp. 295–305, 2019.

[17] W. Sun, G. Zhao, and Y. Peng, "Adaptive optimal output feedback tracking control for unknown discrete-time linear systems using a combined reinforcement q-learning and internal model method," *IET Control Theory Applications*, vol. 13, no. 18, pp. 3075–3086, 2019.

[18] A. P. Valadbeigi, A. K. Sedigh, and F. L. Lewis, "h_∞ static output-feedback control design for discrete-time systems using reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 2, pp. 396–406, 2020.

[19] M. M. Noel and B. J. Pandian, "Control of a nonlinear liquid level system using a new artificial neural network based reinforcement learning approach," *Applied Soft Computing*, vol. 23, pp. 444–451, 2014.

[20] Y.-C. Chang, N. Roohi, and S. Gao, "Neural lyapunov control," in *Advances in Neural Information Processing Systems*, 2019, pp. 3240–3249.

[21] L. Xiao, Y. Zhang, Z. Hu, and J. Dai, "Performance benefits of robust nonlinear zeroing neural network for finding accurate solution of lyapunov equation in presence of various noises," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 9, pp. 5161–5171, 2019.

[22] K. H. Lim, K. P. Seng, L.-M. Ang, and S. W. Chin, "Lyapunov theory-based multilayered neural network," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 56, no. 4, pp. 305–309, 2009.

[23] Z. Li, Y. Xia, C. Su, J. Deng, J. Fu, and W. He, "Missile guidance law based on robust model predictive control using neural-network

optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 8, pp. 1803–1809, 2015.
[24] T. Wang, H. Gao, and J. Qiu, "A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 2, pp. 416–425, 2016.
[25] S. Li, Y. Zhang, and L. Jin, "Kinematic control of redundant manipulators using neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 10, pp. 2243–2254, 2017.
[26] S. Li, Z. Shao, and Y. Guan, "A dynamic neural network approach for efficient control of manipulators," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 5, pp. 932–941, 2019.
[27] G. Zhang, M. Yao, J. Xu, and W. Zhang, "Robust neural event-triggered control for dynamic positioning ships with actuator faults," *Ocean Engineering*, vol. 207, p. 107292, 2020.
[28] B. Zhang, J. Lam, and S. Xu, "Stability analysis of distributed delay neural networks based on relaxed lyapunov–krasovskii functionals," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 7, pp. 1480–1492, 2015.
[29] G. Wen, W. Yu, G. Hu, J. Cao, and X. Yu, "Pinning synchronization of directed networks with switching topologies: A multiple lyapunov functions approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3239–3250, 2015.
[30] X. Wang, B. Niu, G. Wu, J. Li, P. Duan, and D. Yang, "Adaptive rbf neural-network-based design strategy for non-strict-feedback nonlinear systems by using integral lyapunov functions," *IEEE Access*, vol. 6, pp. 75 076–75 085, 2018.
[31] B. Niu, Y. Liu, W. Zhou, H. Li, P. Duan, and J. Li, "Multiple lyapunov functions for adaptive neural tracking control of switched nonlinear nonlower-triangular systems," *IEEE Transactions on Cybernetics*, vol. 50, no. 5, pp. 1877–1886, 2020.
[32] Y. Chen, Y. Shi, and B. Zhang, "Optimal control via neural networks: A convex approach," in *International Conference on Learning Representations*, 2018.
[33] M. Ballesteros, I. Chairez, and A. Poznyak, "Robust min–max optimal control design for systems with uncertain models: A neural dynamic programming approach," *Neural Networks*, vol. 125, pp. 153–164, 2020.
[34] M. Taimoor and L. Aijun, "Lyapunov theory based adaptive neural observers design for aircraft sensors fault detection and isolation," *Journal of Intelligent & Robotic Systems*, pp. 1–13, 2019.
[35] H. Ravanbakhsh and S. Sankaranarayanan, "Learning control lyapunov functions from counterexamples and demonstrations," *Autonomous Robots*, vol. 43, no. 2, pp. 275–307, 2019.
[36] G. Valmorbida and J. Anderson, "Region of attraction analysis via invariant sets," in *2014 American control conference*. IEEE, 2014, pp. 3591–3596.
[37] M. Wu, Z. Yang, and W. Lin, "Exact asymptotic stability analysis and region-of-attraction estimation for nonlinear systems," in *Abstract and Applied Analysis*, vol. 2013. Hindawi, 2013.
[38] H. Kwakernaak and R. Sivan, *Linear optimal control systems*. Wiley-interscience New York, 1972, vol. 1.
[39] C. A. Desoer and M. Vidyasagar, *Feedback systems: input-output properties*. Siam, 1975, vol. 55.
[40] Z. Arstein, "Stabilization with relaxed controls," *Nonlinear analysis*, vol. 7, no. 11, pp. 1163–1173, 1983.

Fábio Meneghetti U. de Araújo Possui graduação em Graduação em Engenharia Mecânica pela Universidade Federal da Paraíba (1995), mestrado em Engenharia Mecânica pela Universidade Federal da Paraíba (1998) e doutorado em Engenharia Eletrônica e Computação pelo Instituto Tecnológico de Aeronáutica (2002). Atualmente é Professor Associado da Universidade Federal do Rio Grande do Norte. Tem experiência na área de Engenharia Elétrica e de Computação, com ênfase em Sistemas de Controle, atuando principalmente nos seguintes temas: Controle de Processos, Automação Industrial e Inteligência Artificial.

Rosana Cibely B. Rego Possui graduação de bacharel em Ciência e Tecnologia (2015) e Bacharel em Engenharia de Computação pela Universidade Federal Rural do Semi-Árido (2017). Mestrado em Engenharia Elétrica pela Universidade Federal Rural do Semi-Árido (2019). Atualmente estudante de Doutorado em Engenharia Elétrica na Universidade Federal do Rio Grande do Norte, com pesquisas na área de Controle Inteligente, controle Neural, redes neurais. Possui conhecimentos na área de programação (C/C++, Java, Python, Fortran, MatLab/Scilab) e eletrônica digital/microcontrolada, lógico programável e analógica.