Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

Controle Inteligente de Robôs Omnidirecionais Utilizando Redes Neurais Recorrentes

Lucas Solano Cadengue

Natal, RN

11 de Março de 2022

Universidade Federal do Rio Grande do Norte
Centro de Tecnologia
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

Controle Inteligente de Robôs Omnidirecionais Utilizando Redes Neurais Recorrentes

Lucas Solano Cadengue

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação (PPGEEC) da Universidade Federal do Rio Grande do Norte como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica e de Computação, orientado pelo Prof. D.Sc. Wallace Moreira Bessa.

Natal, RN

11 de Março de 2022

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Central Zila Mamede

Cadengue, Lucas Solano.
Controle Inteligente de robôs omnidirecionais utilizando redes neurais recorrentes / Lucas Solano Cadengue. - 2022.
51 f.: il.

Dissertação (mestrado) - Universidade Federal do Rio Grande do Norte, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica e de Computação, Natal, RN, 2022.
Orientador: Prof. Dr. Wallace Moreira Bessa.

1. Controle não linear - Dissertação. 2. Controle inteligente - Dissertação. 3. Redes neurais recorrentes - Dissertação. 4. Linearização por realimentação - Dissertação. 5. Modos deslizantes - Dissertação. 6. Robôs móveis - Dissertação. I. Bessa, Wallace Moreira. II. Título.

RN/UF/BCZM CDU 681.3

Elaborado por Ana Cristina Cavalcanti Tinôco - CRB-15/262

Agradecimentos

Ao Prof. DSc. Wallace Moreira Bessa, por me introduzir a esta área que gosto tanto, pelos ensinamentos, pela orientação e pela amizade (líder RoboTeAM).

Aos demais membros do Grupo de Estudos em Robótica e Aprendizagem de Máquina (RoboTeAM) por toda ajuda e parceria nos últimos anos. Em especial aos amigos MSc. Eng. Gabriel da Silva Lima e MSc. Eng. Victor Ramon pelas dúvidas sanadas e pela ajuda no desenvolvimento deste trabalho. Aos demais, citarei, em ordem alfabética: BSc. Aissa Cavalcanti, Eng. BSc. Diago Xavier, Eng. BSc. Fernando Fernandes, Eng. BSc. Gabriel Baumann, DSc. João Deodato, Eng. BSc. Júlio Freire, Msc. Eng. João Lucas, Prof. DSc. Philippe Medeiros e Eng. Vitor Vale.

Aos companheiros de curso pela amizade e companheirismo, Eng. BSc. Ana Karoline Machado, Eng. BSc. Camila Barbosa, Eng. BSc. Gabriel Vantuil, Eng. BSc. Guillherme Amaral, Eng. BSc. Jonathan Martins, Eng. BSc. Rafael Dias, Eng. BSc. Samuel Amico e Eng. BSc. Victor Marcolino.

Ao *Clã* por todo apoio e constante motivação. Em ordem alfabética: MSc. Eng. Bruno Pinheiro, Eng. BSc. Glauber Carvalho, Prof. DSc. Hugo Melo, BSc. Pedro Rabello, BSc. Yllan Gurgel, BSc. Gabriel Felipe e Matheus Palhares.

Aos amigos que carrego desde a época de colégio pela amizade e companheirismo durante todo o curso. BSc. Isaac Pacheco, Iuri Montenegro, Matheus Pifrader, BSc. Victor Augusto e Yago Mavignier.

À minha namorada Louyse Ângelo, por todo o apoio, carinho e motivação, além da paciência.

A todos os meus familiares, especialmente meus pais, meus irmãos, Arthur e Vinicius, às minhas avós, Ivonete e Terezinha e ao tio Silvio pelas conversas motivadoras nos momentos de dificuldade e todo o apoio durante o curso.

A Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e ao Laboratório de Manufatura (LabMan-UFRN) por todo o suporte.

"O peso da evidência de uma afirmação extraordinária deve ser proporcional à sua estranheza."

— Pierre Simon Laplace

Resumo

Devido à sua grande eficiência, segurança e flexibilidade, os robôs móveis estão sendo cada vez mais utilizados na indústria. Entretanto, o controle de posicionamento dos mesmos se trata de um grande desafio dada a natureza não linear dessa planta e a dificuldade de estimar determinados parâmetros, como os efeitos do atrito. Além disso, um rastreamento de trajetória preciso pode ser essencial para determinadas operações com robôs móveis, como no caso de caminhos estreitos. Neste trabalho, controladores inteligentes são propostos para o rastreamento de trajetória de um robô móvel omnidirecional sujeito a dinâmicas não modeladas. As abordagens de controle utilizadas foram os controladores não lineares Linearização por Realimentação (FBL) e Modos Deslizantes (SMC), ambos acoplados de um compensador inteligente que utiliza Redes Neurais Recorrentes com o objetivo de lidar com as incertezas. A arquitetura da rede neural escolhida se baseou na necessidade de compensação de dinâmicas complexas e ao mesmo tempo restrição de complexidade computacional para que o mesmo pudesse ser embarcado no *hardware* de um robô móvel. As propriedades de estabilidade dos controladores foram provadas de acordo com o princípio de estabilidade assintótica segundo Lyapunov e o desempenho das estratégias. Foi avaliado tanto em simulações quanto em experimentos com o Robotino®, um robô móvel omnidirecional produzido pela Festo Didatics. Foi observado um ganho de desempenho no controlador quando comparado com redes neurais sem a recorrência.

Palavras-chave: Controle Não Linear, Controle Inteligente, Redes Neurais Recorrentes, Linearização por Realimentação, Modos Deslizantes, Robôs Móveis.

Abstract

Due to their great efficiency, security and flexibility, mobile robots are being increasingly used in industry. However, their positioning control is a great challenge due to the non-linear nature of this plant and the difficulty of estimating certain parameters, for example, the friction effects. Besides that, a precise tracking might be essential to some operations in mobile robots, such as narrow paths. In this work, non-linear controllers are applied to the trajectory control of an omnidirectional robot under the effect of unmodeled dynamics. The control approaches used in this work were both non-linear control strategies, Feedback Linearization (FBL) and Sliding Modes (SMC) both incorporated with an intelligent compensator utilizing Recurrent Neural Networks in order to assist the control by estimating uncertainties. The chosen architecture of the neural network was based in the need to compensate more complex dynamics and at the same time the restriction of computational complexity so that it could be embedded in the hardware of a mobile robot. The stability properties were proven by the principle of assintotic stability proposed by Lyapunov and the performance of the strategies were verifed through both simulations and experiments using Robotino®, an omnidirectional mobile robot produced by Festo Didatics and a performance gain was observed when compared with the neural network without the recurrence.

Keywords: Nonlinear Control, Intelligent Control, Recurrent Neural Networks, Feedback Linearization, Sliding Modes, Mobile Robots.

Lista de ilustrações

Lista de abreviaturas e siglas

ANN	*Artificial Neural Networks* - Redes Neurais Artificiais
API	*Application Programming Interface*
EDO	Equação Diferencial Ordinária
FBL	*Feedback Linearization* - Linearização por Realimentação
LSTM	*Long Short-Term Memory*
MIMO	*Multiple Input Multiple Output*
PID	*Proportional-Integrative-Derivative* - Controlador Proporcional Integral Derivativo
RBF	Redes de Função de Base Radial
RNN	Redes Neurais Recorrentes
ROS	*Robot Operating System*
SISO	*Single Input Single Output*
SMC	*Sliding Modes Controller* - Controlador por Modos Deslizantes
SMD	*Second-order Sliding Mode Differentiator*

Lista de símbolos

a_i	coeficientes de um polinômio de Hurwitz
$\boldsymbol{B}$	matriz de entrada
b	termo *bias* da rede neural
C	matriz dos pesos da camada de saída
$\boldsymbol{\mathcal{C}}$	matriz composta de matrizes $\boldsymbol{\Lambda}$ e coeficientes a_i
$\boldsymbol{d}$	vetor que comporta as incertezas do sistema
E	energia gasta
$\boldsymbol{e}$	vetor dos erros de rastreamento
F	função de aproximação
$\boldsymbol{f}$	vetor de forças
g	saída do somador da ANN
L	norma Lebesgue
M	matriz de inércia
q	variáveis de estado no referencial inercial
r	função de distância radial
s	medida do erro combinado
T_r	matriz de transformação
t	tempo
$\boldsymbol{u}$	vetor de entradas
V	função candidata a função de Lyapunov
$\boldsymbol{v}$	camada oculta da RNN no modelo de estados
w	pesos da rede neural
$\boldsymbol{x}$	vetor de estados do sistema

$\boldsymbol{x}_d$	vetor das trajetórias desejadas
y	saída do neurônio
z	sinais da entrada genérica da rede neural
α	fator de esquecimento da RNN
β	constante do filtro passa-baixas
$\Delta \boldsymbol{f}$	incerteza relativa ao vetor de forças
$\Delta \boldsymbol{B}$	incerteza relativa à matriz de Entrada
δ	termo de limitação da perturbação
ε	erro de estimativa
η	constante estritamente positiva
θ	ângulo de rotação para cada motor
$\boldsymbol{\kappa}$	vetor de ganhos do controlador SMC
$\boldsymbol{\Lambda}$	matriz diagonal com entradas positivas
λ	parâmetro de convergência do controlador
μ	limite superior da norma dos pesos
ν	taxa de aprendizagem da rede neural
Ξ	região fechada da camada limite do SMC
ρ	resposta desejada da interpolação
σ	desvio padrão da gaussiana
τ	torque de entrada nos motores
τ_f	vetor de atrito
$\boldsymbol{\phi}$	camada limite do controlador por Modos Deslizantes
φ	função de ativação
ψ	ângulo de rotação do robô
Ω	região convexa na qual os pesos da rede neural estão contidos
$\boldsymbol{c}$	centros das funções da RBF

Sumário

1 Introdução

Robôs móveis já estão presentes em diversas aplicações industriais, como por exemplo para transportar cargas e materiais, agricultura e vigilância (RUBIO; VALERO; LLOPIS-ALBERT, 2019). Eles são robôs autônomos que têm a capacidade de se movimentar e interagir com o ambiente em que estão inseridos (NILOY et al., 2021; TZAFESTAS, 2018). Seu papel na cadeia produtiva de algumas indústrias é muito grande dada a eficiência e segurança que ele proporciona. Além disso, eles permitem uma grande flexibilidade tecnológica, uma vez que a indústria não precisa realizar ajustes significativos no chão de fábrica para a sua implementação. Dentre os robôs móveis, os robôs omnidirecionais são uma classe particular que se destaca devido à sua capacidade de se movimentar em qualquer direção no plano, além de possuir uma grande manobrabilidade e poder realizar operações mais complexas, inclusive em caminhos estreitos (TERAKAWA et al., 2018; HAMAGUCHI, 2018; WANG et al., 2020a).

No entanto, apesar da sua utilização estar cada vez maior, o problema de projetar sistemas de posicionamento precisos para robôs móveis ainda desafia muitos engenheiros e pesquisadores interessados nessa área particular da engenharia. Para tal, várias abordagens de controle foram utilizadas, como o tradicional controlador PID (CARMONA et al., 2018; ARDIYANTO, 2010), *Fuzzy*-PID (SU et al., 2016; YANMIN; LIMIN; ZHIBIN, 2013) e PID com Redes Neurais, (WANG et al., 2020b; CUI; PAN; LI, 2013), porém devido às não linearidades relativas à planta, sua performance no rastreamento pode ser insatisfatória e até produzir instabilidades. Além disso, para problemas de rastreamento de trajetória, o PID pode perder desempenho devido à variação do ponto de operação. Para lidar com isso, a abordagem de controladores inerentemente não lineares é adequada para o problema.

Alguns trabalhos utilizaram controladores não lineares para o controle de robôs móveis, como o controlador por Linearização por Realimentação (LIU et al., 2013), o Controle Preditivo Não Linear (ESSEN; NIJMEIJER, 2001), controle por Linearização de Trajetória (LIU et al., 2008), Controle Ótimo (HUANG, 2013), controle por Modos Deslizantes (SOLEA; CERNEGA, 2015; KHANH et al., 2013; DINH et al., 2012; YANG; CHENG, 2013), entre outros. Entretanto, existe a dificuldade de se obter uma modelagem da dinâmica da planta (JEONG; CHWA, 2021), como por exemplo dos efeitos do atrito entre o robô e o chão, devido ao ambiente no qual os robôs móveis são empregados normalmente não ser estruturado. Neste contexto, inteligência computacional pode ser utilizada para estimar a dinâmica não modelada e as incertezas presentes no sistema.

Diversas abordagens utilizando inteligência computacional foram propostas, como compensadores utilizando a Lógica *Fuzzy* (TANAKA; FERNANDES; BESSA, 2013; SHIEV

et al., 2012; KRICHEN; MASMOUDI; DERBEL, 2021; CASTILLO et al., 2020), compensação por meio de aprendizagem de máquinas (*machine learning*), dentre elas, podemos citar as abordagens por aprendizagem por reforço (CADENGUE et al., 2019), aprendizagem supervisionada, como por Processo Gaussiano (LIMA; BESSA; TRIMPE, 2018), além das amplamente utilizadas Redes Neurais Artificiais (DOS SANTOS; BESSA, 2019; XU et al., 2009). Independente da habilidade da estratégia de inteligência computacional de compensar as incertezas, os aspectos relativos à implementação são muitas vezes ignorados, como a complexidade computacional. Devido a isso, o algoritmo inteligente a ser escolhido deve ser simples o suficiente para ser implementado no *hardware* embarcado dos robôs móveis. Além disso, é interessante que o aprendizado seja *online*, para que o robô continue aprendendo através da sua interação com o ambiente.

Dentre as técnicas de inteligência computacional, as Redes Neurais Artificiais são uma das mais utilizadas devido à sua facilidade de implementação e a sua natureza não linear, além de suas propriedades de aproximação universal bem definidas. Adicionalmente, as redes neurais possuem uma flexibilidade em relação à arquitetura que pode ser utilizada para determinados problemas. Como um exemplo da sua utilização, Moreira (2021) propôs um compensador por Redes Neurais, incorporado num controlador por Linearização por Realimentação, para melhorar o controle de um robô omnidirecional. Entretanto, dependendo do grau da incerteza, a compensação de dinâmicas mais complexas com uma boa precisão pode ser uma tarefa desafiante para estruturas simples como uma Rede Neural de uma camada (FEI; CHEN, 2020).

Uma saída para tal seria a utilização de redes neurais recorrentes. Bastante utilizadas em problemas que envolvem séries temporais, as redes neurais recorrentes são particularmente eficientes para este problema (KUNA, 2015; YAO; DATCU, 2018; LUDWIG, 2019), devido ao efeito de memória que a incorporação da recursão produz na rede. Fei e Lu (2018) utilizaram as redes neurais recorrentes como compensadores aplicados a controladores não lineares, entretanto, a arquitetura da rede utiliza uma dupla recursão e múltiplas entradas na rede. Além disso, foi aplicado para um sistema SISO (*Single Input Single Output*) por meio de simulações.

Como citado anteriormente, devido às não linearidades presentes na planta, neste trabalho, foram propostos dois controladores não lineares para realizar o rastreamento de trajetória para um robô móvel omnidirecional:

- a linearização por realimentação (FBL, do inglês *Feedback Linearization*), consistindo em uma técnica que lineariza a planta em malha fechada, permitindo transformar o sistema dinâmico original em um sistema dinâmico equivalente, porém mais simples, cuja principal desvantagem se trata da necessidade de conhecimento com precisão do modelo dinâmico e dos parâmetros do sistema;

- e o controlador por Modos Deslizantes (SMC, do inglês *Sliding Modes Controller*), que é conhecido por ser uma técnica de controle robusto, ou seja, capaz de lidar com incertezas na representação do modelo da planta, consistindo na formação de uma lei de controle que é determinada de forma que as trajetórias do sistema *deslizem* sobre uma região desejada no espaço de estados. Para evitar um chaveamento (*chattering*) no sinal de controle, o que pode causar vibrações, foi implementada uma suavização da lei de controle, utilizando a função saturação, neste caso o controle preciso de uma abordagem tradicional do SMC é substituído por um rastreamento de precisão garantida com base em uma camada limite.

E para a tarefa de lidar com as incertezas, foi proposto um compensador inteligente utilizando a abordagem de redes neurais recorrentes, o qual foi acoplado aos controladores citados anteriormente, para incorporar as incertezas na lei de controle, com objetivo de melhorar o rastreamento de trajetória. Para manter o custo computacional baixo, a arquitetura escolhida possui uma camada oculta e uma recursão simples. As propriedades de estabilidade foram provadas analiticamente. Ademais, o desempenho do controlador inteligente foi avaliado através de simulações e experimentos realizados com o robô omnidirecional Robotino® desenvolvido pela Festo Didactic. Além disso, os resultados da inclusão da recursão na rede foram comparados com as redes neurais tradicionais, mostrando a melhora no rastreamento de trajetória.

1.1 Organização do trabalho

Para facilitar a apresentação, o trabalho foi estruturado em seis capítulos.

O primeiro capítulo tem como propósito introduzir o leitor acerca dos veículos robóticos móveis e a importância do controle de posicionamento dinâmico dos mesmos, além de rapidamente introduzir as metodologias utilizadas.

O Capítulo 2 trata das estratégias de controle não linear bem consolidadas e que foram utilizadas neste trabalho, o Método de controle por Linearização por Realimentação e o Controlador por Modos Deslizantes.

No Capítulo 3 são apresentados os conceitos básicos sobre as Redes Neurais Artificiais e conceitos fundamentais para a utilização das mesmas como o Teorema da Aproximação Universal, tanto para Redes de Função de Base Radial quanto para as Redes Recorrentes.

Já no Capítulo 4, é feita uma pequena introdução sobre robôs móveis e a modelagem matemática do Robotino®, além disso, são demonstradas as propriedades de estabilidade dos controladores propostos utilizando o Teorema de Lyapunov.

A aplicação, por meio de simulações e experimentos realizados utilizando o Robotino®,

é apresentada no Capítulo 5, nele são mostradas as diferenças entre os casos com e sem a utilização da recursão na rede e é feita uma análise estatística sobre esses dados.

No Capítulo 6, são expostas as conclusões do trabalho e as respectivas sugestões para trabalhos futuros.

2 Estratégias de Controle para Sistemas Não Lineares

Pode-se definir controle como um processo por meio do qual consegue-se fazer com que um sistema comporte-se de maneira desejada. Sistemas de controle têm uma influência cada vez maior no nosso dia-a-dia, desde controladores mais simples, como o de aparelhos de ar-condicionado, até exemplos onde é necessária uma estratégia mais sofisticada, como por exemplo o controle para carros autônomos.

De acordo com Franklin, Powell e Emami-Naeini (2013), para alcançar um bom controle, existem quatro requisitos básicos:

- O sistema deve ser sempre estável.
- A saída do sistema deve rastrear o sinal de comando na entrada.
- A saída do sistema não deve responder a perturbações externas.
- Esses requisitos devem ser cumpridos, mesmo se o modelo utilizado no projeto não for totalmente preciso, se a dinâmica do sistema físico mudar ao longo do tempo ou devido a mudanças ambientais.

Portanto, para cumprir estes requisitos, a escolha da estratégia para realizar o controle do sistema desejado deve ser muito bem pensada. Embora controladores lineares tenham ampla utilização, controladores não lineares têm ganhado cada vez mais espaço, devido à dificuldade em muitos casos para modelar o sistema com grande precisão. Uma vez que quase em sua totalidade os sistemas físicos reais são não lineares, essa dificuldade de precisão pode acarretar, por exemplo, a instabilidades no sistema. Outro ponto a ser destacado, para problema de rastreamento de trajetória, o desempenho de controladores lineares pode ser prejudicado devido à mudança do ponto de operação, o que atrapalha o processo de linearização. Dentre as diversas estratégias de controle não linear, iremos abordar duas, Linearização por Realimentação e controle por Modos Deslizantes, que são amplamente utilizadas e que serão implementadas para o controle do sistema proposto.

2.1 Linearização por Realimentação

A primeira das estratégias de controle não linear utilizada neste trabalho se trata da Linearização por Realimentação (*Feedback Linearization* - FBL). Essa técnica consiste em transformar o sistema não linear num equivalente, porém linear em malha fechada.

Esta transformação, assim como o nome indica, é feita através do uso da realimentação. Ela difere totalmente dos métodos de linearização convencionais, uma vez que o sistema é representado em sua totalidade, não uma aproximação em torno de um ponto de operação. Podemos ilustrar com o desenvolvimento abaixo:

Partindo do princípio que um sistema dinâmico, não-autônomo, ou seja, invariante no tempo, MIMO (*Multiple Input-Multiple Output*) e não linear possa ser representado através de uma Equação Diferencial Ordinária (EDO), com n graus de liberdade e n entradas:

$$\boldsymbol{x}^{(n)} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{B}\boldsymbol{u} \tag{2.1}$$

em que $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ representa a matriz de entrada, o vetor $\boldsymbol{f} \in \mathbb{R}^n$ se trata do vetor de forças (em sistemas mecânicos, $\boldsymbol{f}$ geralmente incorpora os efeitos de Coriolis e centrífugos) $\boldsymbol{x}^{(n)}$ representa a n-ésima derivada em relação ao tempo da variável de estado $\boldsymbol{x}$, $\boldsymbol{u}$ o vetor de entradas e $\boldsymbol{x} = [x, \dot{x}, ..., x^{n-1}]^\top$. Além disso, a dependência explícita com o tempo foi removida para simplificar a notação.

Nesse trabalho, serão adotadas as seguintes considerações:

Consideração 2.1. Os estados $\boldsymbol{x}$ são mensuráveis.

Consideração 2.2. A trajetória desejada $\boldsymbol{x}_d$ possui, ao menos, uma derivada com respeito ao tempo. Além disso, cada componente é conhecida e limitada.

Agora considere que a variável de entrada seja projetada para um problema de rastreamento de trajetória, ou seja, fazer com que $\boldsymbol{x} \to \boldsymbol{x}_d$ a medida que $t \to \infty$, onde $\boldsymbol{x}_d = [x_d, \dot{x_d}, ..., x_d^{(n-1)}]^\top$ é a trajetória desejada. Isso significa fazer com que $\boldsymbol{e} \to 0$, sendo $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{x}_d = [e, \dot{e}, ..., e^{n-1}]^\top$ é o erro de rastreamento. Para atingir esse objetivo, modelemos a lei de controle da seguinte forma,

$$\boldsymbol{u} = \boldsymbol{B}^{-1}(-\boldsymbol{f} + \boldsymbol{x}_d^n - a_0\boldsymbol{\Lambda}\boldsymbol{e} - a_1\boldsymbol{\Lambda}\dot{\boldsymbol{e}} - ... - a_{n-1}\boldsymbol{\Lambda}\boldsymbol{e}^{(n-1)}) \tag{2.2}$$

sendo $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ uma matriz diagonal com entradas positivas λ_i. Cancelando todos os termos não lineares através da realimentação, a resposta do sistema dinâmico em malha fechada será linear e terá uma convergência exponencial em zero caso os coeficientes a_i sejam de um polinômio de Hurwitz. Os coeficientes podem ser facilmente calculados através do Binômio de Newton, utilizando a equação abaixo:

$$a_i = \binom{n}{i} \lambda^{n-i-1} \tag{2.3}$$

onde $i = 0, 1, ..., n-1$ e $\lambda \in \mathbb{R}^+$.

A técnica de linearização por realimentação é bastante utilizada na área de robótica industrial, principalmente devido ao projetista do controle saber com grande precisão a modelagem da planta desejada, porém nem sempre isso é possível, principalmente em

ambientes não estruturados. Sendo a presença de incertezas relativas ao modelo e aos parâmetros do sistema o maior problema dessa técnica.

Devido a isso, quando o sistema utilizado é incerto, é necessária a utilização de estratégias robustas ou a incorporação de compensadores para melhorar o desempenho do controlador. Dito isto, apresentaremos o controlador por Modos Deslizantes, conhecido por ser uma técnica de controle robusto utilizado para lidar com tais incertezas. Ademais, será proposto um compensador para estimar tais incertezas, que será apresentado nas próximas seções.

2.2 Controle por Modos Deslizantes

Como abordado anteriormente, controlar um sistema não linear com modelos imprecisos pode ser problemático. Essa imprecisão pode vir de diversas fontes, como incertezas relativas à planta ou representações simplificadas do modelo (linearização de parâmetros, por exemplo).

Uma das abordagens para lidar com essas incertezas é com o uso de controladores robustos, que adicionam termos a lei de controle com esse propósito. Uma abordagem simples de controle robusto é chamada de Controle por Modos Deslizantes (*Sliding Modes Controller* - SMC), nela, um sistema de ordem n é transformado num sistema equivalente de primeira ordem em s. Para isso, considere um superfície $S(t)$, dita de deslizamento, definida no espaço de estado $\mathbb{R}^n$ pela equação $s(x,t) = 0$, sendo $s : \mathbb{R}^n \to \mathbb{R}$ definida para um sistema dinâmico de ordem n, pela equação a seguir:

$$s(\boldsymbol{x}, t) = \left(\frac{d}{dt} + \lambda\right)^{n-1} e \tag{2.4}$$

com $\lambda \in \mathbb{R}^+$.

Desta forma, a lei de controle u deve ser projetada de modo a garantir que $\boldsymbol{x}$ alcance a superfície $s(x,t) = 0$ em um dado intervalo de tempo finito, e após isso, como o nome indica, deslize sobre ela de forma exponencial até atingir x_d. Podemos, então, separar o problema em duas etapas: o modo de aproximação, onde o sistema tende até a superfície de deslizamento e o modo deslizante, no qual a trajetória dos estados está restrita a esta superfície.

Através da equação 2.4, para um sistema de um 2ª ordem ($n = 2$), pode-se obter uma expressão para a superfície de deslizamento igual a $s(\boldsymbol{x}, t) = \dot{e} + \lambda e = 0$. Como representado pela figura 2.1 (LIMA, 2019), podemos ver que o estado inicial e_0 se desloca, durante o modo de aproximação, até a camada limite e, após isso, durante o modo deslizante, o mesmo desliza até o estado desejado dentro da superfície de deslizamento.

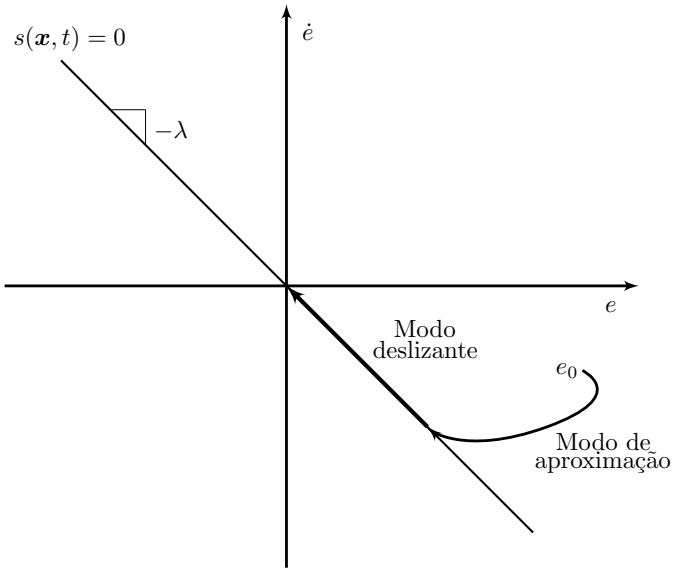

Figura 2.1 – Descrição das etapas do controlador para um sistema de segunda ordem

Para projetar o controlador de modo que ambos os modos de operação sejam satisfeitos, consideremos um problema onde a condição inicial não é igual ao estado desejado, além disso, como discutido anteriormente, iremos atribuir incertezas, aqui denotadas Δ, em ambos os termos da Eq. 2.1, $\boldsymbol{f} = \hat{\boldsymbol{f}} + \Delta\boldsymbol{f}$ e $\boldsymbol{B} = \hat{\boldsymbol{B}} + \Delta\boldsymbol{B}$. Portanto, o sistema anteriormente representado pela Eq. 2.1 passa a ser representado da seguinte forma:

$$\boldsymbol{x}^{(n)} = \hat{\boldsymbol{f}}(\boldsymbol{x}) + \hat{\boldsymbol{B}}\boldsymbol{u} + \boldsymbol{d} \tag{2.5}$$

no qual $\boldsymbol{d} = \Delta\boldsymbol{f} + \Delta\boldsymbol{B}\boldsymbol{u}$ é o termo que comporta as incertezas do sistema. Iremos chamá-lo de perturbação. Além disso, faremos a consideração de que esta perturbação é desconhecida, porém limitada $|\boldsymbol{d}| \leq \boldsymbol{\delta}$.

Proponhamos uma superfície de deslizamento para um sistema de segunda ordem ($n = 2$) MIMO tal qual:

$$\boldsymbol{s} = \dot{\boldsymbol{e}} + \boldsymbol{\Lambda}\boldsymbol{e} \tag{2.6}$$

sendo $\boldsymbol{\Lambda}$ é uma matriz diagonal com entradas positivas λ_i.

Além disso, consideremos a seguinte lei de controle baseada em (SLOTINE; LI, 1991):

$$\boldsymbol{u} = \hat{\boldsymbol{B}}^{-1}(-\hat{\boldsymbol{f}} + \ddot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{\Lambda}\dot{\boldsymbol{e}} - \boldsymbol{\kappa}\mathrm{sgn}(\boldsymbol{s})) \tag{2.7}$$

onde $\boldsymbol{\kappa}\mathrm{sgn}(\boldsymbol{s}) = [\kappa_1\mathrm{sgn}(s_1), \kappa_2\mathrm{sgn}(s_2), ..., \kappa_n\mathrm{sgn}(s_i)]$, é um termo capaz de manter os estados do sistema na superfície de deslizamento. A função $\mathrm{sgn}(s_i)$ é chamada função sinal, nós

podemos a definir como:

$$\text{sgn}(x) = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases} \tag{2.8}$$

O vetor de ganhos $\boldsymbol{\kappa}$ deve ser obtido de forma a garantir a estabilidade do sistema. Para isso, iremos introduzir o Teorema de estabilidade assintótica de Lyapunov, cuja prova pode ser encontrada em Slotine e Li (1991).

Teorema 2.1. Dada uma função V do sistema, dita *função de Lyapunov*, esse sistema será assintoticamente estável se V for uma função definida positiva e $\dot{V}$ for uma função definida negativa.

Definição 2.1. Uma função $V(x)$ é dita definida positiva se, para todo $x \neq 0$, $V(x) > 0$ e $V(0) = 0$.

Definição 2.2. Uma função $V(x)$ é dita definida negativa se $-V(x)$ for definida positiva.

Consideremos a seguinte função candidata a função de Lyapunov relativa a cada grau de liberdade i,

$$V_i(t) = \frac{1}{2} s_i^2 \tag{2.9}$$

Podemos ver que para $s_i = 0$, $V_i(s_i) = 0$, além de que para $s_i \neq 0$, $V_i(s_i) > 0$. Portanto, a função candidata é definida positiva. Agora analisemos $\dot{V}_i(s_i)$.

Diferenciando a Eq. 2.9:

$$\dot{V}_i(t) = s_i \dot{s}_i \tag{2.10}$$

Substituindo a derivada da Eq. 2.6 na Eq. 2.10, temos:

$$\dot{V}_i(t) = s_i(\ddot{e}_i + \lambda_i \dot{e}_i) = s_i(\ddot{x}_i + \ddot{x}_{i_d} + \lambda_i \dot{e}_i) \tag{2.11}$$

Aplicando a Eq. 2.5 e a Eq. 2.7 na Eq. 2.11 e simplificando os termos, teremos:

$$\dot{V}_i(t) = s_i(d_i - \kappa_i \ \text{sgn}(s_i)) \tag{2.12}$$

Considerando que $|d_i| \leq \delta_i$, podemos chegar em:

$$\dot{V}_i(t) \leq s_i(\delta_i - \kappa_i \ \text{sgn}(s_i)) \tag{2.13}$$

Temos por $\kappa_i \geq \delta_i + \eta_i$, chegamos a:

$$\dot{V}_i(t) \leq -\eta_i |s_i| \tag{2.14}$$

sendo η uma constante estritamente positiva.

Portanto, a derivada de V com respeito ao tempo é definida negativa e, segundo Lyapunov, o sistema é assintoticamente estável utilizando a lei de controle proposta. A Eq. 2.14 é chamada *condição de deslizamento*. Com ela, podemos inferir que o quadrado da distância até a superfície $S(t)$ apenas decresce ao longo da trajetória. Portanto, uma vez na superfície, a trajetória do sistema permanece na superfície (SLOTINE; LI, 1991).

Além disso, como pode ser demonstrado por Bessa (2009), essa convergência de $\boldsymbol{e} \to 0$ ocorre dentro de um *tempo finito*, t_{alc}, onde $s(t \geq t_{alc}) = 0$, chega-se a seguinte expressão:

$$t_{alc} \geq \frac{|s_i(0)|}{\eta_i} \tag{2.15}$$

no qual o termo η garante que os estados do sistema alcancem a superfície de deslizamento dentro de um tempo finito.

Apesar disso, a lei de controle utilizando a função $\text{sgn}(s)$ pode causar um efeito de chaveamento bastante intenso na entrada, o *chattering*, tal efeito pode provocar um comportamento indesejado no sistema. Por exemplo em sistemas mecânicos, onde podem ser geradas vibrações.

Para contornar este problema, Slotine e Li (1991) propuseram uma adaptação na lei de controle com o objetivo de suaviza-la. Nesta abordagem, há a adoção de uma camada limite nas vizinhanças da superfície de deslizamento. Para isso, pode-se, por exemplo, substituir a função $\text{sgn}(s)$ da lei de controle por uma função de saturação, que pode ser definida da seguinte forma:

$$\text{sat}(x) = \begin{cases} \text{sgn}(x), & |x| \geq 1 \\ x, & |x| < 1 \end{cases} \tag{2.16}$$

Portanto, com a utilização da função saturação, temos a lei de controle da seguinte forma:

$$\boldsymbol{u} = \hat{\boldsymbol{B}}^{-1}(-\hat{\boldsymbol{f}} + \ddot{\boldsymbol{x}}_{\boldsymbol{d}} - \boldsymbol{\Lambda}\dot{\boldsymbol{e}} - \boldsymbol{\kappa}\text{sat}(\boldsymbol{\phi}^{-1}\boldsymbol{s})) \tag{2.17}$$

na qual $\boldsymbol{\phi} = [\phi_1, \phi_2, ..., \phi_i] \in \mathbb{R}^+$ e corresponde ao vetor que denota à espessura da camada limite. A escolha do valor dessas constantes se dá com base em quão preciso o rastreamento deve ser, caso esse valor seja muito alto, o rastreamento não será tão preciso, porém, caso seja muito pequeno, ocorrerá *chattering*. Podemos observar na Fig.2.2 o espaço de fase com o uso da função saturação, além da sua representação gráfica.

Assim, vemos que com a adição da função saturação, o controle de rastreamento preciso da abordagem tradicional do SMC se torna um controle suavizado com precisão garantida através da camada limite. Ou seja, há um *tradeoff* entre manter o sistema suave com o aumento da camada limite e aumentar a precisão com a diminuição dela.

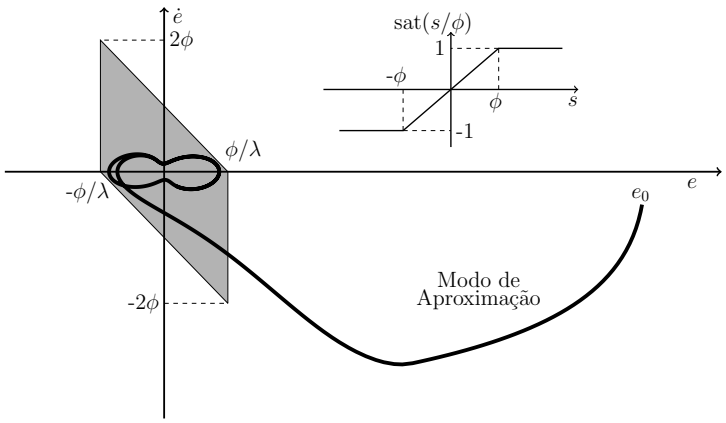

Figura 2.2 – Espaço de fase para um sistema de segunda ordem com a função saturação.

Com a modificação da lei de controle, se faz necessária, mais uma vez, a análise da estabilidade assintótica segundo Lyapunov. Utilizaremos a função candidata a Lyapunov $V_{\phi_i}(t) = \frac{1}{2}s_{\phi_i}^2$ definida positiva, vamos analisar $\dot{V}_{\phi_i}(t)$ onde $s_{\phi_i} = s_i - \phi_i \text{sat}(s/\phi_i)$ denota a distância do estado atual até a camada limite. Como para $s_i > \phi_i$ a função saturação é igual a função sinal, nós temos que para este caso, pode-se utilizar a mesma análise de $\dot{V}_i(t)$ para $\dot{V}_{\phi_i}(t)$, ou seja:

$$\dot{V}_{\phi_i} \leq -\eta_i |s_{\phi_i}| \tag{2.18}$$

Desta forma, pode-se inferir que o sistema converge a camada limite $\boldsymbol{\phi}$. Uma outra análise que pode ser demonstrada para esta convergência é utilizando o Lema de Barbalat para complementar a prova de estabilidade assintótica de Lyapunov.

Lema 2.1. Se uma função diferenciável f possui um limite finito ao passo que $t \to \infty$, e se $\dot{f}$ é uniformemente contínua, então $\dot{f}(t) \to 0$ quando $t \to \infty$.

Definição 2.3. Uma função é dita uniformemente contínua, se para todo t, sua derivada é limitada.

Para analisar a estabilidade, podemos fazer uma análise do Lema de Barbalat segundo a perspectiva de Lyapunov.

Lema 2.2. [Análise do tipo Lyapunov]: Se uma função escalar $V(x,t)$ satisfaz as seguintes condições:

- $V(x,t)$ é limitada inferiormente
- $\dot{V}(x,t)$ é negativa semi-definida
- $\dot{V}(x,t)$ é uniformemente contínua no tempo

então $\dot{V}(x,t) \to 0$ para $t \to \infty$.

Portanto, de acordo com a Consideração 2.2, é possível afirmar que $\dot{V}_\phi$ é uniformemente contínua, então, além disso, a partir da Eq. 2.18, podemos observar que

$V_\phi(t) \leq V_\phi(0)$ e, consequentemente, s_ϕ é limitado inferiormente. Segundo o Lema 2.2, para $t \to \infty$, $\dot{V}_\phi \to 0$. Dessa maneira, para $t \to \infty$, $s_\phi \to 0$, o que garante a convergência assintótica para a região da camada limite.

Uma vez dentro da camada limite, Bessa (2009) demonstra que o erro de rastreamento $\boldsymbol{e}$ sempre se mantém contido dentro de uma região delimitada pelo seguinte teorema:

Teorema 2.2. Seja a camada limite de um sistema de ordem n definido na forma $S_\phi = \{\boldsymbol{e} \in \mathbb{R}^n \mid |s(\boldsymbol{e}, t)| \leq \phi\}$. Qualquer trajetória iniciada dentro da camada limite irá convergir para uma região fechada $\Xi = \left\{ \boldsymbol{e} \in \mathbb{R}^n \mid |e^{(i)}| \leq \dfrac{(i+1)!}{\lambda^{n-i-1}}\phi, i = 0, ..., n-1 \right\}$.

Porém, apesar de diminuir o *chattering* com a introdução da função saturação, um pequeno erro de rastreamento é introduzido. Assim, a incorporação de estratégias de inteligência computacional para compensar as incertezas relacionadas à dinâmica do sistema pode auxiliar a melhorar o desempenho do rastreamento de trajetória.

3 Redes Neurais

Propostas por Mcculloch e Pitts (1943), as Redes Neurais Artificiais (*Artificial Neural Networks* - ANNs), simulam o funcionamento de um neurônio biológico e possuem diversas aplicações nos mais diversos campos, como para detecção de anomalias em indústrias (SIEGEL, 2020), criptografia (DONG; HUANG, 2020), reconhecimento de imagens (TIAN, 2020), sistemas de controle (BESSA et al., 2018; FEI; LU, 2018) dentre outras.

No ramo da predição, as redes neurais tem um apelo ainda maior devido à sua propriedade de serem não-lineares, assim como grande parte dos sinais do mundo real. Além disso, Gershenfeld e Weigend (1993) mostram que modelos lineares não conseguem capturar padrões mais complexos, como ciclos limites, bifurcações e pontos fixos.

De acordo com Haykin (2001) "Uma rede neural é um processador maciçamente e paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso". Esse conhecimento é adquirido através de um processo de aprendizagem que consiste em modificar os pesos sinápticos da rede de forma a atingir um determinado propósito.

A arquitetura tradicional de uma rede neural convencional proposta por White e Rosenblatt (1963) e conhecida como o Perceptron de Rosenblatt é dada por três elementos: os pesos ou sinapses, um somador e uma função de ativação. Os pesos expressam a importância das respectivas entradas às saídas, o somador serve como combinador linear dos pesos e a função de ativação restringe a amplitude da saída do neurônio. Isso é bastante útil para que a rede possa executar ações mais complexas.

Além disso, pode ser adicionado mais um elemento conhecido como *bias*, com o objetivo de transladar a entrada da função de ativação. Podemos representar um neurônio com as seguintes equações:

$$g_k = \sum_{j=1}^{m} w_{kj} z_j \tag{3.1}$$

e

$$y_k = \varphi(g_k + b_k) \tag{3.2}$$

onde os sinais da entrada são representados por $z_1, z_2, ..., z_m$, os pesos do neurônio são $w_{k1}, w_{k2}, ..., k_{km}$, g_k se trata da saída do somador, b_k é o bias, φ é a função de ativação e y_k é a saída do neurônio. Uma representação desse modelo pode ser vista na Fig. 3.1.

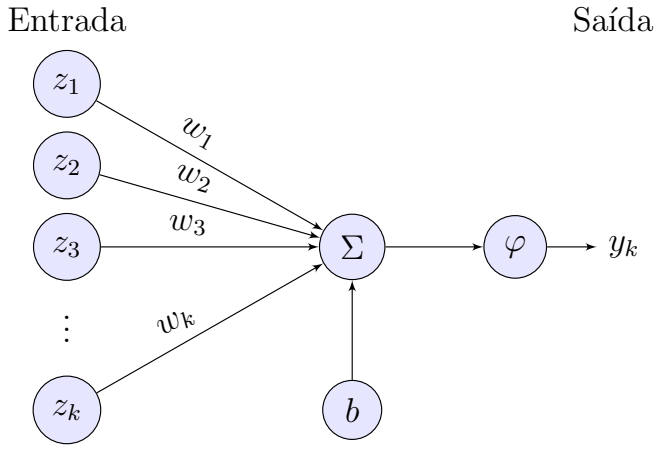

Figura 3.1 – Modelo do neurônio.

3.1 Redes de Função de Base Radial

As Redes de Função de Base Radial (RBF) são mais uma arquitetura existente para as redes neurais artificiais, apresentadas por Powell (1985). Nelas, vemos o projeto de uma rede neural como um problema de ajuste de curvas, onde aprender se trata de encontrar uma superfície em um espaço multidimensional que possa realizar um bom ajuste de acordo com os dados de treinamento.

Podemos definir sua construção da seguinte forma: Uma camada de entrada, uma segunda camada oculta constituída por funções que irão aplicar uma transformação nos dados de entrada e uma terceira camada que terá a saída da rede. Este esquema pode ser representado pela Fig. 3.2.

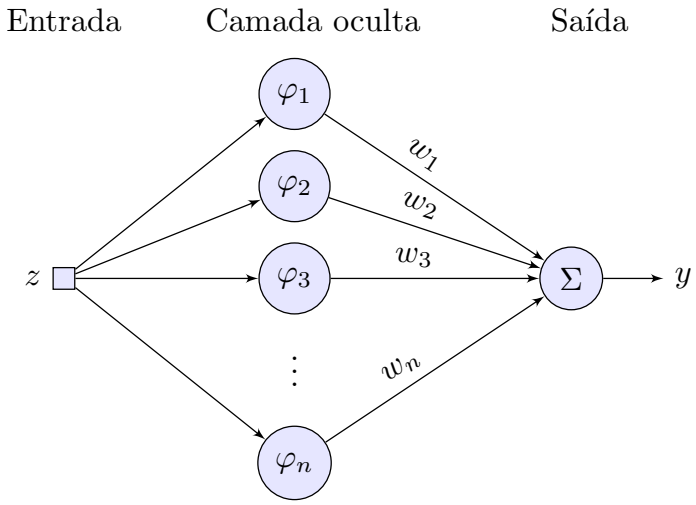

Figura 3.2 – Modelo de uma rede RBF.

De acordo com Broomhead (1988), a generalização em redes neurais pode ser associada a um simples processo de interpolação entre dados conhecidos. Assim, utilizando

a técnica de funções de base radial (RBF) proposta por Powell (1988), escolheremos uma função F com a seguinte forma :

$$F(\boldsymbol{z}) = \sum_{i=1}^{M} w_i \varphi(||\boldsymbol{z} - \boldsymbol{z}_i||) \tag{3.3}$$

onde $\{\varphi(||\boldsymbol{z} - \boldsymbol{z}_i||)|i = 1, 2, ..., N\}$ é um conjunto de N funções (geralmente não-lineares) arbitrárias, conhecidas como *funções de base radial*. Os pontos de dados conhecidos $\boldsymbol{x_i} \in \mathbb{R}^{m_0}, i = 1, 2, ..., N$ são tomados como os centros das funções de base radial.

Sendo o problema de interpolação formulado como (DAVIS, 1963):

$$F(\boldsymbol{z}) = \rho_i, i = 1, 2, ..., N \tag{3.4}$$

onde devemos encontrar uma função $F : \mathbb{R}^N \to \mathbb{R}^I$ que satisfaça a equação 3.4 dado um conjunto de N pontos diferentes $\boldsymbol{z}_i$ e um conjunto correspondente de N números reais ρ_i.

Assim, obtemos o conjunto de equações lineares para pesos desconhecidos inserindo a equação 3.4 em 3.3:

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1N} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{N1} & \varphi_{N2} & \cdots & \varphi_{NN} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix} = \begin{bmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_N \end{bmatrix} \tag{3.5}$$

onde $\boldsymbol{\rho} = [\rho_1, \rho_2, ..., \rho_N]^T$ representa o vetor com a resposta desejada e $\boldsymbol{w} = [w_1, w_2, ..., w_N]^T$ se trata do vetor de pesos. N é o tamanho da amostra de treinamento. Consideremos uma matriz de interpolação $\boldsymbol{\Phi}_{N \times N}$ com elementos φ_{ji}, então podemos reescrever 3.5 como:

$$\boldsymbol{\Phi w} = \boldsymbol{z} \tag{3.6}$$

Assim, chegamos a equação para o vetor de pesos $\boldsymbol{w}$:

$$\boldsymbol{w} = \boldsymbol{\Phi^{-1} z} \tag{3.7}$$

Entretanto, para essa equação ter solução, a inversa da matriz $\boldsymbol{\Phi}$ precisa existir, ou seja, $\boldsymbol{\Phi^{-1}}$ precisa ser não-singular. Para tal, o teorema de Michelli é bastante útil.

De acordo com Micchelli (1986), considerando que $\boldsymbol{z}_{i=1}^N$ seja um conjunto de pontos distintos em $\mathbb{R}^{m_0}$, então a matriz de interpolação $\boldsymbol{\Phi}_{NxN}$, é não singular.

Desta forma, podemos utilizar qualquer função desde que ela seja coberta pelo teorema de Michelli, o que engloba uma grande gama de funções. Dentre elas, a seguinte função é bastante utilizada no estudo de redes RBF:

- As funções gaussianas:

$$\varphi(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{3.8}$$

  para um $\sigma > 0$ e $r \in \mathbb{R}$

onde σ representa o desvio padrão da gaussiana e $r = ||z - z_i||$, a função de distância radial. Para que a função de base radial dada pela Eq. 3.8 seja não-singular, basta que todos os seus pontos sejam distintos. Outro ponto a se observar é que a função gaussiana é uma função *localizada*, isto é, $\varphi(r) \to 0$ quando $r \to \infty$.

3.1.1 Teorema da Aproximação Universal

Uma propriedade muito útil das redes neurais de base radial é a sua propriedade aproximativa. Park e Sandberg (1991) propuseram o *teorema da aproximação universal* para as redes de Função de Base Radial, que pode ser apresentado da seguinte forma: Dada uma função qualquer contínua $f(\boldsymbol{z})$, existe uma rede RBF com um conjunto de centros $\boldsymbol{c}_{i=1}^{m_1}$ e uma largura σ tal que a função $F(\boldsymbol{z})$ obtida pela RBF aproxime $f(\boldsymbol{z})$ na norma L_p, onde $p \in [1, \infty]$.

Em outras palavras, qualquer função pode ser aproximada por uma rede RBF, dado um número suficiente de neurônios. Este teorema é de grande importância para a utilização de redes RBF em problemas práticos, inclusive sendo utilizado em provas de estabilidade de controladores na área de sistemas de controle.

3.2 Redes Neurais Recorrentes

Outra arquitetura de redes neurais bastante utilizada se trata das Redes Neurais Recorrentes (RNNs). Elas diferem das redes neurais tradicionais pela inclusão de entradas passadas na rede com o uso da recursão. Esta recursão funciona criando uma espécie de memória no sistema, sem essa memória, as redes *feedforward* podem não ser capazes de capturar alguns comportamentos dinâmicos mais complexos de sistemas não lineares (MANDIC; CHAMBERS, 2001).

Podemos representar o comportamento dinâmico da rede recorrente através do modelo de espaço de estados. Nessa representação, os estados vão ser representados pelos neurônios ocultos. Após a camada oculta, ocorre uma recursão para realimentar a camada de entrada através de um banco de atrasos unitários. O número de atrasos unitários utilizados para a realimentação irá definir a ordem do modelo. Considerando $\boldsymbol{u}(n)$ como o vetor de entrada e $\boldsymbol{x}(n)$ como o vetor de saída da camada oculta no tempo n, podemos

construir as equações a seguir:

$$\boldsymbol{x}(n+1) = \boldsymbol{v}(\boldsymbol{x}(n), \boldsymbol{u}(n)) \tag{3.9}$$

$$\boldsymbol{y}(n) = \boldsymbol{C}\boldsymbol{x}(n) \tag{3.10}$$

onde $\boldsymbol{v}$ irá representar a função responsável por caracterizar a camada oculta, $\boldsymbol{C}$ é a matriz dos pesos da camada de saída.

A partir dessa definição, uma propriedade muito importante das redes neurais *feedforward* foi estendida para as redes recorrentes como demonstrado em Schäfer e Zimmermann (2006). O teorema foi descrito se utilizando da representação em modelo de estados representado pelas equações 3.9 e 3.10, sendo ele:

Teorema 3.2. Seja $\boldsymbol{v}(\cdot) : \mathbb{R}^J \times \mathbb{R}^I \to \mathbb{R}^J$ mensurável e $\mathbf{x}(\cdot) : \mathbb{R}^J \to \mathbb{R}^N$ seja contínuo. Então, qualquer sistema dinâmico na forma

$$\boldsymbol{x}(n+1) = \boldsymbol{v}(\boldsymbol{x}(n), \boldsymbol{u}(n))$$

$$\boldsymbol{y}(n) = \boldsymbol{C}\boldsymbol{x}(n)$$

pode ser aproximado por um elemento da classe de funções RNN(φ) com uma acurácia arbitrária, onde φ representa a função de ativação. Portanto, apesar da incorporação da recursão na rede, pode-se manter as análises relativas às propriedades de aproximação das redes neurais *feedforward*.

3.2.1 Arquiteturas das Redes Neurais Recorrentes

Podemos definir diversas arquiteturas para redes RNN, cada uma apresenta suas vantagens e desvantagens, e são utilizadas em aplicações bastante diversificadas. A estratégia pioneira foi apresentada por Hopfield (1982), chamado modelo de Hopfield. Nele, forma-se um sistema realimentado de múltiplos laços e o número de laços de realimentação é igual ao número de neurônios. Esses laços de realimentação se ligam a cada um dos outros neurônios da rede, exceto nele próprio. Uma representação dessa arquitetura pode ser observada na Fig. 3.3.

Outra arquitetura que vem se destacando atualmente devido ao aumento de capacidade de processamento se trata da *Long Short-Term Memory* (LSTM), com diversas aplicações (LUDWIG, 2019; YAO; DATCU, 2018; PANAPONGPAKORN; BANJERDPONGCHAI, 2019; LEI; ZHANG, 2019), principalmente no campo do Aprendizado Profundo, uma vez que se trata de uma arquitetura que devido a quantidade de blocos de operações, requer uma grande capacidade tanto de processamento quanto de memória. Como destacado por Rezk et al. (2020), devido a essa necessidade de uma grande capacidade

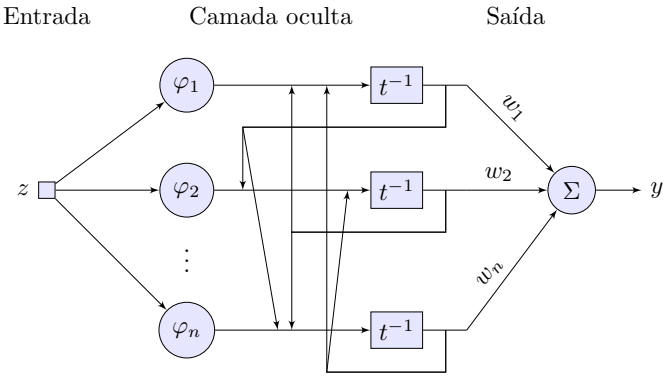

Figura 3.3 – Arquitetura do modelo de Hopfield.

computacional, dificuldades na implementação das RNNs podem surgir, principalmente no contexto de sistemas embarcados.

Desta forma, uma alternativa de arquitetura simples se trata da descrita em Elman (1990), conhecida por *rede recorrente simples*, possui unidades de contexto que armazenam as saídas dos neurônios ocultos dado um passo temporal e após isso as realimenta na camada de entrada. Uma representação desta arquitetura pode ser encontrada na Fig. 3.4.

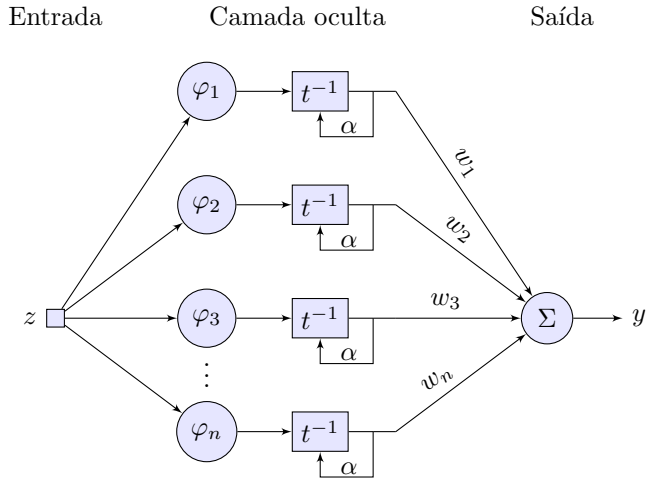

Figura 3.4 – Esquemático da arquitetura da rede recorrente simples

Desta forma, para o caso da arquitetura da rede recorrente simples, podemos reescrever a Eq. 3.3 da seguinte forma:

$$F(\boldsymbol{z}) = \sum_{i=1}^{M} w_i \boldsymbol{h}(||\boldsymbol{z} - \boldsymbol{z}_i||) \tag{3.12}$$

onde,

$$\boldsymbol{h}(||\boldsymbol{z} - \boldsymbol{z}_i||) = \boldsymbol{\varphi}(||\boldsymbol{z} - \boldsymbol{z}_i||) + \alpha \boldsymbol{h}((||\boldsymbol{z} - \boldsymbol{z}_i||), t - \Delta t) \tag{3.13}$$

no qual o parâmetro α se trata do fator de esquecimento, onde a variação dele afeta diretamente o peso do valor anterior na rede. Quanto maior o valor de α, maior a importância dada a valores anteriores.

Assim, a rede tem conhecimento das funções de ativações anteriores, facilitando a realização de tarefas que variam no tempo, como predição de séries temporais. Nesta arquitetura, diferentemente do modelo de Hopfield, a realimentação se dá apenas no próprio nó, o que diminui o custo computacional devido a menor quantidade de realimentações.

4 Posicionamento Dinâmico de Veículos Robóticos

O termo robô, popularizado pelo dramaturgo tcheco Karel Čapek, com sua peça intitulada *Robôs universais de Rossum*, deriva das palavras tchecas *rabota* e *robotnik*, que podem ser traduzidas, respectivamente como "trabalho forçado"e "servo"(MATARIC et al., 2014). Diferente do que essas palavras propõem, os robôs atualmente tem atuações bem mais amplas do que apenas trabalhos forçados.

Os robôs já estão bastante presentes no nosso dia-a-dia, desde robôs que ajudam nas tarefas domésticas, como os robôs aspiradores de pó, até robôs industriais que são responsáveis por grande parte da manufatura de diversos produtos.

Um dos trabalhos mais importantes para o desenvolvimento da robótica, se trata do livro Cibernética, escrito por Wiener (1948), nele foi proposto o estudo de sistemas biológicos desde o nível neuronal, tais princípios foram aplicados em robôs simples, utilizando as abordagens da teoria de controle. Diversos trabalhos posteriores partiram deste princípio de bioinspiração para robôs, entre eles o trabalho de Braitenberg (1986), *Vehicles*. Nele, diversos robôs simples são propostos com o objetivo de produzirem comportamentos que simulam animais, alguns deles poderiam armazenar informações e até construir formas de aprendizado.

Os robôs podem ter diversas características, dependendo do objetivo para o qual ele foi construído e do ambiente no qual eles atuarão. A Fig. 4.1 mostra uma classificação dos robôs em relação à sua estrutura e ambiente que atuará.

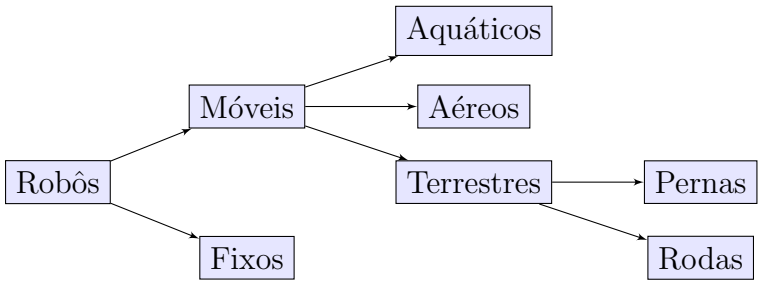

Figura 4.1 – Classificação dos Robôs

Com relação aos robôs móveis terrestres e com rodas, existem diversos tipos de configurações, Shvalb e Hacohen (2019) os categoriza de acordo com a Fig. 4.2.

Cada uma dessas configurações possui suas vantagens, por exemplo, os veículos de duas rodas são bastante úteis em determinadas aplicações devido a serem bastante

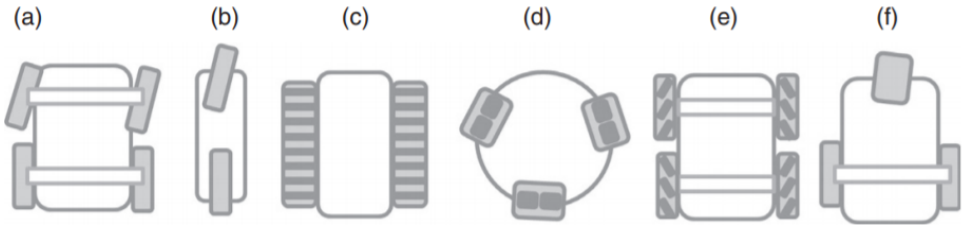

Figura 4.2 – Tipos de veículos com rodas. a) Quadriciclo; b) Duas rodas; c) Esteiras; d) Omni-rodas; e) Rodas *Mecanum*; f) Diferencial. (KAGAN et al., 2019)

estreitos e terem uma boa manobrabilidade, entretanto eles requerem um processo contínuo de controle para sua estabilização, o que aumenta o gasto energético.

Veículos com esteiras podem andar mesmo em terrenos desnivelados, o que os torna bastante úteis para aplicações *off-road*. Por outro lado, eles possuem uma menor eficiência para curvas comparado a outros tipos de mecanismos.

Uma configuração que permite andar em todas as direções planares e com ótima manobrabilidade se trata dos veículos omnidirecionais. As omni-rodas são rodas com rolamentos perpendiculares ao eixo de rotação da roda. Duas rodas são posicionadas de forma não paralela, como mostrado na Fig. 4.3 (BLIESENER et al., 2013). Isso permite se movimentar para qualquer direção apenas as rotacionando com diferentes velocidades.

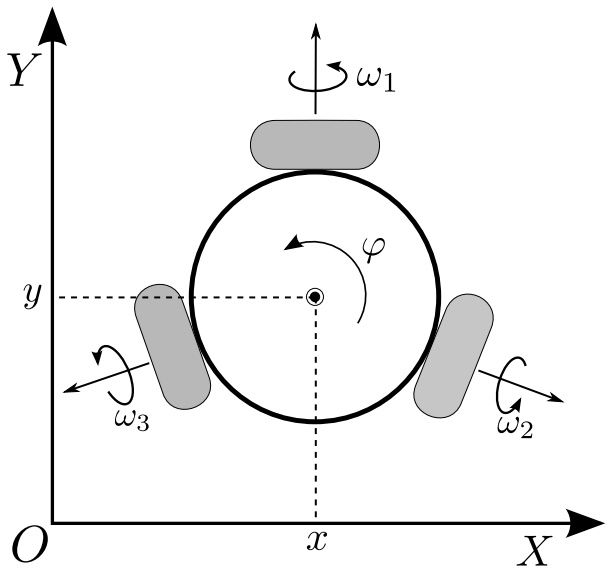

Figura 4.3 – Vista superior de um robô omnidirecional.

Tendo isso em vista, neste trabalho utilizaremos um robô omnidirecional. Para a validação, foi utilizado o Robotino®, um robô omnidirecional produzido pela Festo®, com objetivos primariamente educacionais.

4.1 Robotino®

O Robotino® possui uma grande quantidade de sensores, unidade de controle, duas baterias recarregáveis, uma interface de entrada e saída, além de uma estrutura com espaço para montagem de acessórios como sensores e/ou atuadores.

Ele também dispõe de um sistema operacional Linux instalado na unidade de controle, com uma API (*Application Programming Interface*) instalada, além de um ponto de acesso para comunicação *wireless*. Podemos vê-lo na Fig. 4.4 (BLIESENER et al., 2013).

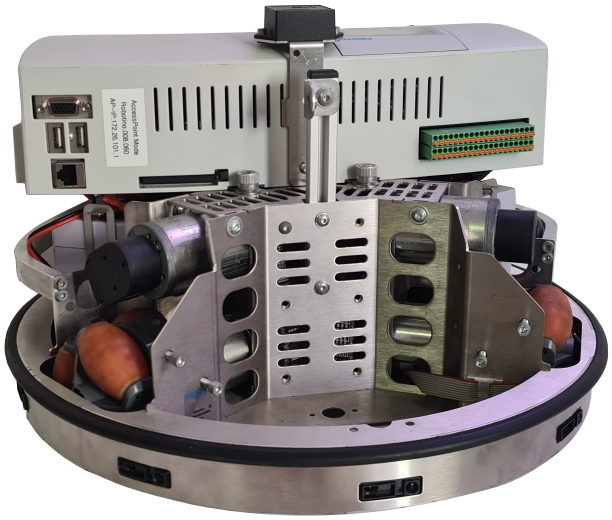

Figura 4.4 – Robotino®.

4.1.1 Sensores e Atuadores

Uma grande gama de sensores pode ser incorporada ao Robotino®, alguns são integrados de fábrica ao chassi e outros podem ser acoplados posteriormente. Dentre os que já vem incorporados, existem nove sensores infra-vermelhos para medição de distância, o sensor anti-colisão, que para o robô no caso de contato com algum objeto que ofereça grande resistência ao movimento. Outro sensor muito útil que pode ser acoplado ao Robotino® se trata da unidade de medição inercial, auxiliando na obtenção e estimação das variáveis de posição e orientação.

O Robotino® possui três unidades motoras individualmente controláveis, cada uma acoplada a uma omni-roda e dispostas a 120 graus uma da outra de forma a permitir o movimento em qualquer posição do plano.

4.1.2 Simulador

Para realizar um estudo preliminar, a Festo® disponibiliza um *software* que permite simular o comportamento do Robotino® em um ambiente virtual como mostrado na Fig. 4.5. Neste trabalho, utilizaremos a versão gratuita do simulador, que pode ser obtida de

forma simples no *site* da Festo®. Apesar de algumas limitações devido ao uso da versão gratuita do Robotino® SIM, como no posicionamento de obstáculos e em modificações no ambiente, o simulador consegue simular de forma satisfatória o desempenho dos sensores e atuadores. O simulador permite integração com diversas linguagens de programação, como Robotino® View, C++, Java, .Net, LabVIEW, MATLAB/Simulink e ROS (*Robot Operating System*). Neste trabalho, a linguagem utilizada foi C++, devido à sua boa velocidade de execução e conexão com a API 1.0.

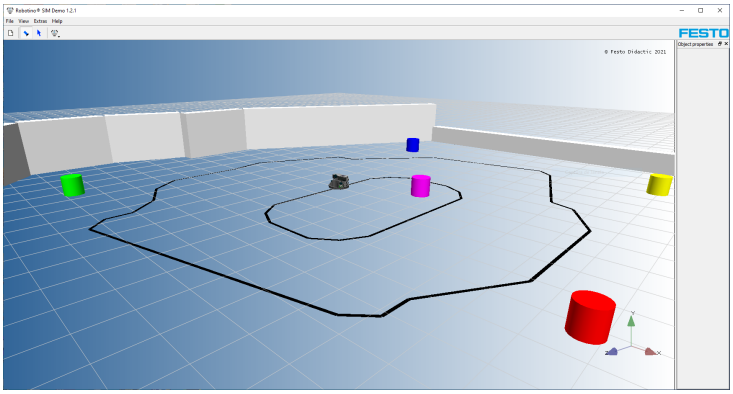

Figura 4.5 – Ambiente virtual do Robotino®.

4.2 Controle Inteligente de Robôs Omnidirecionais

Um robô omnidirecional possui motores acoplados a omni-rodas posicionadas estrategicamente para permitir o movimento no plano horizontal. O sistema pode ser escrito da seguinte forma (RAJ; CZMERK, 2017; BARRETO et al., 2014):

$$(\boldsymbol{M} + \Delta\boldsymbol{M})\ddot{\boldsymbol{\theta}} = \boldsymbol{\tau} - \boldsymbol{\tau}_f, \tag{4.1}$$

onde $\boldsymbol{M} \in \mathbb{R}^{3\times3}$ se trata da matriz de inércia, $\Delta\boldsymbol{M}$ é o termo responsável pelas incertezas de $\boldsymbol{M}$, $\boldsymbol{\theta} = [\theta_1\ \theta_2\ \theta_3]^\top$ é o vetor de estados em relação à posição angular das rodas, $\boldsymbol{\tau} \in \mathbb{R}^3$ é o torque de entrada nos motores, e $\boldsymbol{\tau}_f$ é o vetor de atrito.

As variáveis de estado no referencial inercial, $\boldsymbol{q} = [x\ y\ \psi_z]^\top$, podem ser obtidas de forma conveniente a partir dos estados $\boldsymbol{\theta}$ usando a expressão a seguir (RAJ; CZMERK, 2017):

$$\dot{\boldsymbol{q}} = \boldsymbol{T}_r\dot{\boldsymbol{\theta}}, \tag{4.2}$$

onde $\boldsymbol{T}_r \in \mathbb{R}^{3\times3}$ é a matriz de transformação que relaciona as velocidades angulares de cada roda com as variáveis de estado no referencial inercial.

Combinando a derivada de (4.2) com (4.1), nós podemos obter uma expressão que relaciona o torque aplicado nos motores com as equações de estado inerciais:

$$\ddot{\boldsymbol{q}} = \boldsymbol{f} + \boldsymbol{T}_r \boldsymbol{M}^{-1} \boldsymbol{\tau} + \boldsymbol{d}, \tag{4.3}$$

nas quais $\boldsymbol{f} = \dot{\boldsymbol{T}}_r \boldsymbol{T}_r^{-1} \dot{\boldsymbol{q}}$ e $\boldsymbol{d} = -\boldsymbol{T}_r \boldsymbol{M}^{-1} (\boldsymbol{\tau}_f + \Delta \boldsymbol{M} \ddot{\boldsymbol{\theta}})$ é relacionado com a dinâmica não-modelada e ocasionais perturbações.

4.2.1 Linearização por Realimentação

A lei de controle utilizando a abordagem de linearização por realimentação (SLOTINE; LI, 1991) para o sistema (4.3) é dada por

$$\boldsymbol{\tau} = \boldsymbol{M} \boldsymbol{T}_r^{-1} (-\boldsymbol{f} - \hat{\boldsymbol{d}} + \ddot{\boldsymbol{q}}_d - 2\boldsymbol{\Lambda} \dot{\tilde{\boldsymbol{q}}} - \boldsymbol{\Lambda}^2 \tilde{\boldsymbol{q}}), \tag{4.4}$$

com $\hat{\boldsymbol{d}}$ sendo a estimativa de $\boldsymbol{d}$, $\boldsymbol{\Lambda} \in \mathbb{R}^{3\times3}$ uma matriz diagonal com valores estritamente positivos λ_i, e $\tilde{\boldsymbol{q}} = \boldsymbol{q} - \boldsymbol{q}_d$ o vetor do erro de rastreamento.

Inspirado pela abordagem de controle por modos deslizantes, podemos então definir o seguinte vetor de erro combinado (BESSA et al., 2017):

$$\boldsymbol{s} = \dot{\tilde{\boldsymbol{q}}} + \boldsymbol{\Lambda} \tilde{\boldsymbol{q}}. \tag{4.5}$$

Observando que com $\boldsymbol{s} = \boldsymbol{0}$ o vetor de erro converge exponencialmente a zero e combinando as equações (4.3), (4.4) e (4.5) nós obtemos:

$$\dot{\boldsymbol{s}} + \boldsymbol{\Lambda} \boldsymbol{s} = \boldsymbol{d} - \hat{\boldsymbol{d}}, \tag{4.6}$$

o que significa que a dinâmica em malha fechada representada por $\boldsymbol{s}$ evolui no tempo com base no erro de aproximação $\boldsymbol{d} - \hat{\boldsymbol{d}}$. Desta maneira, é proposta uma rede neural utilizando como entrada as componentes de $\boldsymbol{s}$.

Uma rede neural com funções de base radial recorrente simples com uma camada oculta é proposta para calcular os componentes de $\hat{\boldsymbol{d}}$:

$$\hat{d}_i = \boldsymbol{w}_i^\top \boldsymbol{h}_i(s_i, t), \tag{4.7}$$

onde,

$$\boldsymbol{h}_i(s_i, t) = \boldsymbol{\varphi}_i(s_i) + \alpha \boldsymbol{h}_i(s_i, t - \Delta t) \tag{4.8}$$

nos quais $\boldsymbol{w}_i = [w_{i1} \dots w_{in}]^\top$ é o vetor de pesos, $\boldsymbol{\varphi}_i = [\varphi_{i1} \dots \varphi_{in}]^\top$ representa o vetor com as funções de ativação, n é o número de neurônios na camada oculta e α é a constante de realimentação, também conhecida como fator de esquecimento.

Considerando que a RNN pode realizar uma aproximação com um certo erro de estimativa ε_i, $d_i = \hat{d}_i^* + \varepsilon_i$, onde $\hat{d}_i^*$ é a saída relacionada ao conjunto de pesos ótimos $\boldsymbol{w}_i^*$, as redes neurais recorrentes podem atuar como aproximadores universais.

A partir de (4.6), a condição de atratividade que assegura a estabilidade do controlador proposto pode ser provado por meios de uma análise tipo Lyapunov para os componentes de $\boldsymbol{s}$. Portanto, sendo uma função definida positiva V_i para cada grau de liberdade:

$$V_i(t) = \frac{1}{2}s_i^2 + \frac{1}{2\eta_i}\tilde{\boldsymbol{w}}_i^\top \tilde{\boldsymbol{w}}_i, \tag{4.9}$$

Com η_i sendo uma constante estritamente positiva e $\tilde{\boldsymbol{w}}_i = \boldsymbol{w}_i - \boldsymbol{w}_i^*$.

Considerando que $\dot{\tilde{\boldsymbol{w}}}_i = \dot{\boldsymbol{w}}_i$, a derivada em relação ao tempo de V_i é

$$\begin{aligned}
\dot{V}_i(t) &= s_i\dot{s}_i + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i = [-\lambda_i s_i + d_i - \hat{d}_i]s_i + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i, \\
&= -[\lambda_i s_i - (\hat{d}_i^* + \varepsilon_i - \hat{d}_i)]s_i + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i, \\
&= -[\lambda_i s_i - \varepsilon_i + \tilde{\boldsymbol{w}}_i^\top \boldsymbol{h}_i(s_i,t)]s_i + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i, \\
&= -[\lambda_i s_i - \varepsilon_i]s_i + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top [\dot{\boldsymbol{w}}_i - \eta_i s_i \boldsymbol{h}_i(s_i,t)].
\end{aligned} \tag{4.10}$$

Portanto, atualizando $\boldsymbol{w}_i$ de acordo com $\dot{\boldsymbol{w}}_i = \eta_i s_i \boldsymbol{h}_i(s_i,t)$, $\dot{V}_i$ se torna $\dot{V}_i(t) = -[\lambda_i s_i - \varepsilon_i]s_i \leq -[\lambda_i|s_i| - \varepsilon_i]|s_i|$, sendo semi-definida negativa quando $|s_i| > \varepsilon_i/\lambda_i$. No entanto, os limites de $\boldsymbol{w}_i$ não podem ser assegurados com $\dot{\boldsymbol{w}}_i = \eta_i s_i \boldsymbol{h}_i(s_i,t)$ quando $|s_i| \leq \varepsilon_i/\lambda_i$. Para resolver este problema, o algoritmo de projeção pode ser utilizado para garantir que $\boldsymbol{w_i}$ irá permanecer na região convexa $\Omega_i = \{w_i \in \mathbb{R}^n : \boldsymbol{w}_i^\top \boldsymbol{w}_i \leq \mu_i^2\}$:

$$\dot{\boldsymbol{w}}_i = \begin{cases} \eta_i s_i \boldsymbol{h}_i(s_i,t) & \text{se } ||\boldsymbol{w}_i||_2 < \mu_i \text{ ou} \\ & \text{se } ||\boldsymbol{w}_i||_2 = \mu_i \text{ e } \eta_i s_i \boldsymbol{w}_i^\top \boldsymbol{h}_i(s_i,t) < 0 \\ \left(\boldsymbol{I} - \frac{\boldsymbol{w}_i \boldsymbol{w}_i^\top}{\boldsymbol{w}_i^\top \boldsymbol{w}_i}\right)\eta_i s_i \boldsymbol{h}_i(s_i,t) & \text{c.c.,} \end{cases} \tag{4.11}$$

onde μ_i é o limite superior de $||\boldsymbol{w}_i||_2$.

Sendo $||\boldsymbol{w}_i(0)||_2 \leq \mu_i$, então $|s_i| \leq \varepsilon_i/\lambda_i$ e $||\boldsymbol{w}_i(t)||_2 \leq \mu_i$ com $t \to \infty$. Então, relembrando que $s_i = \dot{\tilde{q}}_i + \lambda_i \tilde{q}_i$, nós temos

$$-\lambda_i^{-1}\varepsilon_i \leq \dot{\tilde{q}}_i + \lambda_i \tilde{q}_i \leq \lambda_i^{-1}\varepsilon_i. \tag{4.12}$$

Assim, multiplicando (4.12) por $e^{\lambda_i t}$ e integrando entre 0 e t:

$$\begin{aligned}
-\lambda_i^{-1}\varepsilon_i e^{\lambda_i t} &\leq \frac{d}{dt}\left(\tilde{q}_i e^{\lambda_i t}\right) \leq \lambda_i^{-1}\varepsilon_i e^{\lambda_i t}, \\
-\frac{\varepsilon_i}{\lambda_i^2}e^{\lambda_i t} - \left[|\tilde{q}_i(0)| + \frac{\varepsilon_i}{\lambda_i^2}\right] &\leq \tilde{q}_i e^{\lambda_i t} \leq \frac{\varepsilon_i}{\lambda_i^2}e^{\lambda_i t} + \left[|\tilde{q}_i(0)| + \frac{\varepsilon_i}{\lambda_i^2}\right], \\
-\frac{\varepsilon_i}{\lambda_i^2} - \left[|\tilde{q}_i(0)| + \frac{\varepsilon_i}{\lambda_i^2}\right]e^{-\lambda_i t} &\leq \tilde{q}_i \leq \frac{\varepsilon_i}{\lambda_i^2} + \left[|\tilde{q}_i(0)| + \frac{\varepsilon_i}{\lambda_i^2}\right]e^{-\lambda_i t}.
\end{aligned} \tag{4.13}$$

Além disso, para $t \to \infty$, teremos:

$$-\frac{\varepsilon_i}{\lambda_i^2} \leq \tilde{q}_i \leq \frac{\varepsilon_i}{\lambda_i^2}. \tag{4.14}$$

Aplicando (4.14) em (4.12), pode ser verificado que

$$-2\frac{\varepsilon_i}{\lambda_i} \leq \dot{\tilde{q}}_i \leq 2\frac{\varepsilon_i}{\lambda_i}. \tag{4.15}$$

Portanto, podemos concluir que o controlador definido por (4.4), (4.7) e (4.11) garante a convergência exponencial para o erro de rastreamento à região fechada $\Xi = \{\tilde{q}_i \in \mathbb{R}^2 : |\tilde{q}_i^{(j)}| \leq (j+1)!\lambda_i^{j-2}\varepsilon_i, j = 0, 1 \text{ e } q_i = x, y, \psi_z\}$.

4.2.2 Modos Deslizantes

Agora, utilizando o método por Modos Deslizantes. A superfície de deslizamento é definida pelo vetor de erro combinado $\boldsymbol{s}$ definido pela Eq. 2.4 e teremos a lei de controle expressa na seguinte forma:

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{T}_r^{-1}(-\boldsymbol{f} - \hat{\boldsymbol{d}} + \ddot{\boldsymbol{q}}_d - \boldsymbol{\Lambda}\dot{\tilde{\boldsymbol{q}}} - \boldsymbol{\kappa}\text{sgn}(\boldsymbol{s})), \tag{4.16}$$

Como discutido anteriormente, para evitar o *chattering*, utilizaremos a função saturação, reescrevendo a lei de controle, temos:

$$\boldsymbol{\tau} = \boldsymbol{M}\boldsymbol{T}_r^{-1}(-\boldsymbol{f} - \hat{\boldsymbol{d}} + \ddot{\boldsymbol{q}}_d - \boldsymbol{\Lambda}\dot{\tilde{\boldsymbol{q}}} - \boldsymbol{\kappa}\text{sat}(\boldsymbol{\phi}^{-1}\boldsymbol{s})), \tag{4.17}$$

onde $\boldsymbol{\phi}$ é uma matriz diagonal com parâmetros ϕ_i positivos que representam a largura da camada limite.

Seguindo de forma análoga ao caso do controlador por linearização por realimentação, combinando (4.3), (4.17) e (4.5), obtemos:

$$\dot{\boldsymbol{s}} + \boldsymbol{\kappa}\text{sat}(\boldsymbol{\phi}^{-1}\boldsymbol{s}) = \boldsymbol{d} - \hat{\boldsymbol{d}}, \tag{4.18}$$

Utilizando os mesmos parâmetros e considerações utilizados anteriormente e a mesma análise sobre a RNN, vamos prosseguir na prova de estabilidade pela análise do tipo Lyapunov para os componentes de **s** mais uma vez. Escolheremos a seguinte função candidata a Lyapunov:

$$V_i(t) = \frac{1}{2}s_{\phi_i}^2 + \frac{1}{2\eta_i}\tilde{\boldsymbol{w}}_i^\top\tilde{\boldsymbol{w}}_i, \tag{4.19}$$

onde $s_{\phi_i} = s_i - \phi\text{sat}(s_i/\phi_i)$, representando a distância até a camada limite ϕ_i.

A derivada em relação ao tempo de V_i é

$$\begin{aligned}\dot{V}_i(t) &= s_{\phi_i}\dot{s}_{\phi_i} + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i = [-\kappa_i \text{sat}(s_i/\phi_i) + d_i - \hat{d}_i]s_{\phi_i} + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i, \\ &= -[\kappa_i \text{sat}(s_i/\phi_i) - (\hat{d}_i^* + \varepsilon_i - \hat{d}_i)]s_{\phi_i} + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i, \\ &= -[\kappa_i \text{sat}(s_i/\phi_i) - \varepsilon_i + \tilde{\boldsymbol{w}}_i^\top \boldsymbol{h}_i(s_i,t)]s_{\phi_i} + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top \dot{\boldsymbol{w}}_i, \\ &= -[\kappa_i \text{sat}(s_i/\phi_i) - \varepsilon_i]s_{\phi_i} + \eta_i^{-1}\tilde{\boldsymbol{w}}_i^\top [\dot{\boldsymbol{w}}_i - \eta_i s_i \boldsymbol{h}_i(s_i,t)]. \end{aligned} \tag{4.20}$$

Atualizando $\boldsymbol{w}_i$ de acordo com $\dot{\boldsymbol{w}}_i = \eta_i s_i \boldsymbol{h}_i(s_i,t)$, e sabendo que durante o modo de aproximação do controlador por modos deslizantes a função saturação é igual à função sinal, teremos que $\dot{V}_i$ se torna $\dot{V}_i(t) = -[\kappa_i \text{sgn}(s_i) - \varepsilon_i]s_{\phi_i}$.

Substituindo $\kappa_i \geq \eta_i + \varepsilon_i$, chegamos a:

$$\dot{V}(t) = \frac{ds_\phi^2}{dt} \leq -\eta|s_\phi| \tag{4.21}$$

Portanto, $\dot{V}(t)$ é definida negativa, então, dada a lei de controle proposta, o sistema converge assintoticamente para a camada limite e como mostrado no Teorema 2.2, se mantém contido dentro dela.

4.3 Implementação

Alguns métodos foram utilizados para possibilitar a implementação dos controladores no Robotino®, entre eles está o derivador por modos deslizantes (SMD - Second-order Sliding Mode Differentiator), utilizado por Bessa et al. (2018), com o objetivo de satisfazer a condição proposta de que todos os estados sejam mensuráveis. A unidade de medição inercial é capaz de obter as variáveis de estado que dizem respeito à posição e orientação (x, y e ψ). Assim, o derivador por modos deslizantes tem como objetivo obter os estados das velocidades. A obtenção do estado estimado $\hat{x}$ pode ser dada através de um sinal conhecido de entrada x como em:

$$\dot{\hat{x}} = -\vartheta \text{ sat}\left(\frac{\hat{x} - x}{\gamma}\right), \tag{4.22}$$

onde ϑ e γ são constantes estritamente positivas.

Como observado por Moreira (2021), é visto que um pequeno ruído pode gerar um grande efeito na sua derivada e para resolver esse problema, foi proposto um filtro passa-baixas de primeira ordem definido por $\dot{x_f} = \beta_0 x - \beta_1 x_f$ onde x_f é o sinal filtrado de x e β_0 e β_1 são constantes positivas.

5 Resultados

Os resultados foram obtidos tanto por meio de simulações no ambiente virtual do Robotino® SIM, quanto por meio de experimentos utilizando o robô omnidirecional Robotino®.

5.1 Simulações

Foram realizadas simulações com o objetivo de verificar o desempenho dos controladores e da influência da RNN para o Robotino® SIM. Sendo o modelo obtido através da Eq. 4.1. Os parâmetros do filtro passa-baixa de primeira ordem comentados anteriormente foram, $\beta_0 = 29,0$ e $\beta_1 = 1,0$. Na RNN, foram utilizadas RBFs do tipo gaussianas como funções de ativação, mostradas na Eq. 5.1.

$$\varphi(\tau_i, \sigma_{i,k}, c_{i,k}) = \exp\left(\frac{-(\tau_i - c_k)^2}{2\sigma_k^2}\right) \tag{5.1}$$

onde foram definidos sete neurônios k com seus respectivos centros c_i da seguinte forma:

$$\mathbf{c}_i = \left[-\frac{\xi_i}{2}, -\frac{\xi_i}{4}, -\frac{\xi_i}{8}, 0, \frac{\xi_i}{8}, \frac{\xi_i}{4}, \frac{\xi_i}{2}, \right]^\top \tag{5.2}$$

e larguras σ_i definidas por

$$\boldsymbol{\sigma_i} = \left[\frac{\xi_i}{3}, \frac{\xi_i}{5}, \frac{\xi_i}{15}, \frac{\xi_i}{30}, \frac{\xi_i}{15}, \frac{\xi_i}{5}, \frac{\xi_i}{3}, \right]^\top \tag{5.3}$$

além disso, $\boldsymbol{\xi} = [0,5; 0,5; 2,0]^\top$, o vetor dos pesos foi inicializado com 0 e sua atualização foi dada de acordo com a Eq. 4.11, as taxas de aprendizagem $\boldsymbol{\nu} = [10,0; 10,0; 10,0]^\top$, o limitante da norma do vetor dos pesos $\mu = 0.1$ e a constante de realimentação da RNN $\alpha = 0,3$. O vetor de estados desejados $\boldsymbol{x_d} = [acos(ft); -asen(ft); -ft - \pi/2]^\top$, com amplitude $a = 0,5$ e frequência $f = \pi/15$ para um tempo de simulação de 60 segundos.

5.1.1 Linearização por Realimentação

Para o controlador por Linearização por Realimentação, a lei de controle utilizada foi segundo a Eq. 4.4. Além disso o ganho do controlador $\boldsymbol{\Lambda} = [1,0; 1,0; 1,0]^\top$. Os resultados podem ser observados nas Figs. 5.1-5.5 e da Tabela 5.1, nela estão explicitados os valores de Erro médio quadrático e Energia médios para cada grau de liberdade. Esta métrica de energia é baseada em Bessa et al. (2018) e representada da seguinte forma:

$$\bar{E}_i = \frac{1}{t_f}\int_0^{t_f} |\tau_i| \; dt. \tag{5.4}$$

Além disso, as unidades para os erros mostradas na Tabela 5.1 são as mesmas dos gráficos da Fig. 5.2, já para a energia, são as mesmas unidades mostradas na Fig. 5.4.

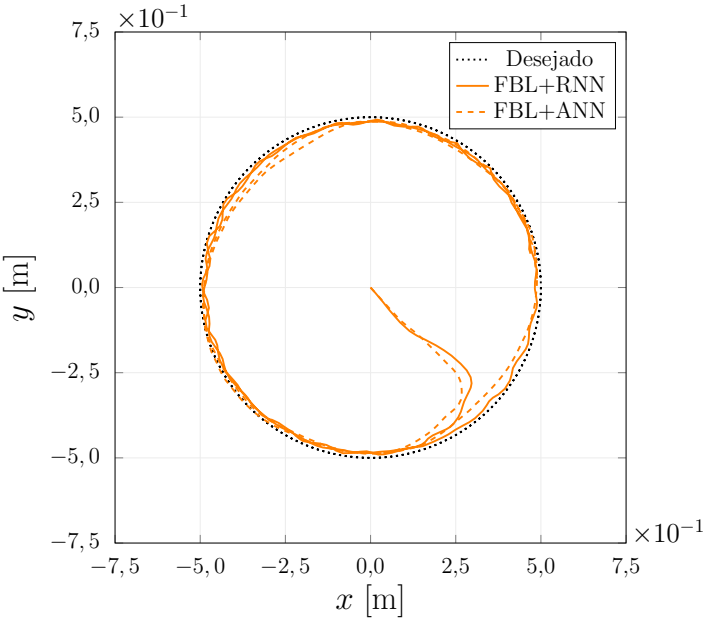

Figura 5.1 – Trajetória do Robotino® utilizando a Linearização por Realimentação.

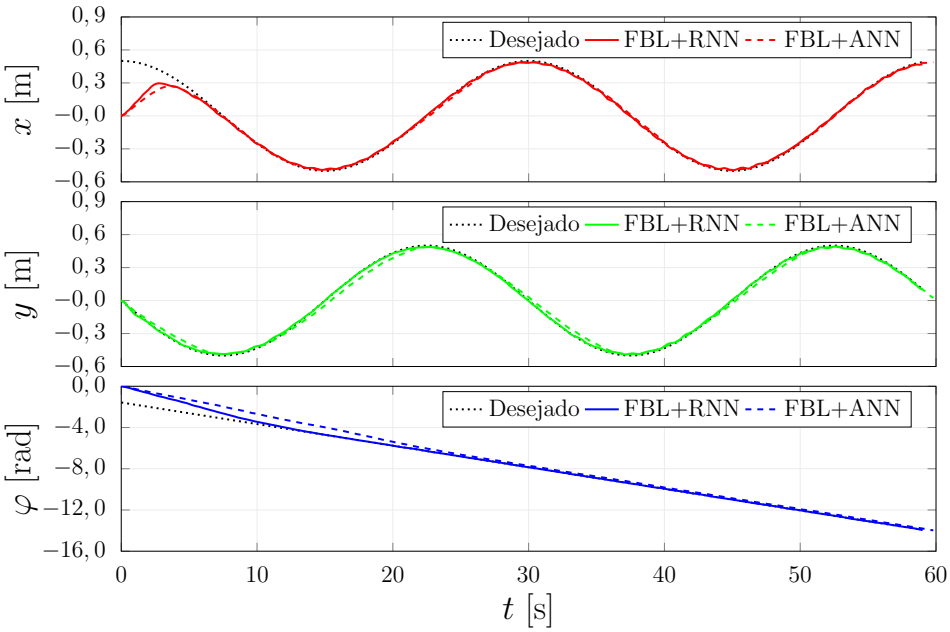

Figura 5.2 – Posição em cada grau de liberdade utilizando a Linearização por Realimentação.

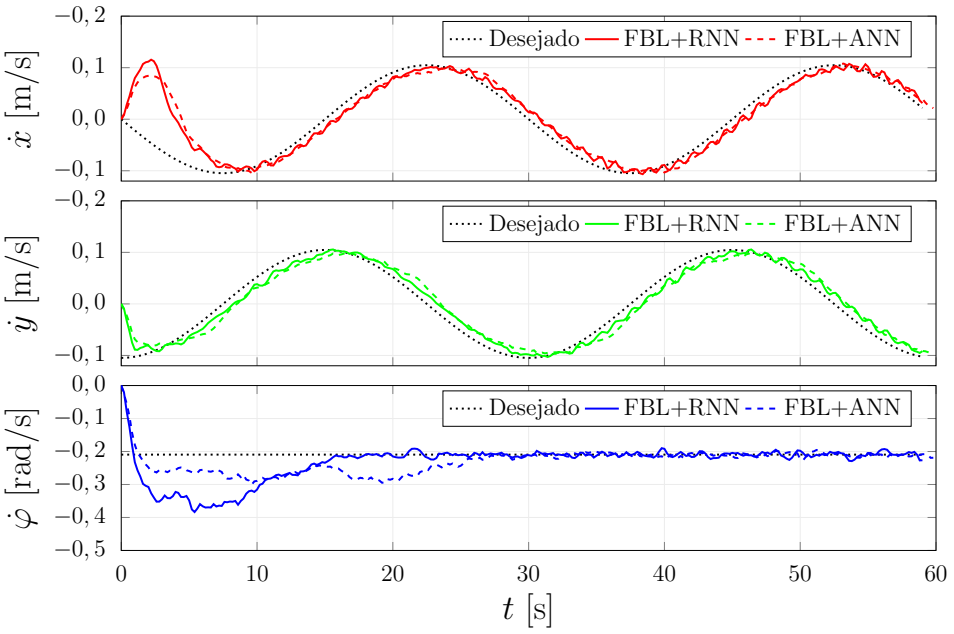

Figura 5.3 – Velocidades em cada grau de liberdade utilizando a Linearização por Realimentação.

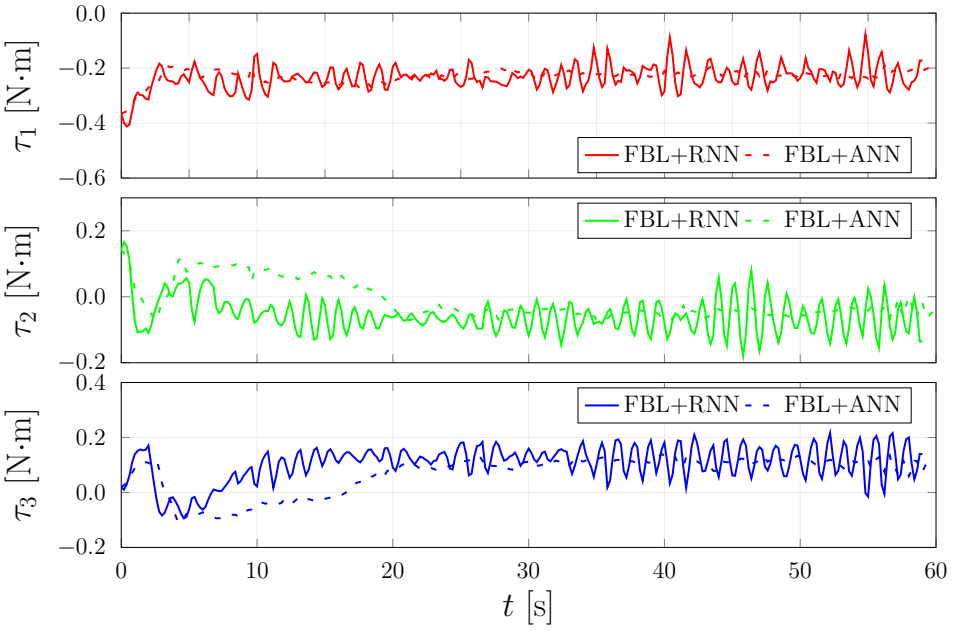

Figura 5.4 – Esforço de controle para cada atuador utilizando a Linearização por Realimentação.

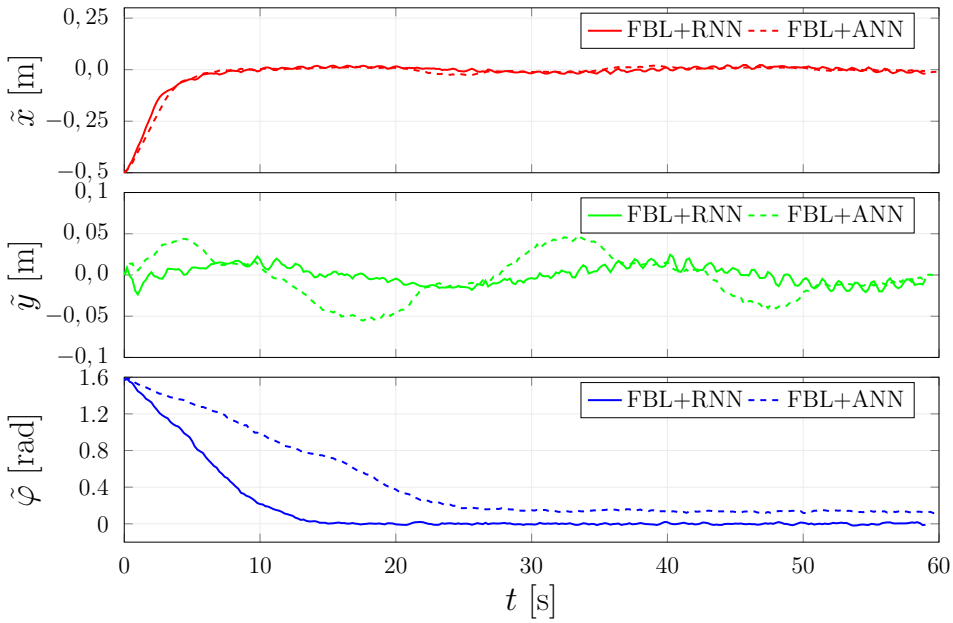

Figura 5.5 – Erro na posição para cada grau de liberdade utilizando a Linearização por Realimentação.

Observando a partir dos gráficos das Fig. 5.2 e 5.3, as propostas de controle com ambas as abordagens para compensação permitiram tanto o rastreamento da trajetória quanto da velocidade desejadas. Podemos observar que com a adição da recorrência, houve uma melhora no desempenho relativo ao rastreamento de trajetória, como pode ser visto tanto na Fig. 5.1 como um todo, quanto individualmente para cada grau de liberdade nas Figs. 5.2 e 5.5. Além disso, se tratando do ângulo ψ, a incorporação da RNN eliminou o erro em regime permanente que existia utilizando a rede neural tradicional. Entretanto, como visto na Fig. 5.4, a adição da recorrência também adicionou uma pequena oscilação no esforço de controle.

5.1.2 Controlador por Modos Deslizantes

Já utilizando o controlador por Modos Deslizantes, a sua lei de controle foi feita de acordo com a Eq. 4.17 e os parâmetros do controlador foram $\boldsymbol{\kappa} = [0,16; 0,16; 2,01]^\top$ e $\boldsymbol{\phi} = [0,5; 0,5; 1,5]^\top$. Podemos observar os resultados através das Figs. 5.6-5.10 e da Tabela 5.1.

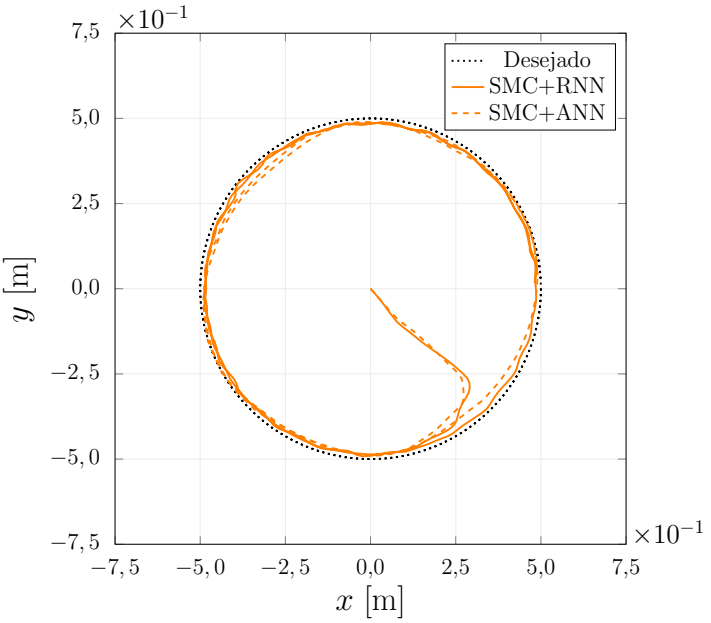

Figura 5.6 – Trajetória do Robotino® utilizando o controlador por Modos Deslizantes.

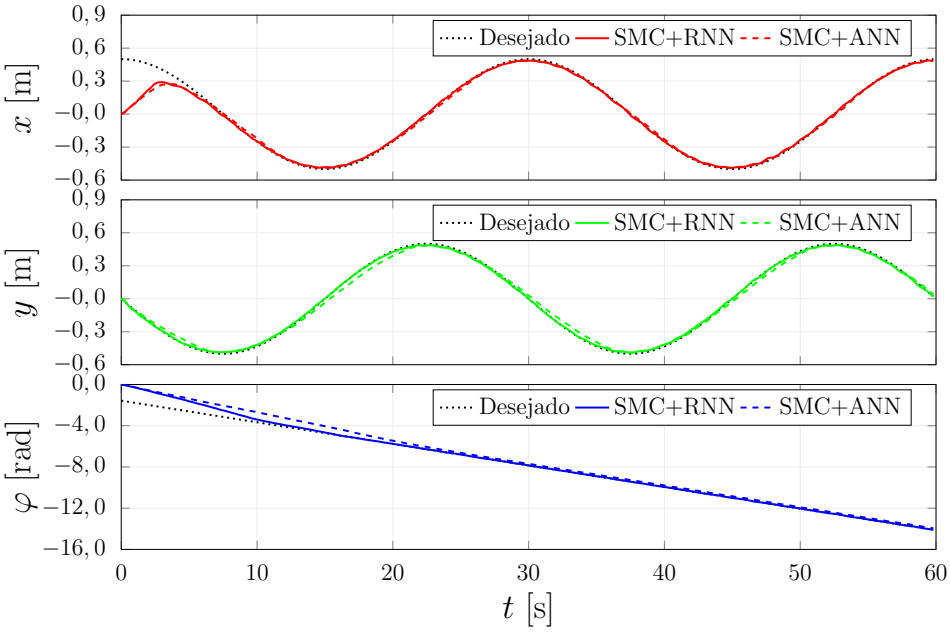

Figura 5.7 – Posição em cada grau de liberdade utilizando o controlador por Modos Deslizantes.

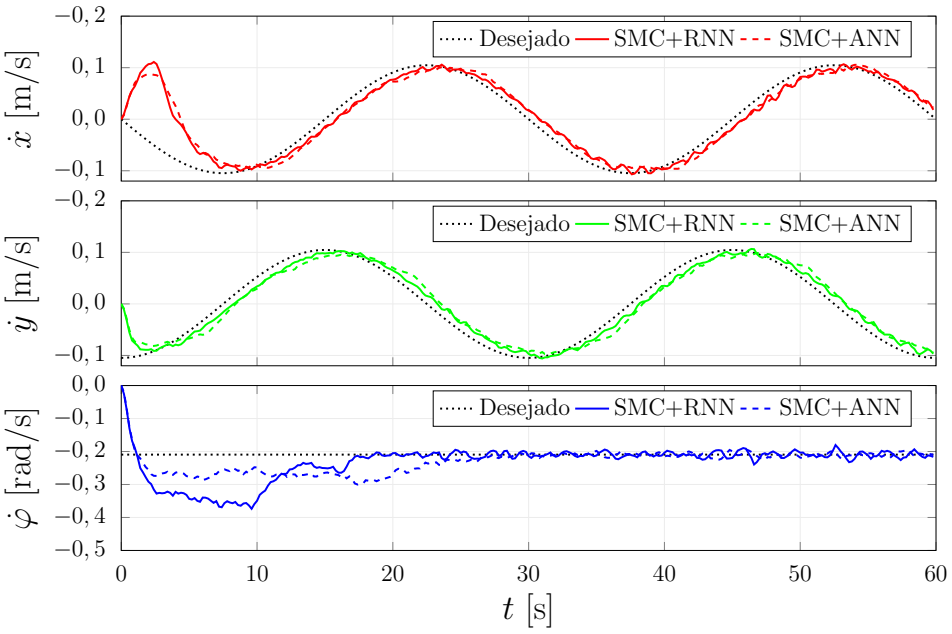

Figura 5.8 – Velocidades em cada grau de liberdade utilizando o controlador por Modos Deslizantes.

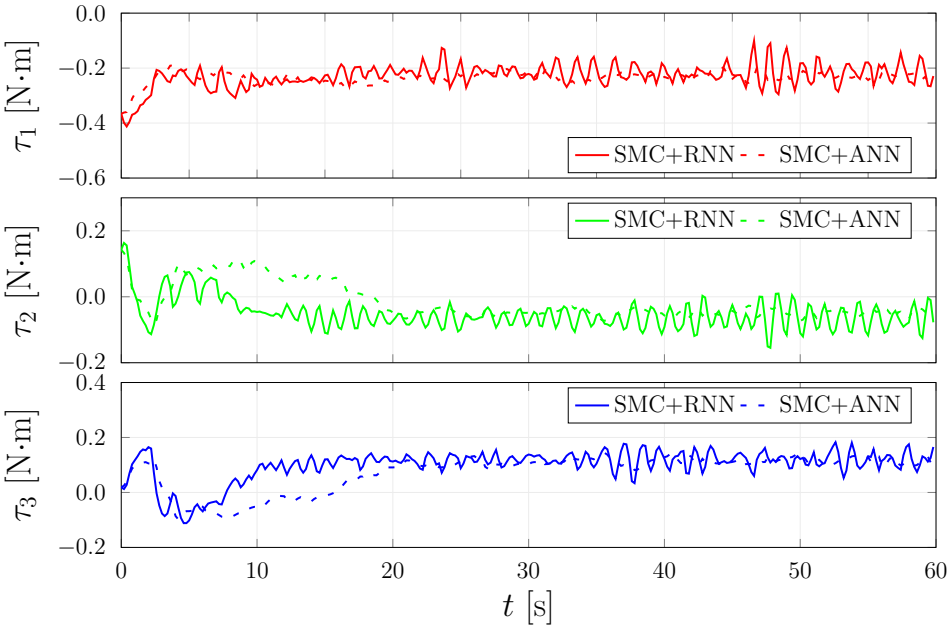

Figura 5.9 – Esforço de controle para cada atuador utilizando o controlador por Modos Deslizantes.

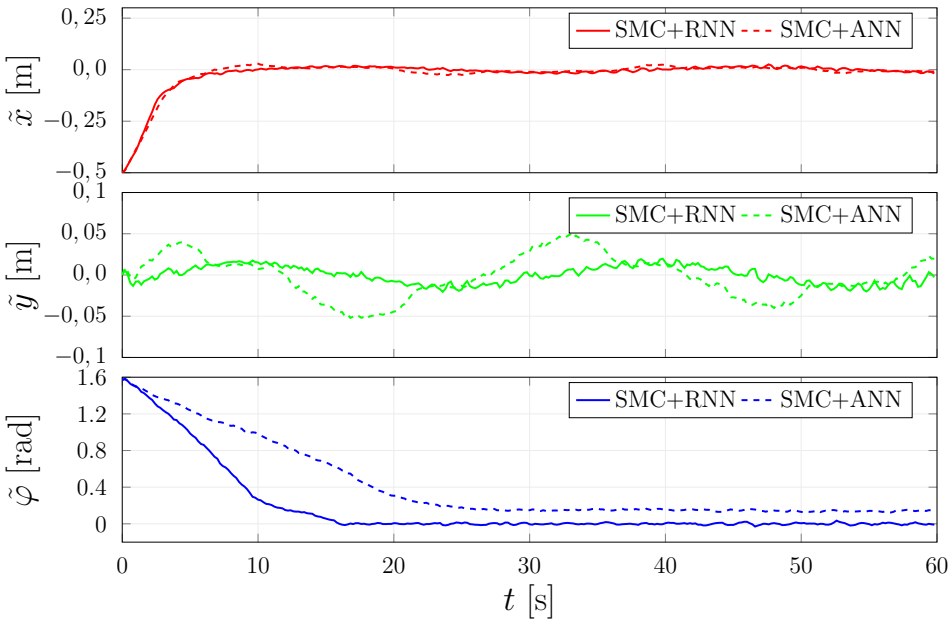

Figura 5.10 – Erro na posição para cada grau de liberdade utilizando o controlador por Modos Deslizantes.

Tabela 5.1 – Erro e Energia médios para cada grau de liberdade.

Controlador	Erro $\overline{x}$	Erro $\overline{y}$	Erro $\overline{\varphi}$	Energia $\overline{\tau_1}$	Energia $\overline{\tau_2}$	Energia $\overline{\tau_3}$
FBL + ANN	0.020650	0.003077	0.438720	0.231247	0.052395	0.090544
FBL + RNN	0.018096	0.000378	0.159864	0.233940	0.065892	0.112778
SMC + ANN	0.019268	0.002110	0.423370	0.234517	0.051834	0.092246
SMC + RNN	0.018686	0.000177	0.176790	0.228002	0.061878	0.108345

Com a implementação deste novo controlador, foi possível observar através das Figs. 5.6 - 5.8 e 5.10 que o rastreamento de trajetória e velocidade foram realizados com um ótimo desempenho, além de podermos observar mais uma vez que a inclusão da recursão na rede neural promoveu uma melhora neste rastreamento, como evidenciado nos valores de erro médio existentes na 5.1, com o detrimento da inclusão de uma pequena oscilação no esforço de controle, como pode ser visto na Fig. 5.9, menor do que no caso do controlador por Linearização por Realimentação. Também pode ser observado, através da Tabela 5.1, que a inserção da recursão na rede não aumentou o esforço de controle tanto quanto no caso da Linearização por Realimentação.

5.2 Resultados Experimentais

Os resultados experimentais foram obtidos utilizando o Robotino®. Para isso, um roteador foi acoplado ao mesmo com o objetivo de acessá-lo remotamente. Foram feitos diversos experimentos variando os parâmetros α e ν da RNN para uma análise mais profunda sobre o efeito da incorporação da recursão na rede. Com o objetivo de tentar

transladar as análises feitas por meio de simulações para os experimentos, os parâmetros utilizados foram mantidos iguais. Houve apenas uma mudança em um dos parâmetros do filtro passa-baixa, estes foram $\beta_0 = 35,0$ e $\beta_1 = 1,0$, devido ao maior ruído do experimento.

5.2.1 Linearização por Realimentação

Utilizando praticamente mesmos parâmetros utilizados no simulador do Robotino®, obtivemos os seguintes resultados experimentais:

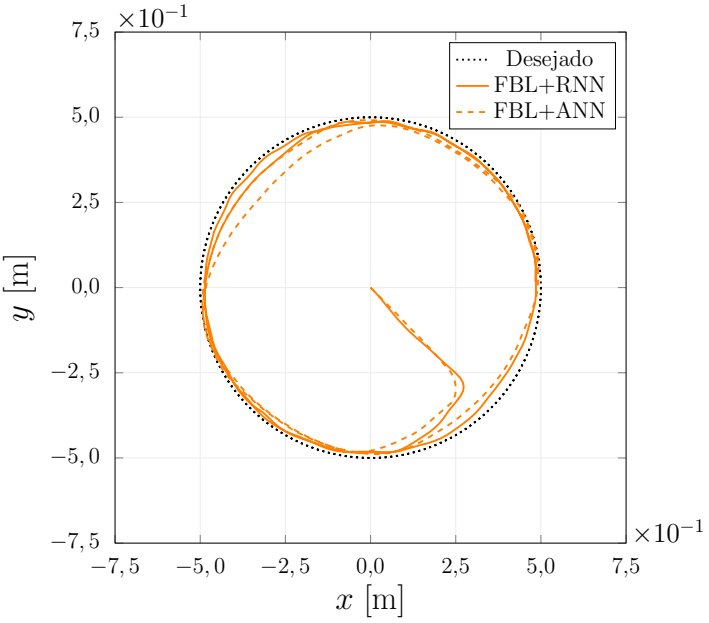

Figura 5.11 – Trajetória do Robotino® utilizando o controlador por Linearização por Realimentação.

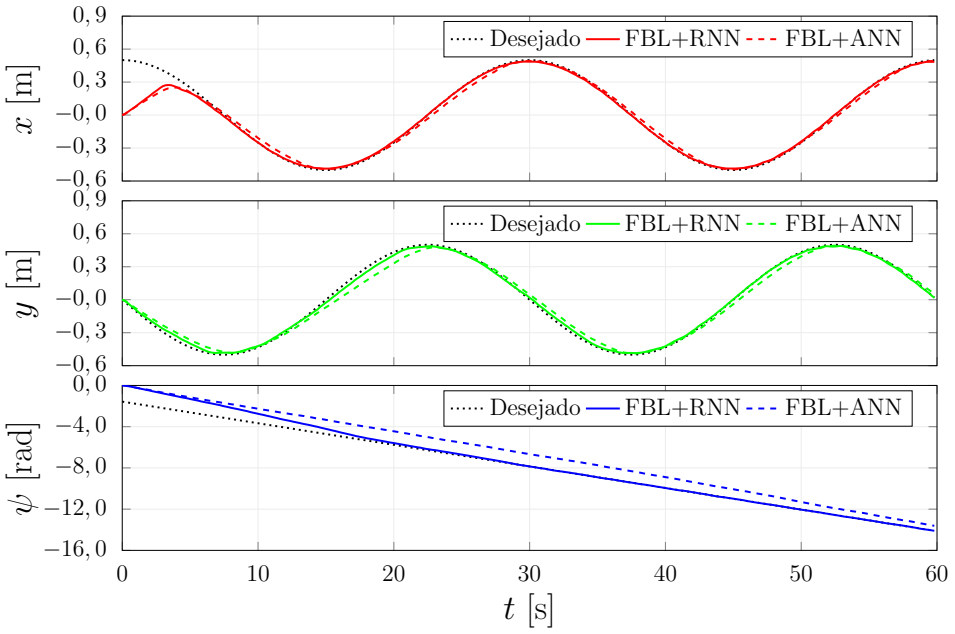

Figura 5.12 – Posição em cada grau de liberdade utilizando o controlador por Linearização por Realimentação.

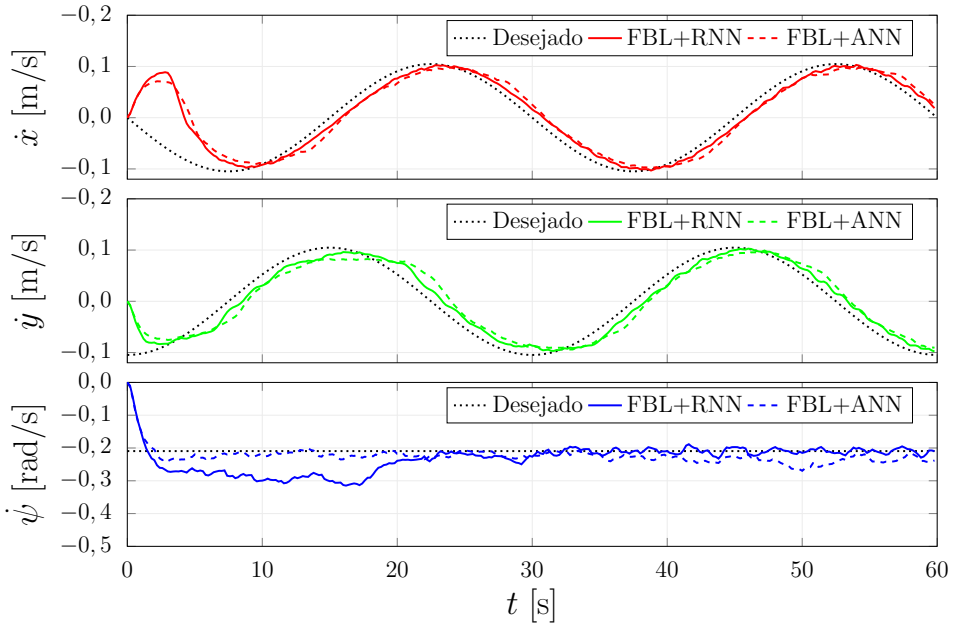

Figura 5.13 – Velocidades em cada grau de liberdade utilizando o controlador por Linearização por Realimentação.

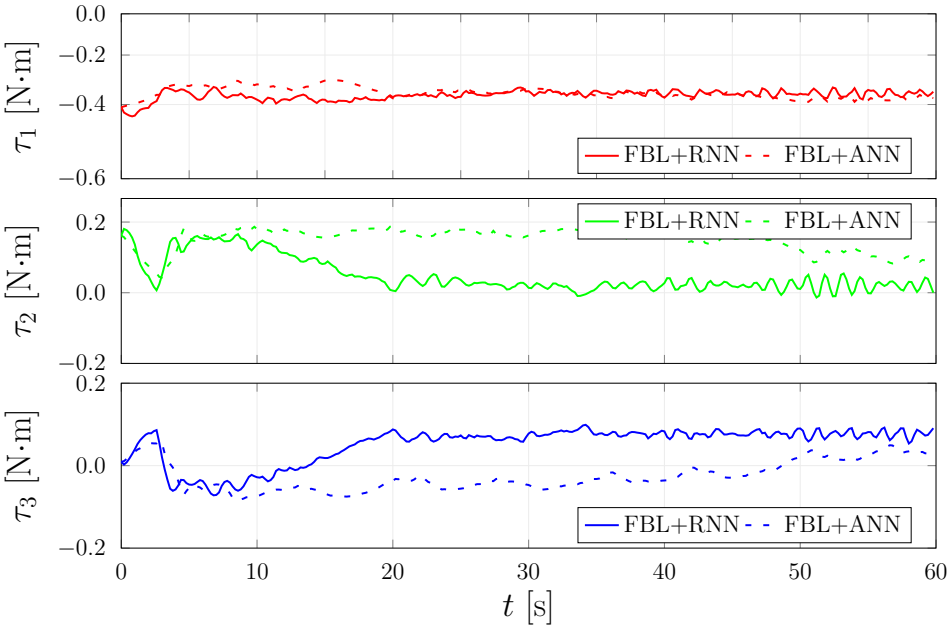

Figura 5.14 – Esforço de controle para cada atuador utilizando o controlador por Linearização por Realimentação.

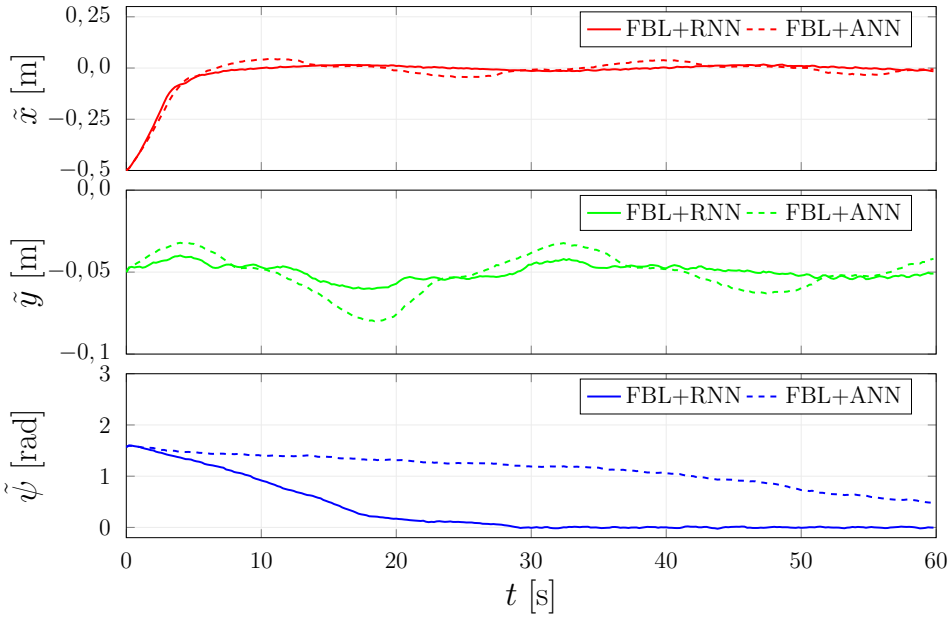

Figura 5.15 – Erro na posição para cada grau de liberdade utilizando o controlador por Linearização por Realimentação.

Podemos observar através das Figs. 5.11 - 5.15 que as análises feitas através das simulações podem ser transladadas para o contexto experimental, ou seja, para este caso, a inclusão da recursão na ANN produziu uma diminuição do erro de rastreamento, sem um ganho considerável de esforço de controle.

Entretanto, com o objetivo de investigar de forma mais profunda as variações que a incorporação da recorrência acarreta no sistema com parâmetros diferentes, variamos

tanto o parâmetro α da RNN quanto o a taxa de aprendizagem da rede $\boldsymbol{\nu}$. Para tal, como se trata de um sistema com três graus de liberdade, sendo dois deles lineares e um angular, primeiramente o ângulo foi transformado em arco para podermos obter uma medida linear.

Além disso, os valores dos três graus de liberdade foram normalizados para que não houvesse um grau de liberdade dominante com base na magnitude de seus valores. Após isso, estes valores normalizados dos três graus de liberdade foram combinados em forma de média para podermos obter uma representação em duas dimensões. Esses valores foram comparados com o a média dos gastos energéticos tal qual a Eq. 5.4 dos três atuadores podendo ser representada da seguinte forma:

$$\bar{E}_q = \frac{\sum_{i=1}^{3} \bar{E}_i}{3} \tag{5.5}$$

Assim, foram comparados os valores dessas medidas de erro e energia para todos os experimentos e representados na Fig. 5.16.

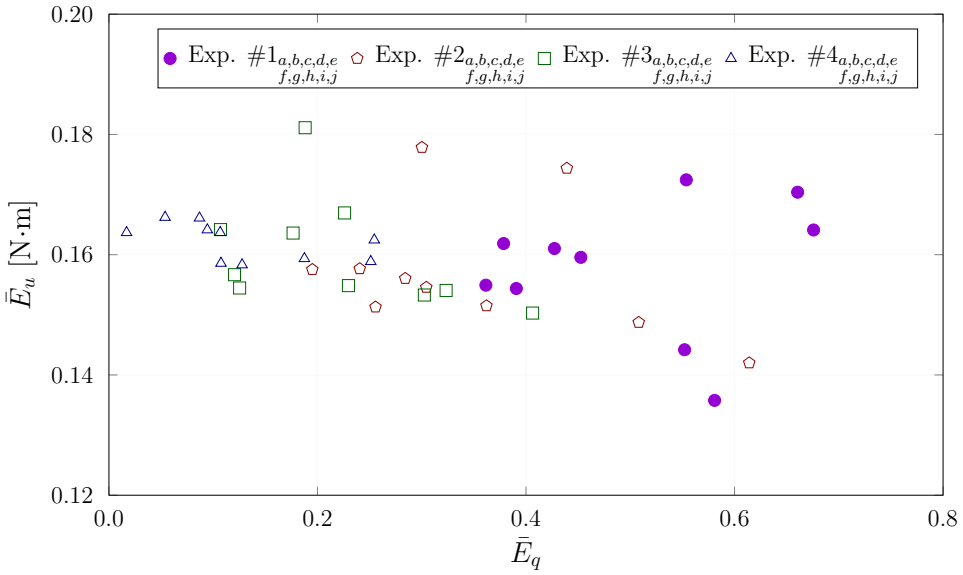

Figura 5.16 – Comparação entre a medida de erro normalizado e o gasto energético.

A descrição da codificação utilizada no gráfico é introduzida a seguir:

- Os experimentos demarcados com o #1 correspondem à rede neural sem recursão, ou seja, $\alpha_1 = 0.0$;
- Os grupos de experimentos demarcados com {#2, #3, #4} correspondem a RNN, sendo cada um deles $\{\alpha_2 = 0.1, \alpha_3 = 0.2, \alpha_4 = 0.3\}$;
- Já para a variação dos parâmetros de $\boldsymbol{\nu}$, os grupos demarcados com {a, b, c, d, e, f, g, h, i, j} correspondem aos valores de $\{\boldsymbol{\nu}_a = [10; 10; 10]$, $\boldsymbol{\nu}_b = [10; 10; 12]$, $\boldsymbol{\nu}_c = [10; 10; 8]$, $\boldsymbol{\nu}_d = [5; 5; 10]$, $\boldsymbol{\nu}_e = [6; 6; 10]$, $\boldsymbol{\nu}_f = [7; 7; 10]$, $\boldsymbol{\nu}_g = [8; 8; 10]$, $\boldsymbol{\nu}_h = [9; 9; 10]$, $\boldsymbol{\nu}_i = [9; 9; 12]$, $\boldsymbol{\nu}_j = [9; 9; 8]\}$.

Ou seja, para exemplificar, o experimento $\#1_b$ possui valor de $\alpha = 0.0$ e $\boldsymbol{\nu} = [10; 10; 12]$.

A partir da Fig. 5.16, podemos observar que para diferentes experimentos, a incorporação da recursão promoveu uma diminuição do erro, sendo ela maior para os valores de $\alpha = 0.3$, além de diminuir a dispersão dos resultados para diferentes valores de fator de aprendizagem. Para ir mais a fundo nesses resultados, a Fig. 5.17 mostra dois diagramas de caixa, um para a medida de erro normalizado e outro para o gasto energético.

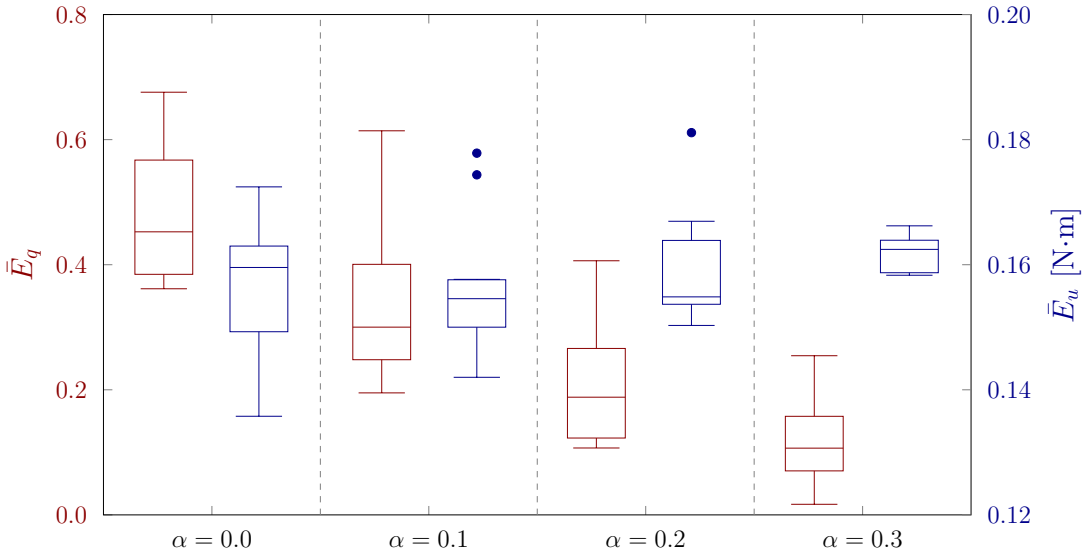

Figura 5.17 – Diagramas de caixa para o erro normalizado e o gasto energético.

Assim, fazendo uma análise estatística que o diagrama de caixa nos permite, podemos observar que o erro de fato diminuiu, assim como a dispersão dos resultados, demonstrado pela diminuição do desvio padrão com o aumento de α, mostrando uma maior robustez da estratégia. Além disso, podemos notar também que o gasto energético teve apenas um aumento discreto, inclusive dentro do desvio padrão do resultado sem a recursão na rede, o que mostra que a inclusão da recursão não necessariamente acarreta em aumento do gasto energético.

5.2.2 Controle por Modos Deslizantes

Para o controlador por Modos Deslizantes, no experimento foram utilizados os mesmos parâmetros da simulação, exceto o parâmetro do filtro já discutido anteriormente. Assim, teremos os seguintes resultados:

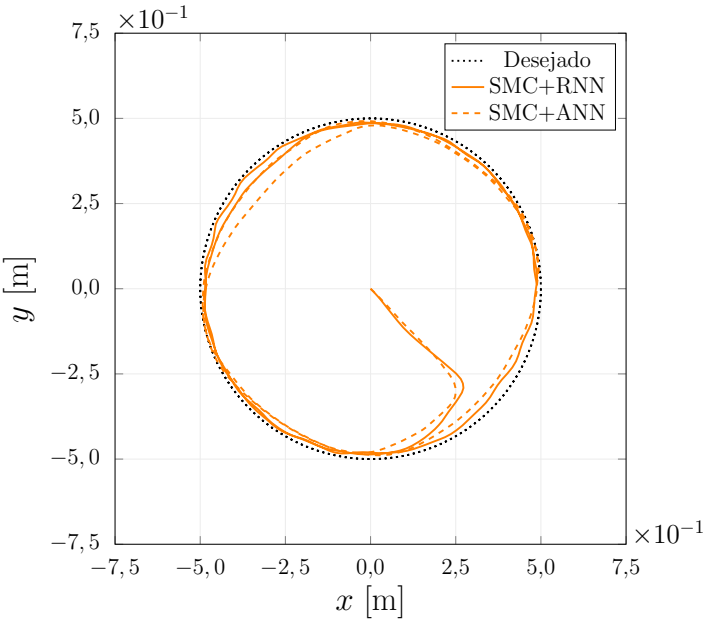

Figura 5.18 – Trajetória do Robotino® utilizando o controlador por Modos Deslizantes.

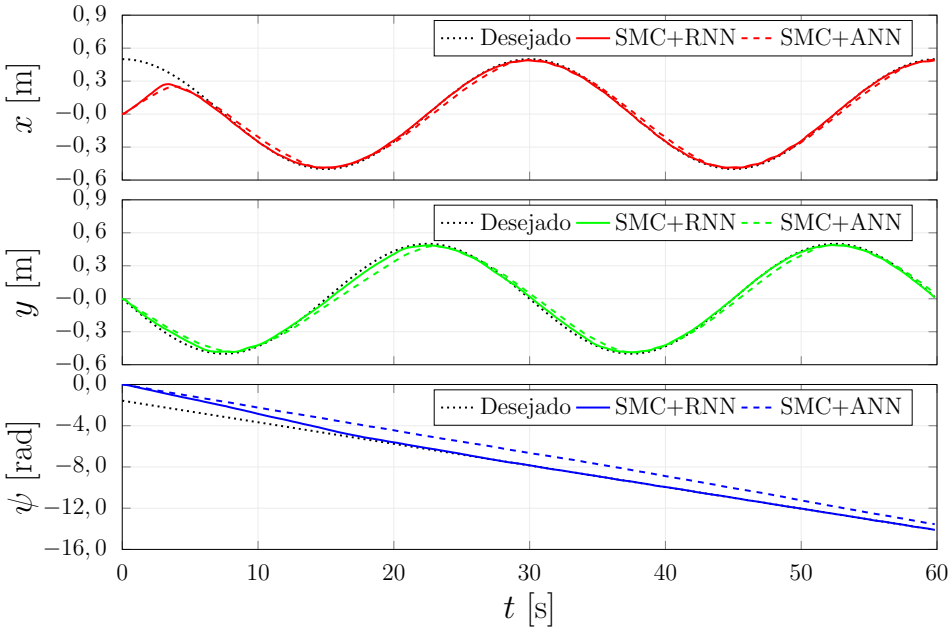

Figura 5.19 – Posição em cada grau de liberdade utilizando o controlador por Modos Deslizantes.

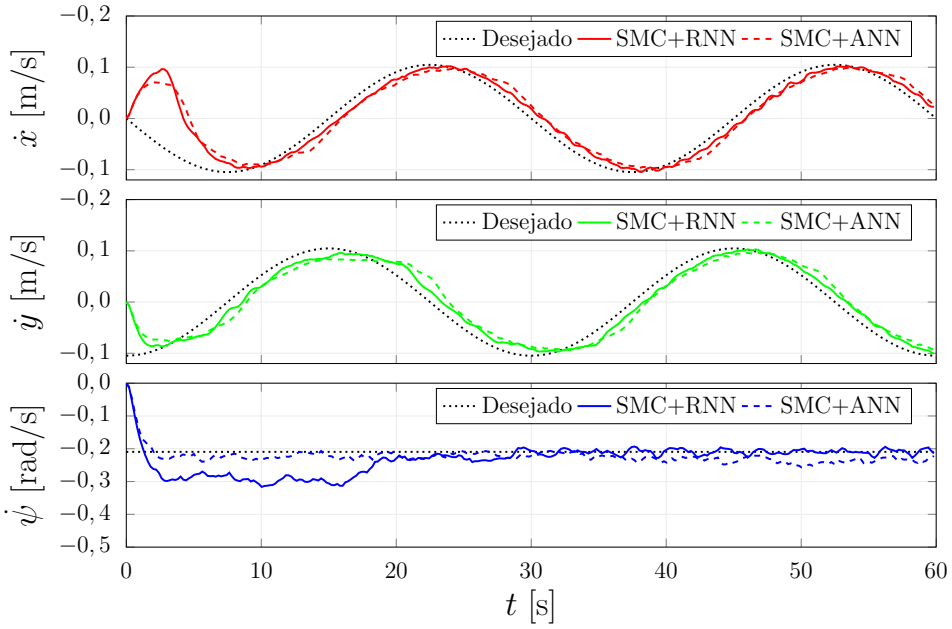

Figura 5.20 – Velocidades em cada grau de liberdade utilizando o controlador por Modos Deslizantes.

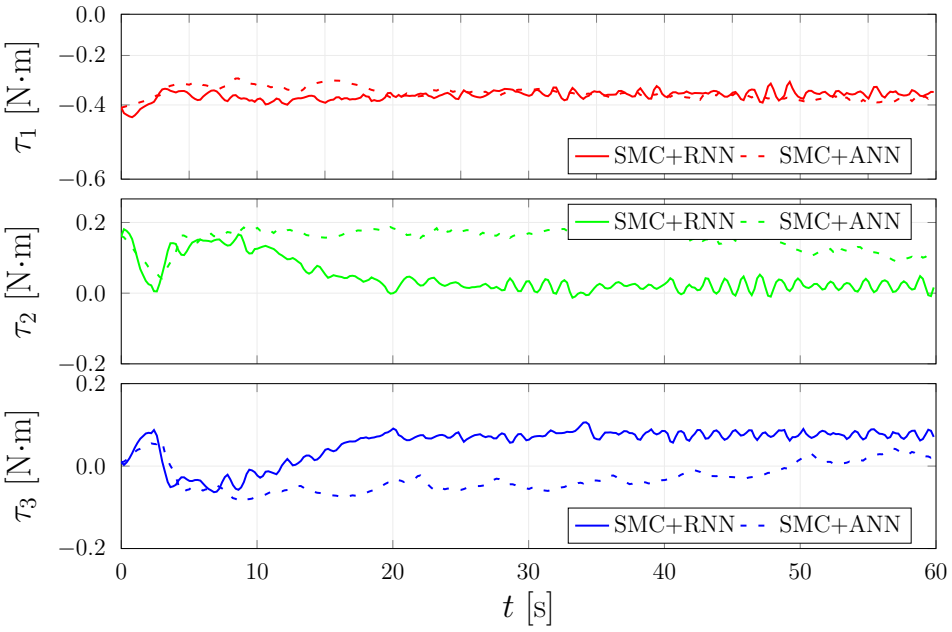

Figura 5.21 – Esforço de controle para cada atuador utilizando o controlador por Modos Deslizantes.

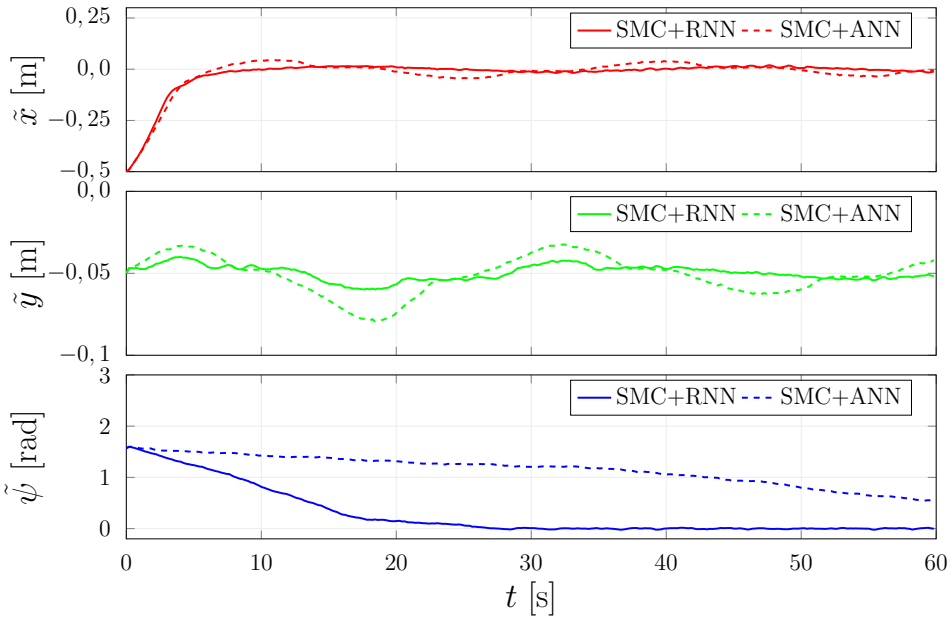

Figura 5.22 – Erro na posição para cada grau de liberdade utilizando o controlador por Modos Deslizantes.

Podemos observar a partir das Figs. 5.18-5.22 que, tal qual no caso do controlador por Linearização por Realimentação, as análises feitas a partir das simulações também podem ser observadas no âmbito experimental, mostrando que o simulador utilizado representa bem o sistema físico.

Mais uma vez iremos analisar o efeito da variação dos parâmetros α e $\boldsymbol{\nu}$ nos experimentos, todos os procedimentos utilizados para essa análise na seção anterior foram realizados também para o controlador por Modos Deslizantes. Obtendo, assim, a Fig. 5.23 mostrada a seguir:

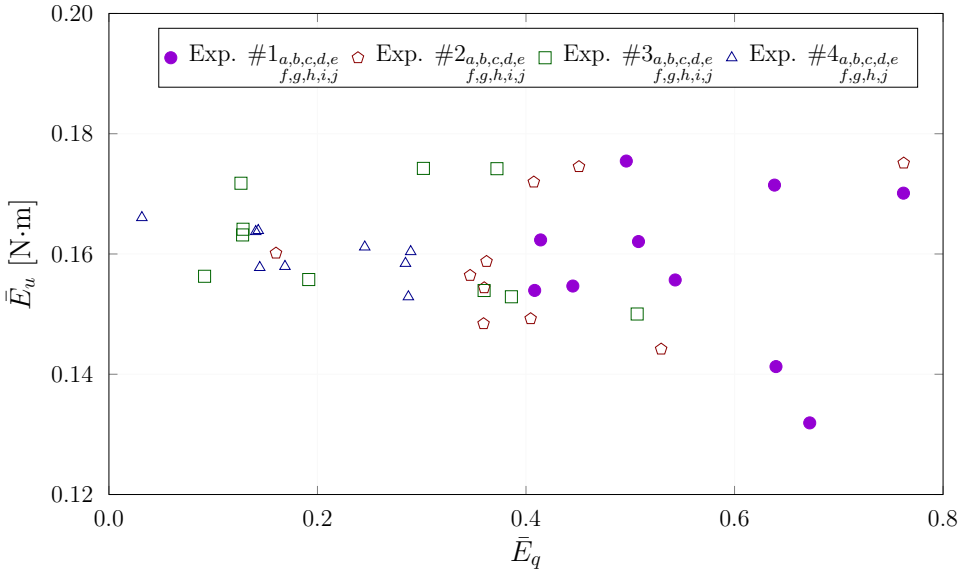

Figura 5.23 – Comparação entre a medida de erro normalizado e o gasto energético para o controlador por Modos Deslizantes.

A descrição dos experimentos é a mesma da seção anterior, por exemplo, o Experimento $\#3_h$ possui $\alpha = 0.2$ e $\boldsymbol{\nu} = [9; 9; 10]$.

Assim como no controlador por Linearização por Realimentação, podemos ver a partir da Fig. 5.23 que para os experimentos realizados, a incorporação da recursão não só diminuiu o erro, quanto a dispersão dos experimentos. Mais uma vez, para uma melhor análise dos resultados, a Fig. 5.24 mostra dois diagramas de caixa com os resultados tanto para o erro normalizado quanto para o gasto energético.

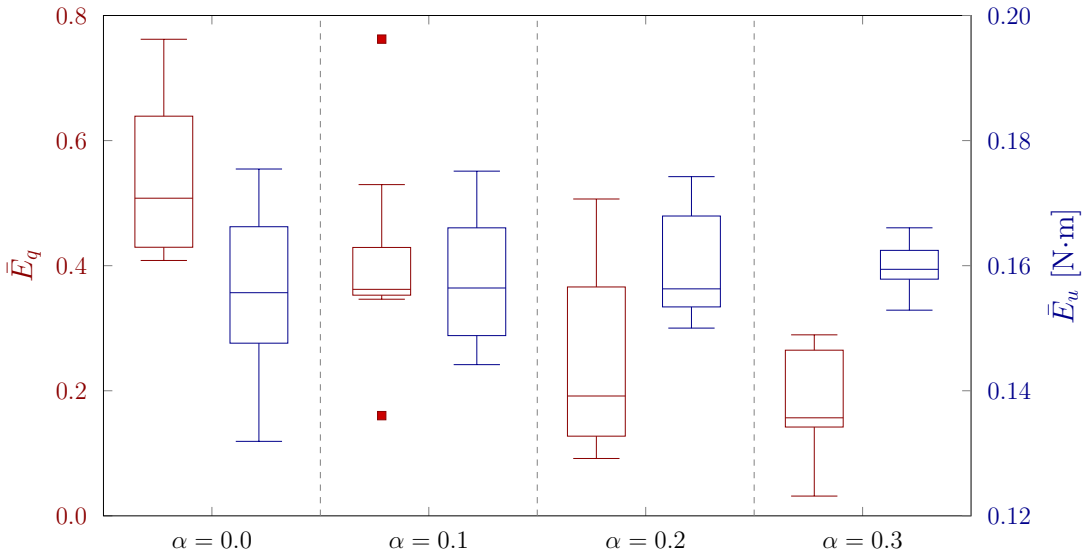

Figura 5.24 – Diagramas de caixa para o erro normalizado e o gasto energético para o controlador por Modos Deslizantes.

Assim, vendo pela abordagem estatística dos diagramas de caixa, podemos observar

que com o aumento de α houve uma diminuição da média do erro, uma diminuição do desvio padrão do gasto energético e um aumento muito discreto do mesmo, mostrando uma maior confiabilidade nos resultados obtidos com essa estratégia.

6 Considerações Finais

Tendo em vista o grande desafio de engenharia que se trata do controle de sistemas dinâmicos e a importância da robótica móvel na área de sistemas autônomos, este trabalho propõe controladores inteligentes de natureza não linear para realizar o controle de um robô omnidirecional. Devido à sua flexibilidade, robôs móveis muitas vezes são implementados em indústrias sem grandes alterações nos *layouts* de fábrica. Sendo assim, se faz necessário um controle bastante preciso para que apesar de haverem caminhos estreitos e obstáculos, o robô possa realizar suas tarefas satisfatoriamente.

Foram implementados dois controladores para a resolução deste problema, ambos não lineares, o controlador por Linearização por Realimentação e o controlador por Modos Deslizantes. Ambos também utilizaram estratégias de compensação de incertezas através de Redes Neurais Artificiais, a rede escolhida foi uma Rede Neural Recorrente com funções de ativação de base radial. Devido aos desafios da implementação de redes recorrentes no contexto de sistemas embarcados, foi utilizada a arquitetura de Rede Recorrente simples. Além disso, as propriedades de estabilidade dos controladores foram provadas de acordo com o princípio de estabilidade assintótica proposto por Lyapunov.

As estratégias propostas foram avaliadas por meios de simulações e experimentos, utilizando o Robotino®, um robô omnidirecional desenvolvido pela Festo®, com propósitos educacionais. Diversos experimentos foram realizados e foi observado que os controladores propostos puderam realizar o rastreamento de trajetória com um bom desempenho. Além disso, através de análises estatísticas se observou que a incorporação da recursão na rede neural, tanto para o controlador por Linearização por Realimentação quanto para o controlador por Modos Deslizantes ajudou a diminuir a média dos erros de rastreamento entre os experimentos, um resultado bastante interessante considerando a necessidade de um controle preciso, e além disso, manteve-se praticamente o mesmo nível de gasto energético. Ademais, a incorporação da recursão também diminuiu o desvio padrão dos resultados, aumentando a confiabilidade da estratégia.

Como sugestão para possíveis trabalhos futuros, é recomendada a implementação de diferentes arquiteturas das redes neurais recorrentes, como o modelo de Hopfield ou uma arquitetura LSTM, o estudo da estabilidade dessas redes como sistemas dinâmicos através do método de Lyapunov, além da análise da possibilidade de implementação das mesmas no hardware limitado de sistemas embarcados.

Referências

ARDIYANTO, I. Task oriented behavior-based state-adaptive pid (proportional integral derivative) control for low-cost mobile robot. In: *2010 Second International Conference on Computer Engineering and Applications*. [S.l.: s.n.], 2010. v. 1, p. 103–107. Citado na página 1.

BARRETO, J. C. L. S. et al. Design and implementation of model-predictive control with friction compensation on an omnidirectional mobile robot. *IEEE/ASME Transactions on Mechatronics*, v. 19, n. 2, p. 467–476, 2014. Citado na página 23.

BESSA, W. M. Some remarks on the boundedness and convergence properties of smooth sliding mode controllers. *International Journal of Automation and Computing*, v. 6, n. 2, p. 154–158, 2009. ISSN 14768186. Citado 2 vezes nas páginas 10 e 12.

BESSA, W. M. et al. A Biologically Inspired Framework for the Intelligent Control of Mechatronic Systems and Its Application to a Micro Diving Agent. *Mathematical Problems in Engineering*, v. 2018, 2018. ISSN 15635147. Citado 3 vezes nas páginas 13, 27 e 28.

BESSA, W. M. et al. Design and adaptive depth control of a micro diving agent. *IEEE Robotics and Automation Letters*, IEEE, v. 2, n. 4, p. 1871–1877, 2017. Citado na página 24.

BLIESENER, M. et al. Robotino workbook. Festo Didactic gmbh & co, Denkendorf, Germany, 2013. Citado 2 vezes nas páginas 21 e 22.

BRAITENBERG, V. *Vehicles: Experiments in Synthetic Psychology*. [S.l.]: Duke University Press, 1986. v. 95. 137–139 p. Citado na página 20.

BROOMHEAD, D. Multivariable functional interpolation and adaptive networks. *Complex Systems*, v. 2, p. 321–355, 1988. Citado na página 14.

CADENGUE, L. S. et al. Linearização por realimentação e aprendizagem por reforço para o controle do sistema de posicionamento do ROV. In: *Proceedings of SBAI 2019*. Ouro Preto: Sociedade Brasileira de Automática, 2019. Citado na página 2.

CARMONA, R. R. et al. Stable pid control for mobile robots. In: *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. [S.l.: s.n.], 2018. p. 1891–1896. Citado na página 1.

CASTILLO, O. et al. Type-2 fuzzy control for line following using line detection images. *Journal of Intelligent & Fuzzy Systems*, v. 39, p. 6089–6097, 2020. Citado 2 vezes nas páginas 1 e 2.

CUI, S.-G.; PAN, H.-L.; LI, J.-G. Application of self-tuning of pid control based on bp neural networks in the mobile robot target tracking. In: *2013 Third International Conference on Instrumentation, Measurement, Computer, Communication and Control*. [S.l.: s.n.], 2013. p. 1574–1577. Citado na página 1.

DAVIS, P. J. *Interpolation and approximation.* 1. ed. [S.l.]: Blaisdell Pub. Co., 1963. ISBN 0486624951 9780486624952. Citado na página 15.

DINH, V. T. et al. Tracking Control of Omnidirectional Mobile Platform with Disturbance Using Differential Sliding Mode Controller. *International Journal of Precision Engineering and Manufacturing*, v. 13, n. 1, p. 39–48, 2012. ISSN 20054602. Citado na página 1.

DONG, T.; HUANG, T. Neural Cryptography Based on Complex-Valued Neural Network. *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, v. 31, n. 11, p. 4999–5004, 2020. ISSN 21622388. Citado na página 13.

DOS SANTOS, J. D.; BESSA, W. M. Intelligent control for accurate position tracking of electrohydraulic actuators. *Electronics Letters*, v. 55, n. 2, p. 78–80, 2019. ISSN 00135194. Citado na página 2.

ELMAN, J. L. Finding structure in time. *Cognitive Science*, v. 14, n. 2, p. 179–211, 1990. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog1402_1>. Citado na página 18.

ESSEN, H. van; NIJMEIJER, H. Non-linear model predictive control for constrained mobile robots. In: *2001 European Control Conference (ECC)*. [S.l.: s.n.], 2001. p. 1157–1162. Citado na página 1.

FEI, J.; CHEN, Y. Dynamic terminal sliding-mode control for single-phase active power filter using new feedback recurrent neural network. *IEEE Transactions on Power Electronics*, IEEE, v. 35, n. 9, p. 9904–9922, 2020. Citado na página 2.

FEI, J.; LU, C. Adaptive Sliding Mode Control of Dynamic Systems Using Double Loop Recurrent Neural Network Structure. *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, v. 29, n. 4, p. 1275–1286, 2018. ISSN 21622388. Citado 2 vezes nas páginas 2 e 13.

FRANKLIN, G. F.; POWELL, J. D.; EMAMI-NAEINI, A. *Sistemas de controle para engenharia*. 6. ed. Porto Alegre: Bookman, 2013. ISBN 978-85-8260-067-2. Citado na página 5.

GERSHENFELD, N. A.; WEIGEND, A. S. Working Papers. *The Future of Time Series: Learning and Understanding*. [s.n.], 1993. Disponível em: <https://ideas.repec.org/p/wop/safiwp/93-08-053.html>. Citado na página 13.

HAMAGUCHI, M. Damping and Transfer Control System with Parallel Linkage Mechanism-Based Active Vibration Reducer for Omnidirectional Wheeled Robots. *IEEE/ASME Transactions on Mechatronics*, IEEE, v. 23, n. 5, p. 2424–2435, 2018. ISSN 10834435. Citado na página 1.

HAYKIN, S. *Redes neurais: princípios e prática*. 2. ed. Porto Alegre: Bookman, 2001. ISBN 978-85-7307-718-6. Citado na página 13.

HOPFIELD, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, National Academy of Sciences, v. 79, n. 8, p. 2554–2558, 1982. ISSN 00278424. Disponível em: <http://www.jstor.org/stable/12175>. Citado na página 17.

HUANG, H. C. SoPC-based parallel ACO algorithm and its application to optimal motion controller design for intelligent omnidirectional mobile robots. *IEEE Transactions on Industrial Informatics*, v. 9, n. 4, p. 1828–1835, 2013. ISSN 15513203. Citado na página 1.

JEONG, S.; CHWA, D. Sliding-Mode-Disturbance-Observer-Based Robust Tracking Control for Omnidirectional Mobile Robots with Kinematic and Dynamic Uncertainties. *IEEE/ASME Transactions on Mechatronics*, v. 26, n. 2, p. 741–752, 2021. ISSN 1941014X. Citado na página 1.

KAGAN, E. et al. Multi-robot systems and swarming. In: _____. *Autonomous Mobile Robots and Multi-Robot Systems*. John Wiley Sons, Ltd, 2019. cap. 9, p. 199–241. ISBN 9781119213154. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119213154.ch9>. Citado 2 vezes nas páginas viii e 21.

KHANH, P. H. K. et al. Trajectory tracking control of omnidirectional mobile robot using sliding mode controller. *International Conference on Control, Automation and Systems*, n. Iccas, p. 1170–1175, 2013. ISSN 15987833. Citado na página 1.

KRICHEN, N.; MASMOUDI, M. S.; DERBEL, N. Autonomous omnidirectional mobile robot navigation based on hierarchical fuzzy systems. *Engineering Computations*, v. 38, n. 2, p. 989–1023, 2021. Citado 2 vezes nas páginas 1 e 2.

KUNA, K. Time Series Prediction using Neural Networks. p. 47, 2015. Citado na página 2.

LEI, K. C.; ZHANG, X. D. An approach on discretizing time series using recurrent neural network. *Proceedings - 2018 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2018*, IEEE, p. 2522–2526, 2019. Citado na página 17.

LIMA, G. S. *Controle de Sistemas Dinâmicos Não Lineares com Compensação por Processo Gaussiano*. Dissertação (Mestrado) — UFRN, Natal, RN, 2019. Citado na página 7.

LIMA, G. S.; BESSA, W. M.; TRIMPE, S. Depth control of underwater robots using sliding modes and Gaussian process regression. *Proceedings - 15th Latin American Robotics Symposium, 6th Brazilian Robotics Symposium and 9th Workshop on Robotics in Education, LARS/SBR/WRE 2018*, IEEE, p. 13–18, 2018. Citado na página 2.

LIU, Y. et al. Omni-directional mobile robot controller based on trajectory linearization. *Robotics and Autonomous Systems*, v. 56, n. 5, p. 461–479, 2008. ISSN 09218890. Citado na página 1.

LIU, Z. et al. Cascaded feedback linearization tracking control of nonholonomic mobile robot. In: *Proceedings of the 32nd Chinese Control Conference*. [S.l.: s.n.], 2013. p. 4232–4237. Citado na página 1.

LUDWIG, S. A. Comparison of Time Series Approaches applied to Greenhouse Gas Analysis: ANFIS, RNN, and LSTM. *IEEE International Conference on Fuzzy Systems*, IEEE, v. 2019-June, 2019. ISSN 10987584. Citado 2 vezes nas páginas 2 e 17.

MANDIC, D. P.; CHAMBERS, J. A. *Recurrent Neural Networks for Prediction Wiley Series in Adaptive and Learning Systems for*. [s.n.], 2001. v. 4. 127–140 p. ISSN 13704621. ISBN 0471495174. Disponível em: <http://books.google.com/books?hl=en{&}lr=

{&}id=mMLw5pdVoIIC{&}oi=fnd{&}pg=PR15{&}dq=RECURRENT+NEURAL+NETWORKS+FOR+PREDICTION{&}ots=Tl-76UCmNb{&}sig=PxDuue>. Citado na página 16.

MATARIC, M. J. et al. *Introdução à robótica*. [S.l.]: Ed Unesp, 2014. Citado na página 20.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, v. 5, n. 4, p. 115–133, 1943. Citado na página 13.

MICCHELLI, C. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constr. Approx*, v. 2, p. 11–22, 1986. Citado na página 15.

MOREIRA, V. R. F. *Controle Inteligente de um Robô Móvel Omnidirecional com Tomada de Decisão Utilizando Aprendizagem por Reforço*. Dissertação (Mestrado) — UFRN, Natal, RN, 2021. Citado 2 vezes nas páginas 2 e 27.

NILOY, M. A. K. et al. Critical design and control issues of indoor autonomous mobile robots: A review. *IEEE Access*, v. 9, p. 35338–35370, 2021. Citado na página 1.

PANAPONGPAKORN, T.; BANJERDPONGCHAI, D. Short-Term Load Forecast for Energy Management Systems Using Time Series Analysis and Neural Network Method with Average True Range. *2019 1st International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics, ICA-SYMP 2019*, IEEE, p. 86–89, 2019. Citado na página 17.

PARK, J.; SANDBERG, I. W. Universal Approximation Using Radial-Basis-Function Networks. *Neural Computation*, v. 3, n. 2, p. 246–257, 06 1991. ISSN 0899-7667. Disponível em: <https://doi.org/10.1162/neco.1991.3.2.246>. Citado na página 16.

POWELL, M. Radial basis functions for multivariate interpolation: a review. *IMA Conference on Algorithms for the Approximation of Functions and Data, RMCS, Shrivenham, UK*, p. 143–167, 1985. Citado na página 14.

POWELL, M. Radial basis function approximation to polynomials. *Numerical Analysis 1987 Proceedings*, Dundee, p. 223–241, 1988. Citado na página 15.

RAJ, L.; CZMERK, A. Modelling and simulation of the drivetrain of an omnidirectional mobile robot. *Automatika*, v. 58, p. 232–243, 11 2017. Citado na página 23.

REZK, N. M. et al. Recurrent Neural Networks: An Embedded Computing Perspective. *IEEE Access*, v. 8, p. 57967–57996, 2020. ISSN 21693536. Citado na página 17.

RUBIO, F.; VALERO, F.; LLOPIS-ALBERT, C. A review of mobile robots: Concepts, methods, theoretical framework, and applications. *International Journal of Advanced Robotic Systems*, v. 16, n. 2, p. 1729881419839596, 2019. Citado na página 1.

SCHÄFER, A. M.; ZIMMERMANN, H. G. Recurrent neural networks are universal approximators. Springer Berlin Heidelberg, Berlin, Heidelberg, p. 632–640, 2006. Citado na página 17.

SHIEV, K. et al. Trajectory control of manipulators using type-2 fuzzy neural friction and disturbance compensator. *IS'2012 - 2012 6th IEEE International Conference Intelligent Systems, Proceedings*, IEEE, n. 1, p. 324–329, 2012. Citado 2 vezes nas páginas 1 e 2.

SHVALB, N.; HACOHEN, S. Motion in the global coordinates. In: _____. *Autonomous Mobile Robots and Multi-Robot Systems*. John Wiley & Sons, Ltd, 2019. cap. 3, p. 65–85. ISBN 9781119213154. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119213154.ch3>. Citado na página 20.

SIEGEL, B. Industrial Anomaly Detection: A Comparison of Unsupervised Neural Network Architectures. *IEEE Sensors Letters*, v. 4, n. 8, p. 6–9, 2020. ISSN 24751472. Citado na página 13.

SLOTINE, J.-J. E.; LI, W. *Applied nonlinear control*. 1. ed. [S.l.]: Prentice-Hall, 1991. ISBN 0-13-040890-5. Citado 4 vezes nas páginas 8, 9, 10 e 24.

SOLEA, R.; CERNEGA, D. Super twisting sliding mode controller applied to a nonholonomic mobile robot. In: *2015 19th International Conference on System Theory, Control and Computing (ICSTCC)*. [S.l.: s.n.], 2015. p. 87–92. Citado na página 1.

SU, X. et al. Control of balancing mobile robot on a ball with fuzzy self-adjusting pid. In: *2016 Chinese Control and Decision Conference (CCDC)*. [S.l.: s.n.], 2016. p. 5258–5262. Citado na página 1.

TANAKA, M. C.; FERNANDES, J. M.; BESSA, W. M. Feedback linearization with fuzzy compensation for uncertain nonlinear systems. *International Journal of Computers, Communications and Control*, v. 8, n. 5, p. 736–743, 2013. ISSN 18419844. Citado 2 vezes nas páginas 1 e 2.

TERAKAWA, T. et al. A Novel Omnidirectional Mobile Robot with Wheels Connected by Passive Sliding Joints. *IEEE/ASME Transactions on Mechatronics*, v. 23, n. 4, p. 1716–1727, 2018. ISSN 10834435. Citado na página 1.

TIAN, Y. Artificial Intelligence Image Recognition Method Based on Convolutional Neural Network Algorithm. *IEEE Access*, v. 8, p. 125731–125744, 2020. ISSN 21693536. Citado na página 13.

TZAFESTAS, S. G. Mobile robot control and navigation: A global overview. *Journal of Intelligent & Robotic Systems*, v. 91, n. 1, p. 35–58, 2018. Citado na página 1.

WANG, D. et al. A robust model predictive control strategy for trajectory tracking of omni-directional mobile robots. *Journal of Intelligent & Robotic Systems*, v. 98, n. 2, p. 439–453, 2020. Citado na página 1.

WANG, T. et al. Research on control system of working robot in nuclear environment based on neural network pid. In: *2020 Chinese Automation Congress (CAC)*. [S.l.: s.n.], 2020. p. 4828–4831. Citado na página 1.

WHITE, B. W.; ROSENBLATT, F. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. *The American Journal of Psychology*, v. 76, n. 4, p. 705, 1963. ISSN 00029556. Citado na página 13.

WIENER, N. *Cybernetics: or Control and Communication in the Animal and the Machine*. 2. ed. Cambridge, MA: MIT Press, 1948. Citado na página 20.

XU, D. et al. Trajectory tracking control of omnidirectional wheeled mobile manipulators: Robust neural network-based sliding mode approach. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, v. 39, n. 3, p. 788–799, 2009. ISSN 10834419. Citado na página 2.

YANG, Y. C.; CHENG, C. C. Robust adaptive trajectory control for an omnidirectional vehicle with parametric uncertainty. *Transactions of the Canadian Society for Mechanical Engineering*, v. 37, n. 3, p. 405–413, 2013. ISSN 03158977. Citado na página 1.

YANMIN, L.; LIMIN, D.; ZHIBIN, F. Motion control of robot based on fuzzy adaptive pid algorithm. In: *Proceedings of 2013 3rd International Conference on Computer Science and Network Technology*. [S.l.: s.n.], 2013. p. 1310–1313. Citado na página 1.

YAO, W.; DATCU, M. Deep neural networks based semantic segmentation for optical time series. *International Geoscience and Remote Sensing Symposium (IGARSS)*, v. 2018-July, p. 4403–4406, 2018. Citado 2 vezes nas páginas 2 e 17.