

**INSTITUTO FEDERAL DO ESPÍRITO SANTO  
CURSO DE ENGENHARIA ELÉTRICA**

**CHRISTOFER GALDINO BERNARDO**

**CONTROLE ANTIBALANÇO DA CARGA PARA UMA PONTE ROLANTE  
DIDÁTICA ACIONADA COM USO DE COMANDOS DE VOZ**

Vitória  
2019

CHRISTOFER GALDINO BERNARDO

**CONTROLE ANTIBALANÇO DA CARGA PARA UMA PONTE ROLANTE  
DIDÁTICA ACIONADA COM USO DE COMANDOS DE VOZ**

Trabalho de Conclusão de Curso apresentado à  
Coordenadoria do Curso de Engenharia Elétrica do  
Instituto Federal do Espírito Santo como requisito parcial  
para obtenção do título de Graduação em Engenharia  
Elétrica.

Orientador: Prof. Dr. Luis Eduardo Martins de Lima

Vitória  
2019


**CHRISTOFER GALDINO BERNARDO**

**CONTROLE ANTIBALANÇO DA CARGA PARA UMA PONTE ROLANTE  
DIDÁTICA ACIONADA COM USO DE COMANDOS DE VOZ**


Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo, como requisito parcial para a obtenção do título de Engenheiro Eletricista.

Aprovado em 26 de Junho de 2019

**COMISSÃO EXAMINADORA**

  
**Prof. Dr. Luis Eduardo Martins de Lima**  
Instituto Federal do Espírito Santo  
Orientador

  
**Profa. Dra. Mariana Rampinelli Fernandes**  
Instituto Federal do Espírito Santo  
Examinadora

  
**Prof. Dr. Leandro Bueno**  
Instituto Federal do Espírito Santo  
Examinador

## DECLARAÇÃO DO AUTOR

Declaro, para fins de pesquisa acadêmica, didática e técnico-científica, que este Trabalho de Conclusão de Curso pode ser parcialmente utilizado, desde que se faça referência à fonte e ao autor.

Vitória, 26 de junho de 2019.

  
Christofer Galdino Bernardo

*Para Jesus Cristo, que é minha vida.*

*Para Adenilde e Henrique, que me trouxeram à vida e proporcionaram todo o sustento e amor.*

*Para Tayla, por todo o seu apoio e carinho.*

*Para todos os amigos e professores, que me acompanharam nessa caminhada.*

## **AGRADECIMENTOS**

Agradeço primeiramente a Deus pela graça, salvação, fôlego de vida e força necessária para concluir esse tão sonhado curso.

Agradeço minha família e amigos por sempre ter me apoiado e ajudado, suprimo-me em todo o amor, cuidado e compreensão.

Agradeço ao meu orientador Prof. Dr. Luis Eduardo Martins de Lima pelas orientações tão preciosas e por toda a atenção durante o desenvolvimento desse trabalho.

Agradeço aos meus professores de curso que contribuíram não só na minha formação acadêmica e profissional, como também na minha formação como pessoa.

Agradeço aos meus colegas de curso que foram grandes incentivadores e ajudadores nessa caminhada, proporcionando momentos de refrigério.

Agradeço ao Ifes por todo o suporte e estrutura oferecidos em todos esses anos.

Muito obrigado!

*“Quanto mais estudo a natureza, mais fico  
maravilhado com o Criador.”*

*(Louis Pasteur)*

## RESUMO

O emprego de sistemas automatizados tem se tornado uma necessidade crescente em diversos setores e aplicações como, por exemplo, a movimentação de cargas e materiais entre os setores fabris com o uso de pontes rolantes. A implementação de controle automático para um processo desse tipo pode não apenas melhorar o desempenho, como também eliminar riscos de acidentes com operadores. Com isso, esse trabalho de graduação apresenta uma proposta de controle antibalanço aplicado a uma ponte rolante didática, sendo possível o seu acionamento por comandos de voz através de uma Interface Gráfica para o Usuário. Optou-se pela aplicação de controle inteligente *fuzzy* para o antibalanço da carga e controle clássico Proporcional-Derivativo, PD, com sintonia Ziegler-Nichols para os movimentos laterais do carro. Todo o controle é embarcado em um microcontrolador ATmega 2560, responsável por comutar entre as duas formas de controle, bem como realizar a comunicação com a interface. Além disso, utilizou-se coeficientes mel-cepstrais para a caracterização dos sinais de voz e a Rede Neural Artificial, RNA, *perceptron* multicamadas para a classificação dos coeficientes extraídos. Após a realização de testes de movimentação do carro e análise das oscilações da carga, constatou-se que o controlador *fuzzy* para o movimento lateral aliado à sintonia Ziegler-Nichols para o controlador antibalanço PD e ao reconhecimento de comandos de voz por RNA mostraram-se eficazes para a aplicação em estudo.

Palavras-chave: Controle antibalanço. Controle *fuzzy*. Comando de voz. Coeficientes mel-cepstrais. Rede Neural Artificial.



## **ABSTRACT**

The usage of automated systems has become a growing necessity in several sectors and applications as, for example, the movement of loads and materials between factories sectors with the use of overhead cranes. The implementation of automatic control for a process of this kind can not only improve the performance, but also eliminate the risks of accidents with the operators. Thus, this graduation project presents a proposal of anti-balance control applied to a didactic overhead crane, with the possibility of its activation by voice commands throw a Graphical User Interface. It was opted for the application of fuzzy intelligent control for the load anti-balance and Proportional-Derivative, PD, classic control with Ziegler-Nichols tuning for the car sidelong movement. The whole control is embedded in an ATmega 2560 microcontroller, responsible for commuting between the two forms of control, as well as accomplish the communication with the interface. Moreover, it was used the Mel-Frequency Cepstral Coefficients for the characterization of the voice signals and a Multilayer Perceptron artificial neural network for the classification of the extracted coefficients. After the accomplishment of the car movement tests and load balance analysis, it was verified that the fuzzy controller for the sidelong movement allied to the Ziegler-Nichols tuning for the anti-balance PD control and the speech commands recognition by artificial neural network were effective for the studied application.

**Keywords:** Anti-balance control. Fuzzy control. Voice commands. Mel-frequency cepstral coefficients. Artificial neural networks.

## LISTA DE FIGURAS

Figura 1 – Exemplo de ponte rolante em operação.....	18
Figura 2 – Protótipo de ponte rolante utilizado no trabalho. ....	20
Figura 3 – Diagrama de corpo livre de um pêndulo simples.....	22
Figura 4 – Exemplo de uma GUI. ....	24
Figura 5 – ATmega 2560.....	25
Figura 6 – Resposta de um sistema para um degrau.....	26
Figura 7 – Diagrama de blocos de um sistema de controle convencional.....	26
Figura 8 – Curva de resposta em "S". ....	28
Figura 9 – Sinal de saída com oscilação mantida de período $P_{cr}$ . ....	29
Figura 10 – Funções de pertinência de um conjunto <i>fuzzy</i> . ....	30
Figura 11 – Diagrama de blocos de um controlador <i>fuzzy</i> típico.....	31
Figura 12 – Esquema de sistema genérico de reconhecimento de fala. ....	32
Figura 13 – Gráfico de frequências Mel por frequências lineares. ....	36
Figura 14 – Exemplo de banco de filtros mel. ....	37
Figura 15 – <i>Perceptron</i> de camada única. ....	40
Figura 16 – <i>Perceptron</i> multicamadas.....	41
Figura 17 - Estrutura da GUI. ....	43
Figura 18 – Fluxograma do modo de comando manual da GUI.....	45
Figura 19 – Fluxograma do funcionamento do modo temporizado. ....	46
Figura 20 – Interface para o modo de comando por voz. ....	47
Figura 21 – Indicadores de cores na estrutura da ponte rolante para os comandos por voz.....	47
Figura 22 – Fluxograma do modo de comando por voz da GUI.....	48
Figura 23 – Respostas (a) elétrica e (b) em frequência da carga da ponte rolante...49	
Figura 24 – Estruturação e exemplo do pacote de comunicação de dados. ....	50
Figura 25 – Conjuntos <i>fuzzy</i> de (a) entrada e (b) saída para a configuração com três funções de pertinência. ....	55
Figura 26 – Conjuntos <i>fuzzy</i> de (a) entrada e (b) saída para a configuração com cinco funções de pertinência. ....	57
Figura 27 – Conjuntos <i>fuzzy</i> de (a) entrada e (b) saída para a configuração com sete funções de pertinência. ....	58

Figura 28 – Comparação dos conjuntos <i>fuzzy</i> de entrada e saída para controladores com (a) ganho proporcional unitário e (b) ganho proporcional 2. ....	60
Figura 29 – Sistema malha fechada para ensaio de sintonia pelo segundo método de Ziegler-Nichols. ....	62
Figura 30 – Oscilações mantidas com $f_{cr}$ de 4,95 Hz em ensaio de sintonia Ziegler-Nichols. ....	63
Figura 31 – Comparação entre as simulações da resposta ao degrau do sistema para (a) ganhos encontrados neste trabalho e (b) ganhos ajustados encontrados por Baiôco (2012) em seu trabalho. ....	64
Figura 32 – Fluxograma do processo de controle embarcado ao microcontrolador. ....	68
Figura 33 – Fluxograma do processo de reconhecimento de voz utilizado. ....	69
Figura 34 – Exemplo de MFCCs para o sinal de voz correspondente à palavra "amarelo". ....	72
Figura 35 – MFCCs para todos as 100 diferentes gravações correspondentes à palavra "amarelo". ....	72
Figura 36 – Comparação de MFCCs para sinais de voz correspondentes às palavras (a) "amarelo" e (b) "verde". ....	73
Figura 37 – Comparação de MFCCs para sinais de voz correspondentes às palavras (a) "vermelho" e (b) "azul". ....	73
Figura 38 – Comparação das formas de onda da carga para os testes nos percursos de (a) ida e (b) volta da ponte para controlador com 3 conjuntos <i>fuzzy</i> e ganho 5. ....	78
Figura 39 – Forma de onda da carga para percurso de ida e controlador com 3 conjuntos <i>fuzzy</i> e ganho 20, permitindo observar sobressinal máximo na parada. ....	79
Figura 40 – Comparação das respostas da carga para movimento lateral do carro (a) sem controle antibalanço e (b) com controle antibalanço da carga. ....	82
Figura 41 – Gráfico da atuação do controlador (em verde) de acordo com a movimentação da carga (em amarelo). ....	83
Figura 42 – Comparação de MFCCs para sinais de voz correspondentes às palavras (a) "vermelho" e (b) "verde". ....	86

## LISTA DE QUADROS

Quadro 1 – Base de regras para controlador com três conjuntos <i>fuzzy</i> .....	56
Quadro 2 – Base de regras para controlador com cinco conjuntos <i>fuzzy</i> . ....	57
Quadro 3 – Base de regras para controlador com sete conjuntos <i>fuzzy</i> . ....	59

## LISTA DE TABELAS

Tabela 1 – Valores das constantes na regra de sintonia de Ziegler-Nichols para o primeiro método. ....	28
Tabela 2 – Valores das constantes na regra de sintonia de Ziegler-Nichols para o segundo método.....	29
Tabela 3 – Frequências centrais para banco de filtros mel. ....	38
Tabela 4 – Posições para o movimento no modo de comando de voz de acordo com as respectivas cores.....	47
Tabela 5 – Relação entre leitura dos ângulos e valores de tensão da carga. ....	65
Tabela 6 – Dados comparativos de controladores <i>fuzzy</i> com 3, 5 e 7 conjuntos para excursão de ida. ....	76
Tabela 7 – Dados comparativos de controladores <i>fuzzy</i> com 3, 5 e 7 conjuntos para excursão de volta. ....	77
Tabela 8 – Resultados dos três testes de desempenho para as diferentes topologias de redes neurais treinadas com 100 amostras de áudio por cor.....	85
Tabela 9 – Resultados dos três testes de desempenho para as diferentes topologias de redes neurais treinadas com 50 amostras de áudio por cor.....	88
Tabela 10 – Resultados dos três testes de desempenho para as diferentes topologias de redes neurais treinadas com 20 amostras de áudio por cor.....	88
Tabela 11 – Comparação dos resultados dos testes de desempenho da rede de topologia $2N3 + M$ quando treinada com 100, 50 e 20 amostras de treinamento.....	89

## LISTA DE ABREVIACÕES, SIGLAS E SÍMBOLOS

$\theta$  – Ângulo entre o fio do pêndulo e a vertical

$\mu$  – Grau de pertinência

$\pi$  – Constante matemática *pi*

$\omega$  – Frequência angular

A/D – Analógico/Digital

DCT – *Discrete Cosine Transform*

eFLL – *embedded Fuzzy Logic Library*

$f_{cr}$  – Frequência crítica de oscilação

FFT – *Fast Fourier Transform*

$f_M$  – Frequência máxima do sinal

FP – Função de pertinência

g – Aceleração da gravidade

GUI – *Graphical User Interface*

HMM – *Hidden Markov Model*

IHM – Interface Homem-Máquina

$K_{cr}$  – Ganho proporcional crítico

$K_d$  – Ganho derivativo

$K_p$  – Ganho proporcional

L – Comprimento do fio em um pêndulo

LD – Lento-Direita

LE – Lento-Esquerda

LN – Longe-Negativo

LP – Longe-Positivo

LSF – *Line Spectral Frequencies*

MFCC – *Mel-Frequency Cepstral Coefficients*

MLN – Muito Longe-Negativo

MLP – Muito Longe-Positivo

MP – Muito Perto

MRD – Muito Rápido-Direita

MRE – Muito Rápido-Esquerda

P – Perto

$P_{cr}$  – Período crítico de oscilação

PD – Proporcional-Derivativo

PID – Proporcional-Integral-Derivativo

PMC – *Perceptron* multicamadas

PN – Perto-Negativo

PP – Perto-Positivo

PWM – *Pulse Width Modulation*

RAV – Reconhecimento Automático de Voz

RD – Rápido-Direita

RE – Rápido-Esquerda

RNA – Rede Neural Artificial

T – Período

$T_d$  – Tempo derivativo

Z – Zero

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO .....</b>	<b>18</b>
1.1	OBJETIVOS .....	20
1.2	ESTRUTURA DO TRABALHO .....	20
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA .....</b>	<b>22</b>
2.1	MOVIMENTO PENDULAR .....	22
2.2	INTERFACE GRÁFICA DO USUÁRIO .....	23
2.3	ATMEGA 2560 .....	24
2.4	CONTROLE HÍBRIDO .....	25
<b>2.4.1</b>	<b>Controle clássico .....</b>	<b>26</b>
<b>2.4.2</b>	<b>Sintonia de controladores PID .....</b>	<b>27</b>
2.4.2.1	Primeiro método .....	27
2.4.2.2	Segundo método .....	29
<b>2.4.3</b>	<b>Lógica e controle <i>fuzzy</i> .....</b>	<b>30</b>
2.4.3.1	Ganho proporcional <i>fuzzy</i> .....	32
2.5	COMANDOS POR VOZ .....	32
<b>2.5.1</b>	<b>Pré-processamento .....</b>	<b>33</b>
2.5.1.1	Conversão A/D e pré-ênfase .....	33
2.5.1.2	Análise espectral .....	34
2.5.1.3	Extração de características .....	35
<b>2.5.2</b>	<b>Classificação .....</b>	<b>38</b>
2.6	REDE NEURAL ARTIFICIAL .....	39
<b>2.6.1</b>	<b><i>Perceptron</i> multicamadas .....</b>	<b>39</b>
<b>3</b>	<b>DESENVOLVIMENTO .....</b>	<b>42</b>
3.1	DESENVOLVIMENTO DA INTERFACE GRÁFICA PARA O USUÁRIO .....	42
<b>3.1.1</b>	<b>Comando manual .....</b>	<b>44</b>



3.1.2	Comando por voz .....	46
3.1.3	Comunicação dos dados .....	48
3.2	PROJETO DE CONTROLE HÍBRIDO .....	51
3.2.1	Zona morta do motor .....	51
3.2.2	Controle <i>fuzzy</i> do movimento lateral.....	52
3.2.2.1	Configuração do controlador <i>fuzzy</i> .....	53
3.2.2.2	Montagem das funções de pertinência.....	53
3.2.2.3	Ganho proporcional .....	59
3.2.3	Controle PD antibalanco da carga .....	61
3.2.3.1	Aplicação do segundo método de sintonia Ziegler-Nichols .....	61
3.2.3.2	Comparação com abordagem por modelo matemático.....	63
3.2.3.3	Condicionamento da leitura do potenciômetro da carga .....	65
3.2.4	Integração dos controles no microcontrolador.....	67
3.3	APLICAÇÃO DE UM MÉTODO DE ACIONAMENTO A PARTIR DE COMANDOS DE VOZ.....	67
3.3.1	Aquisição do sinal de voz e pré-processamento .....	69
3.3.2	Análise espectral.....	70
3.3.3	Extração de características.....	70
3.3.4	Classificação por Rede Neural Artificial .....	73
4	RESULTADOS .....	75
4.1	CONFIGURAÇÃO DO CONTROLADOR <i>FUZZY</i> .....	75
4.2	GANHOS IDEAIS PARA O CONTROLADOR ANTIBALANÇO.....	81
4.3	DESEMPENHO DE DIFERENTES TOPOLOGIAS DE REDES NEURAIS....	85
4.4	TESTES COM O SISTEMA INTEGRADO.....	89
5	CONCLUSÃO .....	91
5.1	PERSPECTIVA PARA TRABALHOS FUTUROS.....	91
	REFERÊNCIAS.....	93

## 1 INTRODUÇÃO

Indústrias de médio a grande porte geralmente possuem a necessidade de realizar o movimento de cargas e materiais entre os setores fabris. Em muitos casos, essas cargas são volumosas e pesadas, chegando a pesar dezenas ou até centenas de toneladas. Com isso, para que o transporte da carga seja possível, é comum o uso de pontes rolantes.

Uma ponte rolante é um tipo de guindaste cuja principal função é o transporte horizontal da carga, no qual a carga é içada sendo suspensa da superfície de apoio em geral por cabos de aço, e deslocada por um carro que se movimenta ao longo do eixo transversal da ponte. Esses instrumentos são amplamente utilizados em grandes indústrias do ramo siderúrgico e em portos. Na Figura 1, é mostrada uma foto de uma ponte rolante sendo utilizada no transporte de uma carga em região portuária.

Figura 1 – Exemplo de ponte rolante em operação.



Fonte: Guincho 24 horas (2019).

Considerando sua aplicação, cargas de diversos tipos são movimentadas com o auxílio da ponte rolante. Muitas dessas cargas são valiosíssimas, e é imprescindível que o seu traslado seja feito de forma precisa e com o mínimo de oscilação para que a carga não sofra nenhum dano, nem ofereça risco de acidentes em função de uma eventual queda. Todavia, essa não é uma tarefa fácil: na maioria dos casos, o controle

da movimentação da carga por meio de uma ponte rolante é realizado de forma não-automática com a ação de um operador presente no chão de fábrica, e de acordo com sua experiência.

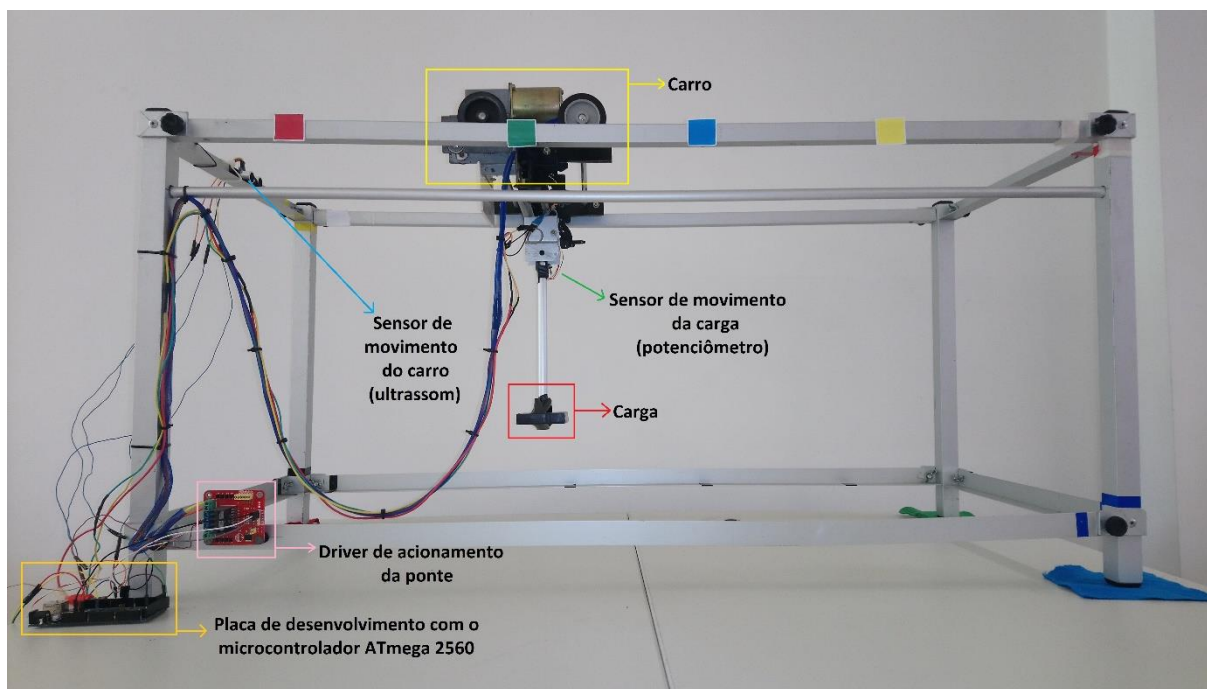
Caso o operador não seja experiente ou cometa erros, tema abordado por Mandal et. al (2015), possivelmente a oscilação da carga não será reduzida, podendo até mesmo ser aumentada. Isso proporciona riscos à carga, podendo danificá-la e causando grandes prejuízos à indústria, assim como riscos de acidentes ao operador, expondo-o em função da quebra do equipamento ou queda da carga, já que este operador está presente no chão de fábrica a fim de controlar a ponte rolante. Além disso, outros fatores imprevisíveis podem aumentar a oscilação da carga, como a presença de ventos laterais na ponte (no caso de um sistema externo) e deficiências de *design* e construção das pontes, como bem observaram Marquez, Venturino e Otegui (2014). Essas dificuldades poderiam ser contornadas por meio de um sistema de controle automático antibalço da ponte rolante, permitindo que esta seja comandada remotamente pelo operador, afastando-o dos riscos de acidentes no chão de fábrica. Algumas contribuições para o assunto já foram desenvolvidas, como é o caso do estudo realizado por Baiôco (2012), construindo um protótipo de ponte rolante (o mesmo utilizado neste trabalho) e controlando o sistema antibalço com uso de controle clássico e emprego do algoritmo Proporcional-Integral-Derivativo, PID. Além disso, é interessante ressaltar as contribuições dadas por Assis (2015), que desenvolveu um sistema inteiramente baseado em lógica *fuzzy* a fim de controlar os movimentos da ponte rolante, como também os estudos de Yu, Moreno-Armendariz e Rodriguez (2011), integrando a funcionalidade do algoritmo PID à compensação neural adaptativa.

Portanto, para este trabalho, uma possível solução para a eliminação automática do balanço de uma carga transportada por uma ponte rolante é o emprego de controle híbrido integrando controle inteligente por lógica *fuzzy*, para o controle do movimento de deslocamento, e o controle clássico, com uso do algoritmo Proporcional-Derivativo, PD, para eliminação do balanço, com disponibilidade de uma Interface Homem-Máquina, IHM, e comandos de voz para indicar ao sistema qual a posição final desejada para a carga.

Para a realização deste projeto, um protótipo de ponte rolante é utilizado. Esse protótipo é composto de um pequeno carro que se movimenta sobre o eixo de uma maquete construída em alumínio, além de uma carga içada por meio de uma pequena

haste de alumínio conectada ao carro. Na Figura 2, pode-se ver uma foto do protótipo que será utilizado, com seus respectivos componentes devidamente indicados.

Figura 2 – Protótipo de ponte rolante utilizado no trabalho.



Fonte: Elaborado pelo autor (2019).

## 1.1 OBJETIVOS

Este trabalho tem como objetivo geral o desenvolvimento de um sistema para controle automático antibalanço da carga e acionamento por comandos de voz de uma ponte rolante didática. Os objetivos específicos desse trabalho consistem em:

- criar uma interface gráfica para o usuário operador da ponte;
- desenvolver o projeto de controle híbrido antibalanço para o protótipo;
- estudar e implementar técnicas de acionamento de sistemas por comando de voz;

## 1.2 ESTRUTURA DO TRABALHO

Esse trabalho apresenta cinco capítulos, sendo estruturados da seguinte forma:

o capítulo 1 apresenta uma introdução ao tema abordado e seus conceitos, bem como alguns estudos presentes na literatura acerca do mesmo assunto.

O capítulo 2 apresenta uma fundamentação teórica de todos os conhecimentos necessários para a elaboração desse trabalho.

O capítulo 3 apresenta o desenvolvimento metodológico do trabalho desenvolvido baseado nos conhecimentos adquiridos anteriormente, isto é, desenvolvimento da interface gráfica, projeto dos controladores e aplicação de um método de acionamento a partir de comando de voz.

O capítulo 4 apresenta a análise e comparação dos resultados encontrados nos testes executados sobre o protótipo e suas implicações para o trabalho.

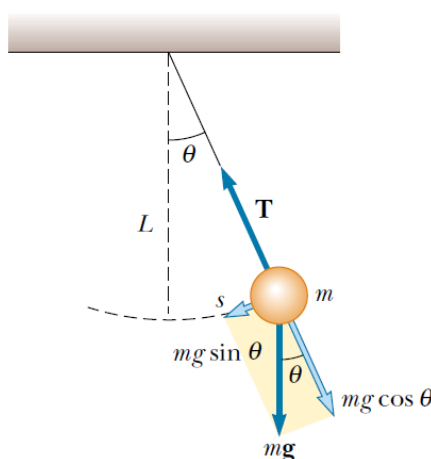
O capítulo 5 apresenta as conclusões e dificuldades do estudo feito, bem como sugestões para projetos futuros.

## 2 FUNDAMENTAÇÃO TEÓRICA

### 2.1 MOVIMENTO PENDULAR

O movimento de uma carga em um protótipo de ponte rolante, como o que será analisado neste trabalho, pode ser comparado ao movimento de um pêndulo simples, dado que a carga possui apenas um grau de liberdade, isto é, seu movimento é limitado ao balanço pendular na direção longitudinal da ponte. Pode-se observar na Figura 3 um diagrama de corpo livre do pêndulo simples e todas as forças envolvidas no seu movimento, em **negrito>**.

Figura 3 – Diagrama de corpo livre de um pêndulo simples.



Fonte: Halliday (2010, p. 402).

O pêndulo simples é composto por uma partícula de massa  $m$  suspensa por um fio de comprimento  $L$  e massa desprezível. Considerando as distribuições de forças indicadas anteriormente e aplicando a segunda lei de Newton ao eixo de movimento tangencial da partícula, segundo Halliday, Resnick e Walker (2010) tem-se que o período da movimentação de um pêndulo simples pode ser descrito por

$$T = \frac{2\pi}{\omega} = 2\pi \sqrt{\frac{L}{g}} \quad (1).$$

Tal equação, para o período do movimento de um pêndulo simples, somente pode ser considerada caso o ângulo  $\theta$  seja pequeno, isto é, menor que  $10^\circ$ , aproximadamente.

Quando isso ocorre, considerando a expansão por série de Taylor da função seno, isto é,

$$\text{sen } \theta = \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} + \dots + (-1)^{n+1} \cdot \frac{\theta^{2n-1}}{(2n-1)!} \quad (2),$$

é possível realizar a aproximação  $\text{sen } \theta \approx \theta$ . De acordo com Medina, Velazco e Salinas (2005), em aspectos práticos, é possível considerar uma angulação de até 23° como pequena, sendo possível realizar a aproximação citada anteriormente e, portanto, a utilização da equação 1.

Neste trabalho, o objetivo principal é precisamente eliminar qualquer movimento pendular da carga durante a translação da ponte rolante. Os parâmetros do controle da ponte rolante, bem como a supervisão de todo o processo de movimentação da mesma estarão disponíveis ao usuário por meio de uma interface gráfica, que será apresentada no próximo item.

## 2.2 INTERFACE GRÁFICA DO USUÁRIO

Segundo a *Encyclopædia Britannica* (2010, p.1, tradução nossa), uma Interface Gráfica para o Usuário, GUI (*Graphical User Interface*) pode ser definida como “[...] um programa de computador que permite que uma pessoa se comunique com um computador por meio de símbolos, metáforas visuais e dispositivos apontadores”. GUIs são utilizadas em praticamente todas as aplicações computacionais, além de serem importantes ferramentas em processos industriais de automação e controle, tecnologia clínica, engenharias, entre outros. Na Figura 4, pode-se ver um exemplo de uma GUI desenvolvida no *software* MatLab® R2012b (MATHWORKS, 2012).

É importante que a GUI seja bem construída de forma a garantir ao usuário uma interface clara, objetiva e intuitiva. Isto porque o desenho e construção de uma interface podem interferir diretamente na decisão do usuário, como é bem demonstrando por Stupak et al. (2010).

Neste trabalho, a GUI será o meio pelo qual o usuário poderá enviar comandos ao controlador embarcado em um microcontrolador, o ATmega 2560, que será visto a seguir.

Figura 4 – Exemplo de uma GUI.

**Rede**

Quantidade de redes a serem geradas:

Tensão de linha da rede (V):

Comprimento total da rede (km): Min  Max

Número de barras de passagem: Min  Max

Proporção dos cabos disponíveis:

Cabos	Proporção (%)
Cabo A	25
Cabo B	25
Cabo C	25
Cabo D	25

**Carga**

Proporção de cargas nas barras da rede (%):

Demanda das cargas na rede (kVA): Min  Max

Fator de potência das cargas na rede:  ±  %

Proporção de unidades de GD nas barras da rede (%):

Demanda das GDs na rede (kVA): Min  Max

Fator de potência das GDs na rede:  ±  %

Proporção do tipo de carga:

Tipo	Proporção (%)
Comercial	50
Residencial	15
Industrial	25
Rural	2
Municipal e/o...	3
Iluminação p...	5

**GERAR REDES**

Fonte: Elaborado pelo autor (2017).

### 2.3 ATMEGA 2560

ATmega é um controlador CMOS 8 bits de baixa potência baseado em uma arquitetura RISC aperfeiçoada, a AVR. Existem diversos controladores ATmega, sendo o ATmega 2560 o utilizado neste trabalho. Dentre os seus 54 pinos digitais, 16 podem ser usados como saídas de modulação por largura de pulso, PWM (*Pulse Width Modulation*), dos quais 4 possuem resolução fixa de 8 bits (255 valores diferentes) e os outros 12 possuem resolução programável de 2 a 16 bits. Além disso, o ATmega 2560 também possui 16 pinos analógicos com 10 bits de resolução (1024 valores diferentes), 4 UARTs, frequência de oscilação interna (cristal) de 16MHz, 256KB de memória flash, 8 KB de memória RAM e interface para comunicação serial (MICROCHIP, 2014). Um exemplo de microcontrolador ATmega 2560 pode ser visto na Figura 5.



Figura 5 – ATmega 2560.



Fonte: Microchip (2014).

O microcontrolador aplicado neste trabalho é utilizado com a placa de desenvolvimento Arduino MEGA 2560 e programado via interface de desenvolvimento do Arduino.

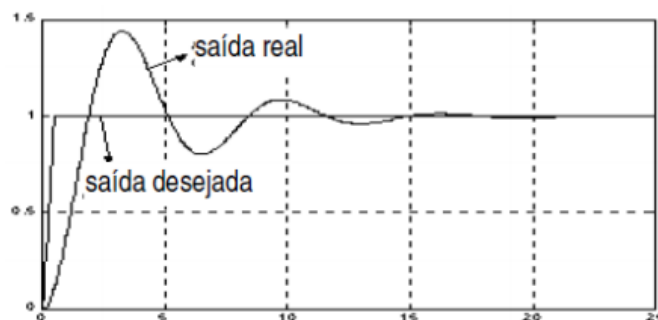
As características do tipo de controle e do controlador que será embarcado no ATmega 2560 serão vistas na próxima seção.

## 2.4 CONTROLE HÍBRIDO

Muitos sistemas dinâmicos apresentam um comportamento em resposta a um determinado sinal (tensão, pressão, temperatura, etc.) que os torna inviáveis em algum tipo de aplicação, como em processos industriais, por exemplo. O controle de um sistema dinâmico tem geralmente como objetivo reduzir ou, idealmente, anular o erro (diferença entre entrada e saída), levando o sistema a um estado controlado de saída, conforme desejado pelo projetista (CHOQUEHUANCA, 2010). Pode-se observar na Figura 6 uma resposta típica de um sistema para um degrau: o sistema apresenta grande oscilação no regime transitório até que, enfim, alcança o valor do sinal da referência no regime permanente.

Existem muitas estratégias de controle presentes na literatura e que são efetivamente utilizadas na indústria. Todavia, a estratégia de controle que será utilizada neste trabalho é a estratégia de controle híbrido.

Figura 6 – Resposta de um sistema para um degrau.



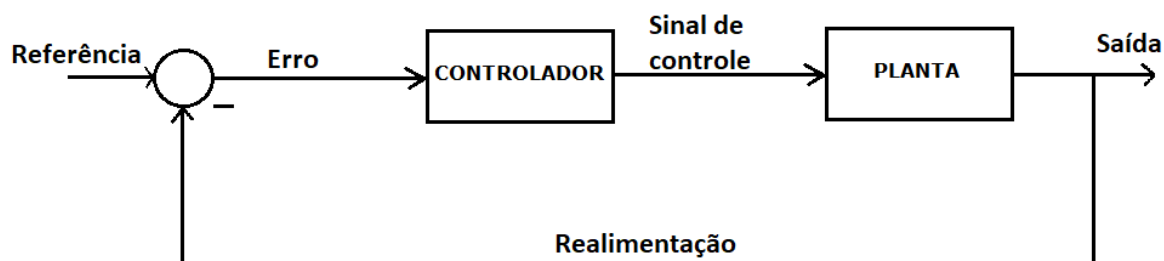
Fonte: Choquehuanca (2010, p. 40).

Por controle híbrido, entende-se como a junção de duas técnicas de controle para a solução de controle de um único sistema, exato intuito deste trabalho. O controle híbrido aplicado neste trabalho tem como objetivo unir as estratégias de controle clássico com as estratégias de controle *fuzzy*. Para tal, será apresentado o embasamento teórico de tais estratégias nas seções que seguem.

#### 2.4.1 Controle clássico

O controle clássico é uma estratégia de controle muito eficaz para sistemas cuja modelagem dinâmica é acessível ou de simples obtenção, e em geral, são modelos linearizáveis. Na Figura 7, pode-se observar um diagrama de blocos simplificado típico de um sistema de controle convencional.

Figura 7 – Diagrama de blocos de um sistema de controle convencional.



Fonte: Elaborado pelo autor (2017).

Um dos tipos de controladores clássicos mais utilizados é o controlador Proporcional-Integral-Derivativo, PID, e suas variações. Os controladores PID são amplamente utilizados na indústria em função de sua confiabilidade, além do fato de existirem

diversas arquiteturas possíveis e métodos de sintonia, como o bem conhecido método de Ziegler-Nichols (YEROGLU; TAN, 2011).

Uma das variações do controlador PID é o controlador Proporcional-Derivativo, PD. A ação de tal controlador pode ser definida pela equação e função de transferência:

$$u(t) = K_p e(t) + K_p T_d \frac{de(t)}{dt} \quad (3)$$

$$\frac{U(s)}{E(s)} = K_p (1 + T_d s) \quad (4),$$

onde  $K_p$  é o ganho proporcional e  $T_d$  é uma constante denominada tempo derivativo. O produto entre os termos  $K_p$  e  $T_d$  na Equação 3 pode ser simplificado por uma outra constante,  $K_d$ , definido como o ganho derivativo. O controlador PD tem sua saída proporcional à taxa de variação do sinal de erro existente; ou seja, quanto maior a velocidade de mudança do sinal de erro, maior será a magnitude do sinal de controle (OGATA, 2001).

#### 2.4.2 Sintonia de controladores PID

Para que as ações de controle estabeleçam a dinâmica desejada no comportamento do sistema, é necessária uma adequada sintonia dos ganhos do controlador. Existem diversas regras para a sintonia de controladores PID; uma das mais utilizadas foi apresentada por Ziegler e Nichols (1942), sendo essa a regra de sintonia utilizada neste trabalho.

A regra Ziegler-Nichols determina valores de  $K_p$ ,  $T_i$  (tempo de integral) e  $T_d$  com base nas características da resposta transitória do sistema. Existem dois métodos de sintonia Ziegler-Nichols que poderão ser aplicados de acordo com a característica transitória obtida (OGATA, 2001).

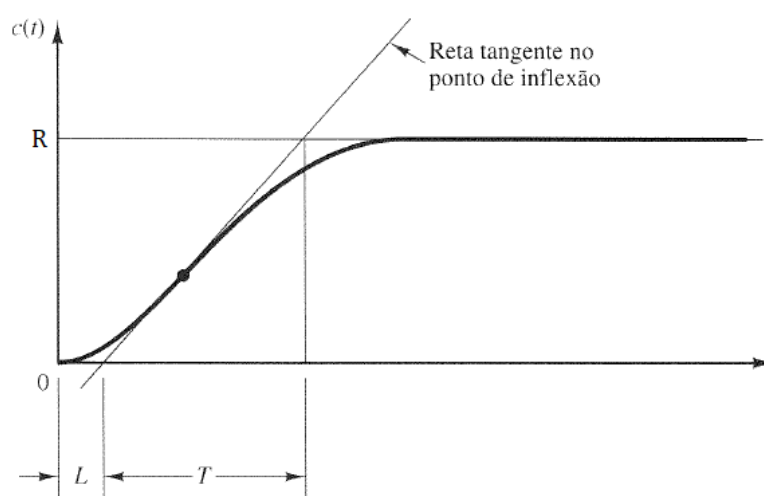
##### 2.4.2.1 Primeiro método

O primeiro método aplica-se apenas para os processos que não envolvem integradores nem polos dominantes complexos-conjugados. Para saber se um sistema apresenta essas características, basta obter a curva de resposta do sistema

a uma excitação em degrau unitário; se a curva obtida apresentar um formato em “S” (Figura 8), isso significa que o processo não apresenta integradores nem polos dominantes complexos-conjugados, ou seja, esse método é aplicável.

Através da obtenção da curva de resposta, ao se traçar uma reta tangente ao ponto de inflexão e determinar-se os pontos de intercessão com o eixo do tempo e com a reta  $c(t) = R$ , é possível obter dois importantes parâmetros da resposta do sistema, como visto na Figura 8: o tempo de retardo  $L$  e a constante de tempo  $T$ .

Figura 8 – Curva de resposta em “S”.



Fonte: Adaptado de Ogata (2001, p. 545)

Com os valores de  $L$  e  $T$ , é possível aplicá-los na obtenção dos valores de  $K_p$ ,  $T_i$  e  $T_d$  de acordo com a fórmula sugerida por Ziegler e Nichols na Tabela 1.

Tabela 1 – Valores das constantes na regra de sintonia de Ziegler-Nichols para o primeiro método.

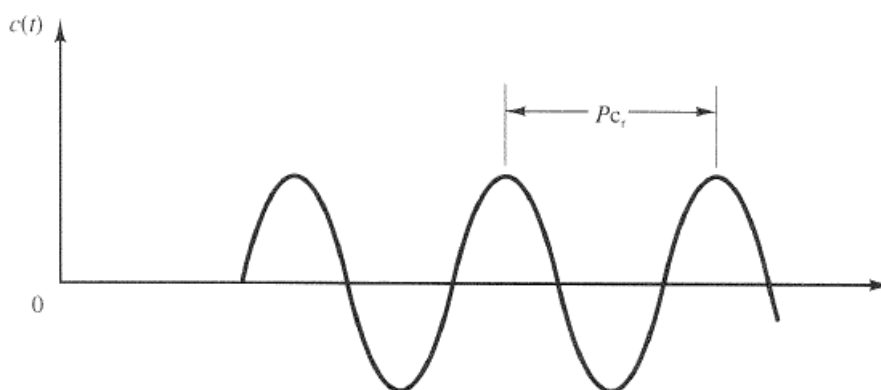
Tipo de controlador	$K_p$	$T_i$	$T_d$
P	$\frac{T}{L}$	$\infty$	0
PI	$0,9 \frac{T}{L}$	$\frac{L}{0,3}$	0
PID	$1,2 \frac{T}{L}$	$2L$	$0,5L$

Fonte: Ogata (2001, p. 546)

### 2.4.2.2 Segundo método

Diferentemente do primeiro método, o segundo método pode ser aplicado a processos que envolvem integradores e/ou polos dominantes complexos-conjugados. Para aplicar esse método, inicialmente anula-se a ação integral e derivativa do controlador sobre o sistema, permitindo apenas a ação proporcional. Feito isso, o ganho proporcional é aumentado até um valor crítico  $K_{cr}$  em que a resposta do sistema apresenta oscilações mantidas de período  $P_{cr}$ , como visto na Figura 9. Se o sistema não apresentar oscilações, quaisquer que sejam os valores de  $K_p$ , então o método não se aplica.

Figura 9 – Sinal de saída com oscilação mantida de período  $P_{cr}$ .



Fonte: Ogata (2001, p. 546)

Com os valores de  $K_{cr}$  e  $P_{cr}$ , é possível aplicá-los na obtenção dos valores de  $K_p$ ,  $T_i$  e  $T_d$  de acordo com a fórmula sugerida por Ziegler e Nichols na Tabela 2.

Tabela 2 – Valores das constantes na regra de sintonia de Ziegler-Nichols para o segundo método

Tipo de controlador	$K_p$	$T_i$	$T_d$
P	$0,5K_{cr}$	$\infty$	0
PI	$0,45K_{cr}$	$\frac{1}{1,2}P_{cr}$	0
PID	$0,6K_{cr}$	$0,5P_{cr}$	$0,125P_{cr}$

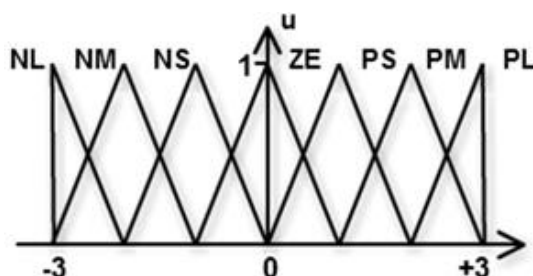
Fonte: Ogata (2001, p. 547)

### 2.4.3 Lógica e controle *fuzzy*

A lógica *fuzzy*, ou lógica nebulosa, ou ainda lógica difusa tem como objetivo adaptar a forma de pensar do homem e suas particularidades ao tratar matematicamente informações imprecisas (SHAW; SIMOES, 2007). É uma lógica multivalorada e que pode ser vista como uma extensão da lógica booleana.

O pioneiro na criação da lógica *fuzzy* foi o professor Lofti Zadeh. Seu primeiro trabalho sobre o tema abordou a questão de conjuntos *fuzzy*, definindo-os como classes que, para cada objeto contido em um conjunto, existe um número associado dentro do intervalo  $[0,1]$ , indicando o grau de pertinência,  $\mu$ , desse objeto à classe. Diferente dos chamados conjuntos comuns, como Zadeh se refere aos grupos que tratam da lógica booleana convencional, nos conjuntos *fuzzy* a função de pertinência, FP, de determinado objeto pode assumir qualquer valor dentro do intervalo  $[0,1]$ . Segundo Zadeh (1965), quanto mais o grau de pertinência estiver próximo da unidade, maior é a pertinência da variável analisada ao conjunto em questão. Pode ser visto, na Figura 10, um exemplo de funções de pertinência para um determinado conjunto *fuzzy*.

Figura 10 – Funções de pertinência de um conjunto *fuzzy*.



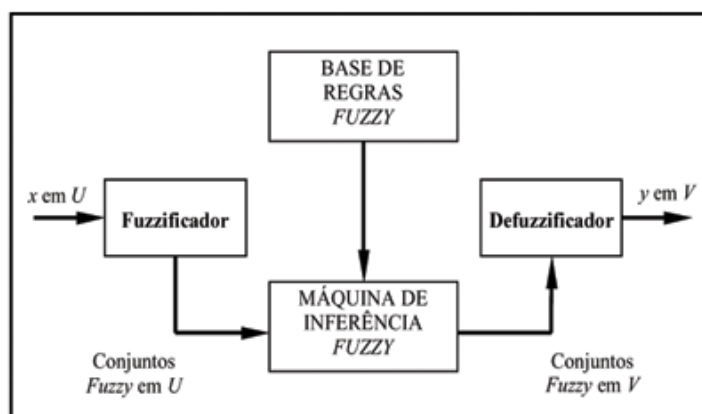
Fonte: Zhai et al. (2015, p. 16)

Em outra publicação a respeito da lógica *fuzzy*, Zadeh (1990) afirma que a ideia fundamental da lógica difusa é prover modelos para situações que são mais aproximadas do que efetivamente exatas, tendo em vista que assim é a forma de pensar do homem em grande parte das ocasiões. Com isso, como a lógica e controle *fuzzy* integram o conhecimento perceptual, a intuição e a análise de um especialista (ZHAI et al., 2015), essa é uma poderosa ferramenta na descrição do comportamento de sistemas altamente complexos ou mal definidos, onde algumas informações sobre o sistema estejam faltando, de tal forma que estes não possam receber uma descrição matemática precisa.

Um controlador *fuzzy* (Figura 11) é constituído de quatro etapas básicas:

1. fuzzificação: definição das FPs e obtenção de valores de grau de pertinência para uma dada variável dentro do universo.
2. Base de Regras: relação dos conjuntos *fuzzy* de entrada com os de saída por meio de inferências do tipo SE-ENTÃO.
3. Máquina de inferência: representação da ação do especialista interpretando a base de regras e aplicando-a ao objeto de estudo.
4. Defuzzificação: utilização da resposta da máquina de inferência para o cálculo do valor do sinal de controle de saída de acordo com as funções de pertinência de saída.

Figura 11 – Diagrama de blocos de um controlador *fuzzy* típico.



Fonte: Vieira et al. (2007, p. 118).

Existem diversos métodos de inferência, dentre os quais duas se destacam: a Mandani e a Takagi-Sugeno. No método de inferência Mandani, os conjuntos *fuzzy* de entrada são diretamente relacionados aos conjuntos *fuzzy* de saída; esse método é geralmente utilizado nas abordagens baseadas exclusivamente no conhecimento do especialista. Já método de inferência de Takagi-Sugeno, os conjuntos *fuzzy* de entrada são relacionados aos conjuntos *fuzzy* de saída por meio de equações lineares; esse método é muito efetivo em uma aproximação orientada a dados, isto é, quando dados medidos estão disponíveis ao projetista (ADOKO et al., 2013).

Os controladores *fuzzy* podem operar em algumas formas de controle já conhecidas como o controle proporcional. Para essa aplicação, é necessário que o controlador possua um ganho proporcional.

### 2.4.3.1 Ganho proporcional *fuzzy*

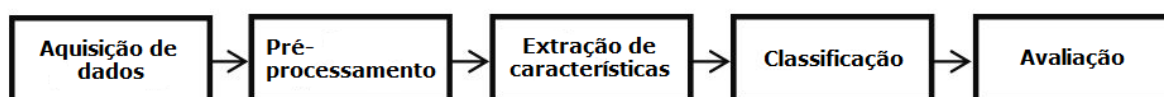
Uma forma simples de se incluir um ganho proporcional a um controlador *fuzzy*, especialmente aplicável aos controladores do tipo Mandani, é alterar o tamanho das bases das FPs de entrada ou saída. O ganho proporcional em um controlador Mandani é definido pela razão entre o tamanho da base da função de pertinência de saída pela de entrada. Se o tamanho da base de uma dessas funções é alterado, o ganho é alterado também. Por exemplo, se os tamanhos das bases das FPs de entrada são reduzidos à metade dos de saída, o controlador apresentará um ganho de valor 2. Isso ocorre porque haverá uma transição mais abrupta entre conjuntos *fuzzy* adjacentes, fazendo com que uma pequena variação no valor da variável *fuzzy* de entrada apresente uma grande variação na pertinência dessa mesma variável, resultando em uma grande variação correspondente na saída.

O controle *fuzzy* será responsável pelos movimentos laterais do carro da ponte rolante didática usada neste trabalho, a qual pode ser controlada por meio de comandos aplicados pelo usuário. Existem diversos tipos de comandos possíveis como, por exemplo, os comandos manuais e os comandos por voz, sendo este último objeto de estudo abordado na próxima seção desse trabalho.

## 2.5 COMANDOS POR VOZ

Um sistema comandado por voz pode ser muito útil em diversas aplicações, como centrais telefônicas, acesso e controle remotos de sistemas, inteligência artificial, autonomia de sistemas, entre outros. Entretanto, para que um sistema possa realizar um comando baseado em um sinal de voz, antes é necessário que tal sinal seja adquirido, processado, avaliado e interpretado, o que caracteriza os sistemas de reconhecimento de fala (HICKERSBERGER; ZAGLER, 2013). Uma interpretação válida de um sistema genérico de reconhecimento de fala pode ser visualizada na Figura 12.

Figura 12 – Esquema de sistema genérico de reconhecimento de fala.



Fonte: Adaptado de Silva (2013).



Em geral, sistemas de reconhecimento de fala realizam o processo de reconhecimento de padrões no sinal de voz. Tal reconhecimento depende de muitos fatores para que possa ser bem-sucedido, tais como duração da fala, idioma da fala, sexo do indivíduo que está falando, entre outros fatores. Por exemplo, o reconhecimento de dígitos pode ser um desafio de alta complexidade pelo fato de este ser um sinal de curta duração e também porque alguns dígitos podem soar parecidos com outros, o que geraria uma interpretação errada por parte da máquina (SILVA; SOUZA; BATISTA, 2013).

### 2.5.1 Pré-processamento

O pré-processamento é a primeira etapa de um sistema de Reconhecimento Automático de Voz, RAV, após a aquisição do sinal de voz. O propósito dessa etapa é a preparação do sinal de voz de forma a extrair parâmetros que o caracterizem a fim de possibilitar, por parte de um classificador, a diferenciação entre diferentes sinais de voz. Essa etapa pode ser dividida em três principais tarefas: (1) conversão A/D e pré-ênfase; (2) análise espectral e (3) extração de características (CIPRIANO, 2001).

#### 2.5.1.1 Conversão A/D e pré-ênfase

O sinal de voz é, em sua essência, um sinal analógico. Portanto, assim que o sinal de voz é capturado por um transdutor (um microfone, por exemplo) é necessário realizar a conversão analógico-digital, A/D, a fim de que esse sinal seja representado digitalmente, permitindo, assim, a aplicação dessa representação em sistemas digitais como o sistema RAV.

Tendo o sinal de voz em sua representação digital, é interessante que este seja submetido a um filtro de pré-ênfase que tem o propósito de “equalizar o espectro de voz e melhorar o desempenho da análise espectral” (CIPRIANO, 2001). Esse filtro permitirá que algumas frequências (geralmente altas) tenham suas magnitudes aumentadas, equalizando o espectro em termos de magnitude. A função de transferência do filtro de pré-ênfase é definida pela Equação 5:

$$H_{pre}(z) = 1 - \alpha_{pre}z^{-1} \quad (5),$$

sendo  $\alpha_{pre}$  o coeficiente de pré-ênfase que varia de 0,9 a 1,0. Tendo o sinal de voz sido adquirido, convertido digitalmente e equalizado pelo filtro de pré-ênfase, o próximo passo é realizar a análise espectral desse sinal.

#### 2.5.1.2 Análise espectral

A etapa de análise espectral começa pelo janelamento, que é dividir o sinal em diversos quadros de mesma duração. O janelamento é necessário para o processamento do sinal de voz porque, embora esse tipo de sinal seja estocástico e não-estacionário, pode ser considerado estacionário em curtos intervalos de tempo referentes a cada quadro possibilitando, assim, as análises espectrais (RABINER; SCHAFER, 1978). O janelamento ocorre com a sobreposição de quadros adjacentes a fim de reduzir grandes variações nas transições e descontinuidades no início e fim de cada quadro. Além disso, é ideal que cada janela possua duração de 10 a 50 milissegundos e as sobreposições sejam de 10 a 30 milissegundos, como afirmam Rabiner e Juang (1993). Uma janela comumente aplicada é a janela de Hamming (HAMMING, 1989), modelada pela equação abaixo

$$w(n) = 0,54 - 0,46 \cos\left(\frac{2\pi n}{N-1}\right), 0 \leq n \leq N-1 \quad (6),$$

em que  $w(n)$  é a janela de Hamming e  $N$  é a sua largura.

Tendo sido aplicado o janelamento, pode-se prosseguir para a análise espectral propriamente dita, sendo realizada por meio da Transformada Discreta de Fourier, DFT (*Discrete Fourier Transform*) implementada pela Transformada Rápida de Fourier, FFT (*Fast Fourier Transform*), algoritmo de rápida execução da DFT, reduzindo o tempo de processamento e melhorando o desempenho. A aplicação da FFT visa levar o sinal para o domínio da frequência por dois principais motivos: primeiro, a percepção de sons na audição humana é dependente da frequência (RABINER; SCHAFER, 2007); segundo, a manipulação matemática e extração de parâmetros do sinal de voz é facilitada, estando esse no domínio da frequência, como poderá ser visto no próximo tópico.

### 2.5.1.3 Extração de características

Diversos tipos de coeficientes do sinal de voz podem ser extraídos a fim de análise do sinal. Dentre estes é possível destacar os coeficientes mel-cepstrais. Os coeficientes mel-cepstrais, MFCC (*Mel-Frequency Cepstral Coefficients*), são derivados de um tipo de representação cepstral do sinal, sendo o cepstro definido por Bogert, Healy e Tukey (1963 apud RANDALL, 2017, p. 3) como “o espectro de potência do logaritmo do espectro de potência” de um dado sinal. Outros coeficientes também podem ser usados, como o de Frequências de Linha Espectral, LSF (*Line Spectral Frequencies*). Apesar disso, este trabalho foca na utilização dos coeficientes MFCC tendo em vista que são mais utilizados em projetos de reconhecimento de voz e são mais bem-conceituados que outros coeficientes. Para compreender melhor como se dá a extração desses coeficientes, é necessário antes analisar a modelagem matemática do sinal de voz.

Segundo Rabiner e Schafer (1978), o sinal de voz,  $y(n)$ , é fruto da convolução entre o sinal de excitação da glote,  $u(n)$ , a resposta impulsiva do trato vocal,  $v(n)$ , e a radiação,  $r(n)$ , como pode ser visto na equação a seguir.

$$y(n) = u(n) \otimes v(n) \otimes r(n) \quad (7).$$

Por conveniência, é comum combinar  $v(n)$  e  $r(n)$  em  $v'(n)$ , resultando em

$$y(n) = u(n) \otimes v'(n) \quad (8).$$

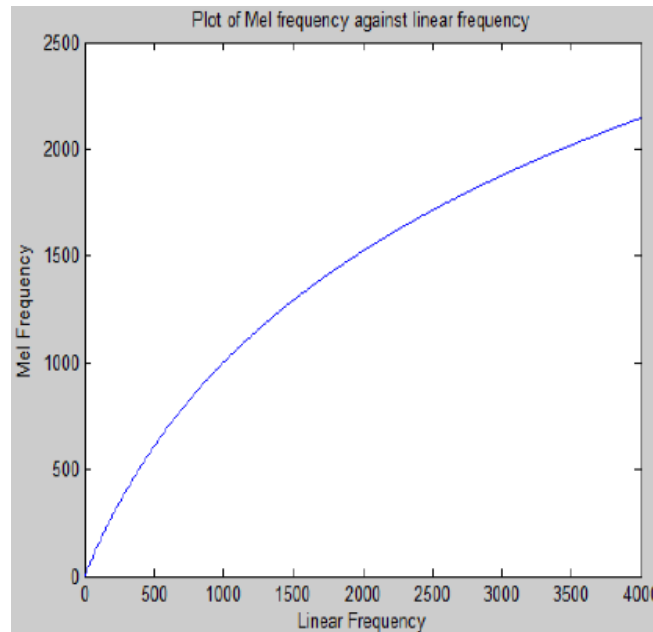
A manipulação e extração de parâmetros de um sinal convoluído no domínio do tempo é muito complexa. Para facilitar essa análise, o sinal de voz é levado ao domínio da frequência por meio da transformada de Fourier, como visto na Seção 2.5.1.2. Além disso, a FFT permitirá obter o espectro de potência do sinal, desejado para a extração de características. Após a FFT e a obtenção do espectro de potência, o sinal é representado pela Equação 9, em que  $Y(s)$  representa o espectro de potência do sinal de voz.

$$Y(s) = U(s)V'(s) \quad (9).$$

Para prosseguir na obtenção dos MFCCs, é necessário que as frequências do sinal de voz em hertz sejam convertidas para a escala mel. A escala mel tem o propósito de representar de forma mais fiel a percepção auditiva humana, considerando que existe uma diferença entre a frequência real e a interpretada pelo ouvido humano,

principalmente para baixas frequências (TAYLOR, 2009). Além disso, como pode ser visto na Figura 13, a escala mel apresenta uma relação logarítmica com a escala de frequência linear, simulando o comportamento do ouvido humano, essencialmente logarítmico (SIDDHANT; CHEERAN, 2014).

Figura 13 – Gráfico de frequências Mel por frequências lineares.



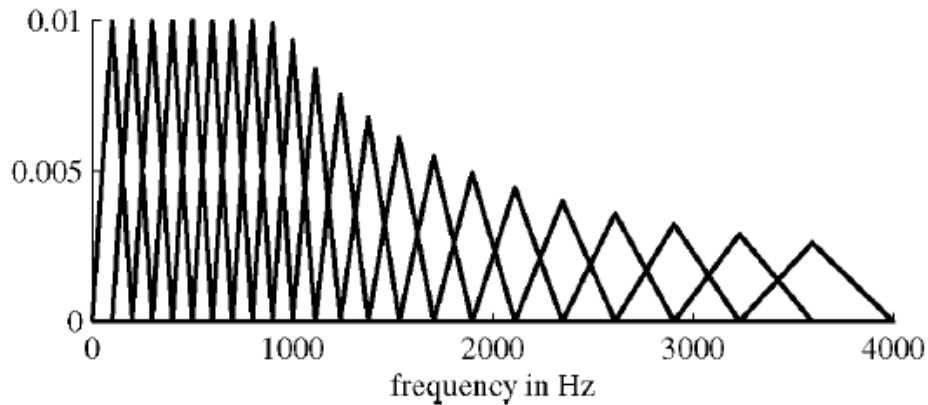
Fonte: Siddhant (2014, p. 1822).

A relação entre as frequências das escalas hertz e mel pode ser descrita pela equação:

$$F_{mel} = 2595 \cdot \log_{10} \left( 1 + \frac{F_{Hz}}{700} \right) \quad (10).$$

Para realizar a conversão entre as escalas linear e mel, é utilizado um banco de filtros mel. O banco de filtros mel é composto por vários filtros passa-banda triangulares, uniformemente espaçados na escala mel, porém não uniformemente espaçados na escala linear, resultando em um número maior de filtros na região de baixa frequência. Na escala linear, os filtros possuem largura de banda constantes para frequências até 1 kHz, sendo espaçados segundo uma escala logarítmica para frequências acima dessa faixa, como visto na Figura 14 (RABINER; SCHAFER, 2007). Na Tabela 3, é mostrado um banco de filtros mel com suas respectivas frequências centrais, podendo ser observada uma variação linear até 1 kHz e logarítmica acima disso.

Figura 14 – Exemplo de banco de filtros mel.



Fonte: Rabiner (2007, p. 71).

Como visto na Equação 9, as componentes do sinal de voz estão combinadas por meio de uma multiplicação. Para que essas componentes sejam separadas a fim de se obter os MFCCs, após a filtragem é obtido o logaritmo do sinal de voz, o que fará com que as componentes sejam separadas em parcelas de uma soma. Isso tem como base a propriedade do logaritmo em que o logaritmo de um produto é a soma dos logaritmos das parcelas desse mesmo produto. Em resumo, o resultado é mostrado na Equação 11:

$$\log Y(s) = \log U(s) + \log V'(s) \quad (11).$$

Por fim, para se obter os coeficientes mel-cepstrais, é aplicada a transformada discreta do cosseno, DCT (*Discrete Cosine Transform*), ao logaritmo do espectro de potência. A DCT irá levar o sinal de voz de volta ao domínio do tempo, agora com suas componentes desconvoluídas e separadas (GUARAV; GOPAL, 2012). Portanto, os coeficientes são obtidos por meio da equação a seguir:

$$c(i) = \sum_{k=1}^M \log Y(s) \cos \left[ i \left( k - \frac{1}{2} \right) \frac{\pi}{M} \right], \quad i = 1, \dots, L \quad (12),$$

em que  $c$  é o coeficiente mel-cepstral,  $i$  é o índice do coeficiente,  $k$  é o índice do filtro,  $M$  é o número total de filtros e  $L$  é o número total de coeficientes. Além disso, é comum descartar coeficientes de ordem superior tendo em vista que a DCT acumula a maior parte da informação do sinal nos coeficientes de ordem inferior, reduzindo custo computacional (YUAN, 2004), além do fato de que os coeficientes de ordem inferior concentram as informações do trato vocal, desejado para realizar o reconhecimento de um comando de voz (TAYLOR, 2009).

Tendo os coeficientes mel-cepstrais em mãos, a próxima etapa é aplicá-los em um classificador responsável por realizar a identificação do comando de voz.

Tabela 3 – Frequências centrais para banco de filtros mel.

Índice do filtro	Frequência central (Hz)	Índice do filtro	Frequência central (Hz)
1	100	11	1149
2	200	12	1320
3	300	13	1516
4	400	14	1741
5	500	15	2000
6	600	16	2297
7	700	17	2639
8	800	18	3031
9	900	19	3482
10	1000	20	4000

Fonte: Adaptado de Martins (2014).

### 2.5.2 Classificação

Considerando as informações baseadas nas características do sinal de voz que foram extraídas com o auxílio dos coeficientes, um determinado classificador é utilizado para realizar a interpretação do sinal de voz. Tais classificadores funcionam como algoritmos de aprendizagem de máquina, possibilitando o reconhecimento automático de fala por parte da máquina. Alguns classificadores se destacam, como é o caso do Modelo Oculto de Markov, HMM (*Hidden Markov Model*), um classificador de mecanismo simples e flexível para a modelagem de sequências de comprimento variável e bem-conceituado na classificação de dígitos da língua inglesa (NIMJE; SHANDILYA, 2011). Além disso, a utilização de uma Rede Neural Artificial, RNA, como classificador de palavras é bem comum como mostra Gevaert et al. (2010),

sendo esse o foco deste trabalho.

## 2.6 REDE NEURAL ARTIFICIAL

Por Haykin (1998, p. 2), rede neural é um “processador de distribuição massivamente paralela feito de unidades de processamento simples, o qual tem uma propensão natural por armazenar conhecimento experimental e torná-lo disponível a uso”. Seu funcionamento tem o propósito de reproduzir o funcionamento do cérebro humano, conseguindo assemelhar-se bem em dois aspectos: primeiro, adquire conhecimento do ambiente por meio de processos de aprendizado; e segundo, os pesos sinápticos da rede, que representam as interconexões neurais, são capazes de armazenar o conhecimento adquirido.

As RNAs são amplamente utilizadas em processos de aprendizado de máquina por apresentarem diversas características vantajosas a esses processos como adaptação a partir da experiência, capacidade de aprendizado e generalização, organização de dados, tolerância a erros, armazenamento distribuído e prototipagem facilitada (NUNES et al., 2017).

Devido a tão numerosas vantagens, as redes neurais artificiais estão presentes em diversos estudos e projetos como desenvolvimento de sistema antifraude no monitoramento de transações (SAPOZHNIKOVA et al., 2017) e modelagem para predição e estimativa de volume de chuva e fluxo de água em um rio (FARIAS et al., 2013).

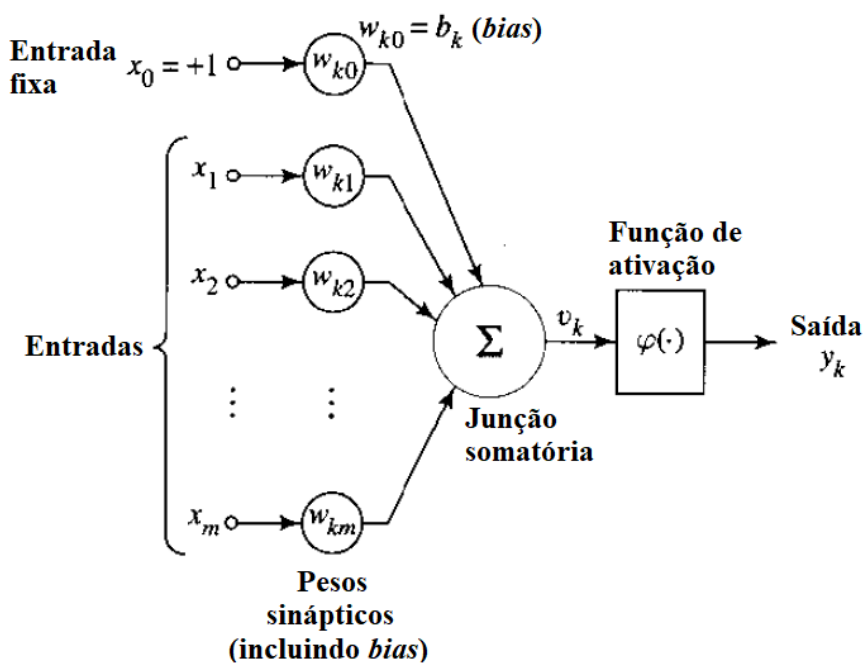
Existem diversas topologias de redes neurais; uma das mais conhecida e utilizadas, aplicada também nesse trabalho, é a rede *perceptron* multicamadas.

### 2.6.1 *Perceptron* multicamadas

Primeiramente apresentado por Rosenblatt (ROSENBLATT, 1958), o *perceptron* de camada única é a forma mais simples de uma rede neural. Constituído de um único neurônio, através dos ajustes dos pesos sinápticos e correlação da saída com a entrada por meio de uma função de ativação esse modelo primitivo de rede neural poderia resolver problemas de separação de dados ditos linearmente separáveis, isto

é, conjuntos que podem ser organizados em lados opostos de um hiperplano. A Figura 15 mostra um *perceptron* simples com todas as suas características.

Figura 15 – *Perceptron* de camada única.



Fonte: Adaptado de Haykin (1998, p.13).

Em 1986, Rumelhart, Hinton e Williams (1986) introduziram uma nova forma de aprendizado chamado *back-propagation*, ampliando o mundo das redes neurais.

O algoritmo *back-propagation* divide-se em duas etapas:

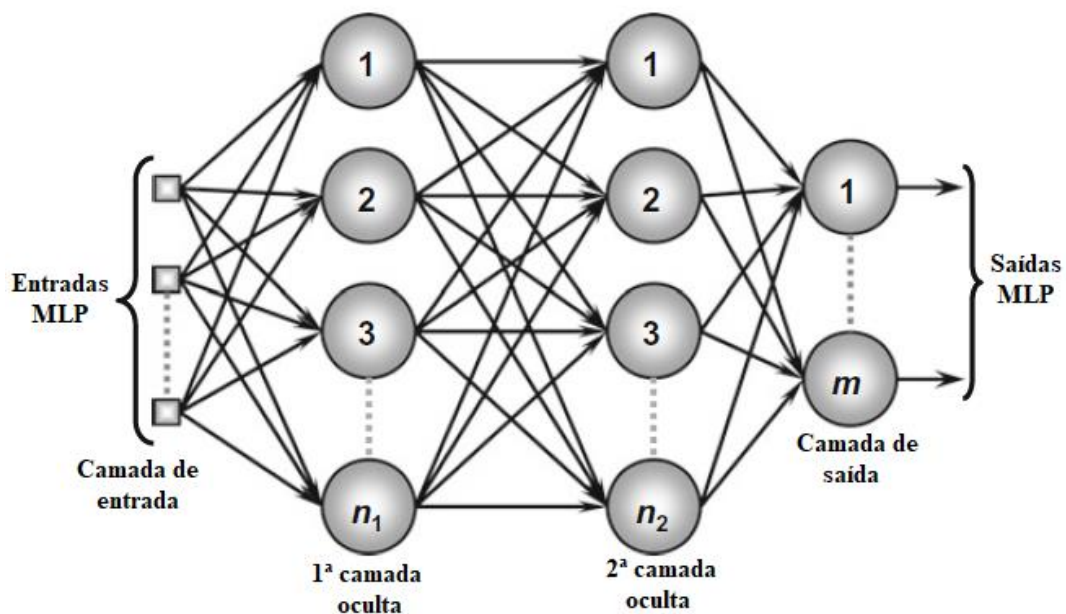
1. propagação adiante (*forward*) – os sinais de entrada de uma amostra de conjunto de treinamento são aplicados à entrada da rede, sendo propagados até a produção de sua saída.
2. Propagação reversa (*backward*) – os sinais de saída são comparados com as respostas desejadas, gerando erros que serão utilizados para ajustar os pesos e limiares de todos os neurônios.

A aplicação desse algoritmo pressupõe uma alteração fundamental na topologia inicial do *perceptron*: a adição de camadas intermediárias de neurônios, chamadas de camadas ocultas. Com esse acréscimo de camadas intermediárias, teríamos o que se chama de *perceptron* multicamadas, PMC. Nessa topologia, todos os neurônios de uma camada estão conectados a todos os outros das camadas adjacentes. As entradas e saídas estão conectadas às camadas externas, enquanto que a presença



das camadas ocultas possibilitará a resolução de problemas não-linearmente separáveis, expandindo os usos e aplicações das redes neurais ao que vemos hoje. A Figura 16 mostra um PMC com todas as suas características.

Figura 16 – *Perceptron* multicamadas.



Fonte: Adaptado de Nunes et al. (2017, p. 56).

### 3 DESENVOLVIMENTO

A partir da fundamentação teórica apresentada, é possível abordar como cada etapa do trabalho foi desenvolvida e implementada com suas respectivas metodologias.

#### 3.1 DESENVOLVIMENTO DA INTERFACE GRÁFICA PARA O USUÁRIO

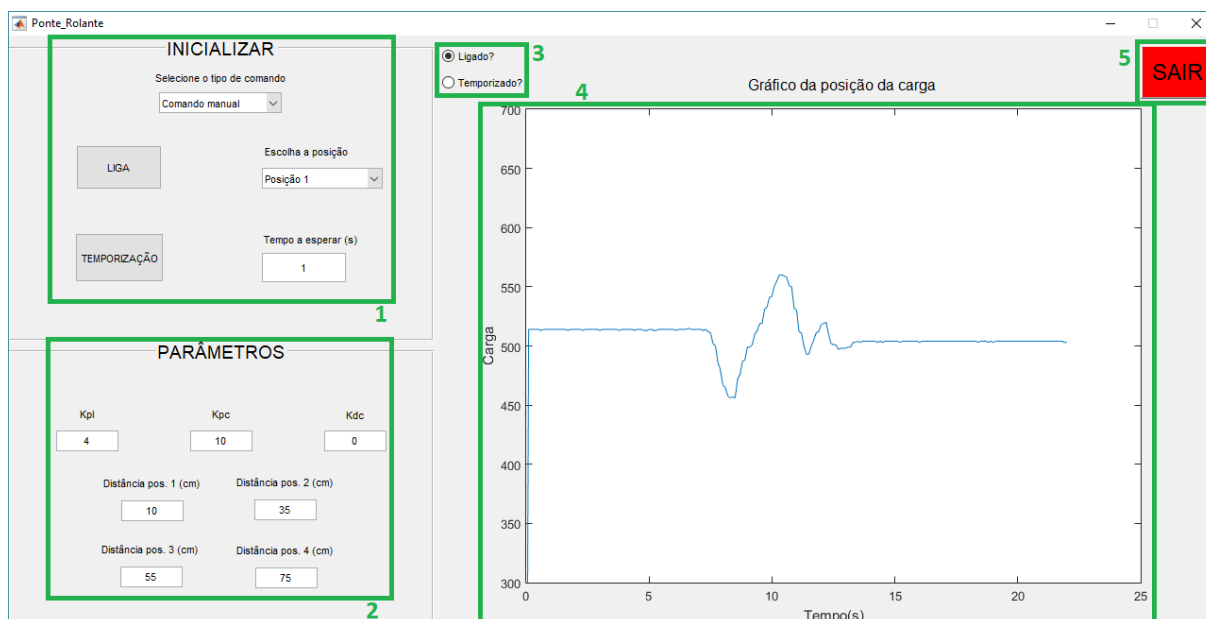
Para o desenvolvimento da Interface Gráfica para o Usuário, foi utilizado o *software* MatLab® R2012b (MATHWORKS, 2012). O motivo da escolha de tal programa se dá pela versatilidade e robustez da ferramenta.

Antes que a criação da GUI utilizada neste trabalho fosse iniciada, diversos testes foram necessários a fim de possibilitar a familiarização com a ferramenta de desenvolvimento de interfaces no MatLab®, bem como com a comunicação entre o *software* e o *hardware*. Alguns desses testes incluíram o envio de alguns caracteres da GUI para o ATmega 2560, verificando se a recepção e interpretação desses ocorreu de forma correta, por exemplo.

Um importante problema que necessitava ser contornado para a implementação da interface era a necessidade de supervisionar um processo em tempo real utilizando uma ferramenta que é orientada a evento. Por exemplo, como seria possível verificar a movimentação da carga na ponte rolante de forma autônoma por parte da interface? Uma solução encontrada para resolver essa questão foi a utilização de temporizadores na GUI. Tais temporizadores funcionariam como interrupções programadas, permitindo receber dados periódicos de uma fonte (como a carga da ponte rolante, por exemplo) sem a necessidade de explicitar isso por meio de comandos diretos na GUI.

A GUI desenvolvida apresenta algumas seções, enumeradas de acordo com o que se pode ver na Figura 17.

Figura 17 - Estrutura da GUI.



Fonte: Elaborado pelo autor (2018).

São mostradas abaixo, em ordem numérica, as funções das seções enumeradas vistas na Figura 17.

1. Inicializar: permite seleccionar o tipo de comando desejado (manual ou por voz) a fim de iniciar o movimento da ponte por meio do acionamento dos respectivos botões, além de possibilitar a ativação do movimento da ponte em modo de temporização.
2. Parâmetros: apresenta todos os parâmetros de ganho ( $K_{pl}$  – ganho proporcional para o movimento lateral;  $K_{pc}$  e  $K_{dc}$  – ganhos proporcional e derivativo para antibalanco da carga) e posições para a movimentação da ponte, permitindo ao usuário alterá-los conforme desejar.
3. LEDs de indicação: indicam alguns estados da ponte rolante em tempo real como, por exemplo, se ela está ligada (movimentando-se) e/ou se sua movimentação está em modo de temporização.
4. Gráfico da posição da carga: supervisiona, em tempo real, a movimentação da carga de acordo com o tempo de execução do programa.
5. Sair: ao ser pressionado, encerra todas as conexões e fecha a janela, finalizando o programa.

A interface possibilita dois modos básicos de comando dos movimentos na ponte rolante: comando manual e comando por voz. Será visto um pouco mais sobre esses modos nas próximas seções.

### **3.1.1 Comando manual**

No modo de comando manual, o usuário deve definir o valor de todos os parâmetros, inclusive as distâncias referentes à cada posição da ponte, tendo como referencial o sensor ultrassom responsável pela realimentação do movimento lateral do carro. Tendo os parâmetros sido definidos, o acionamento do carro da ponte é feito de forma manual pelo usuário, isto é, basta que o usuário escolha a posição desejada e então clique no botão “LIGA”. Com isso, o programa irá extrair da interface os valores dos parâmetros de ganho definidos pelo usuário, bem como a posição desejada para a movimentação, a fim de montar o pacote de dados para a comunicação serial com o microcontrolador (ver Seção 3.1.3 para mais detalhes sobre o pacote de dados). A GUI mostrada na Figura 17 representa o modo de comando manual da interface.

O usuário também pode optar por clicar no botão “TEMPORIZAÇÃO” antes de efetivamente acionar o carro. Nesse caso, assim que o botão “LIGA” for acionado, o carro da ponte entrará no modo temporizado. Nesse modo, o carro moverá ininterruptamente entre as posições escolhidas para o movimento, tendo um intervalo de espera entre uma posição e outra definido pelo usuário na seção “INICIALIZAR”, presente na interface. Para tal, antes que o carro inicie seu movimento, será solicitado que o usuário escolha, dentre a lista de posições, a posição referente a cada movimento da ponte, de acordo com a sequência de movimento. Essa sequência é armazenada pelo programa em um vetor, sendo acessado na transição de uma posição para outra.

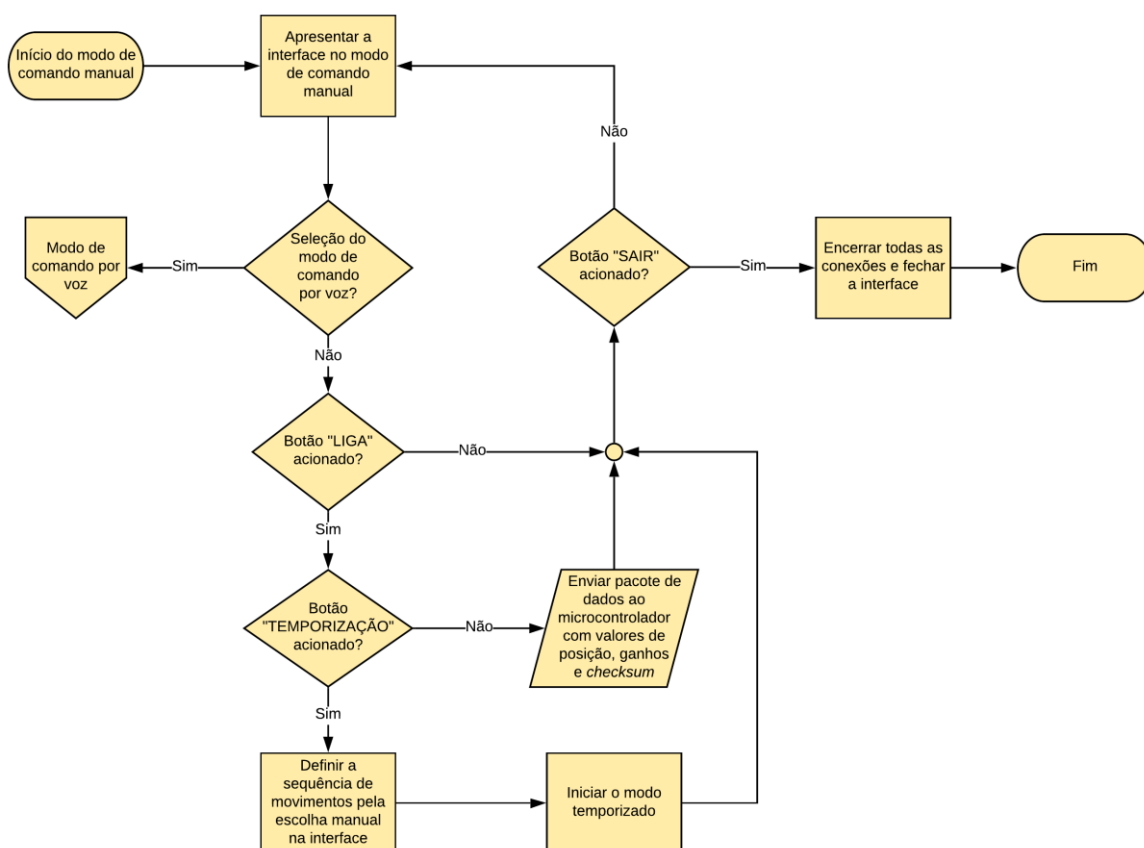
Para que o movimento temporizado seja possível, o programa irá utilizar uma informação presente no pacote de dados proveniente do microcontrolador que informa se o carro da ponte rolante chegou no seu destino. Caso ele tenha chegado, isso significa que o carro está parado em uma das posições e, portanto, a temporização deve ser ativada. O temporizador responsável por essa operação irá cronometrar o tempo de espera de acordo com o valor extraído da interface, em segundos. Assim que o tempo estoura, um novo pacote de dados com a informação sobre a próxima

posição é enviado ao microcontrolador, o que fará com que o carro se movimente até que tenha alcançado essa posição. Enquanto o carro se movimenta entre duas posições, o temporizador está desativado e somente será reativado no momento em que o carro chegar à próxima posição desejada. Esse movimento é contínuo e somente poderá ser interrompido caso o usuário clique novamente no botão “TEMPORIZAÇÃO”, desabilitando-a.

O fluxograma mostrado na Figura 18 traz um apanhado geral do funcionamento da interface no modo de comando manual. Já o fluxograma da Figura 19 mostra com detalhes o funcionamento do programa quando acionado o modo temporizado.

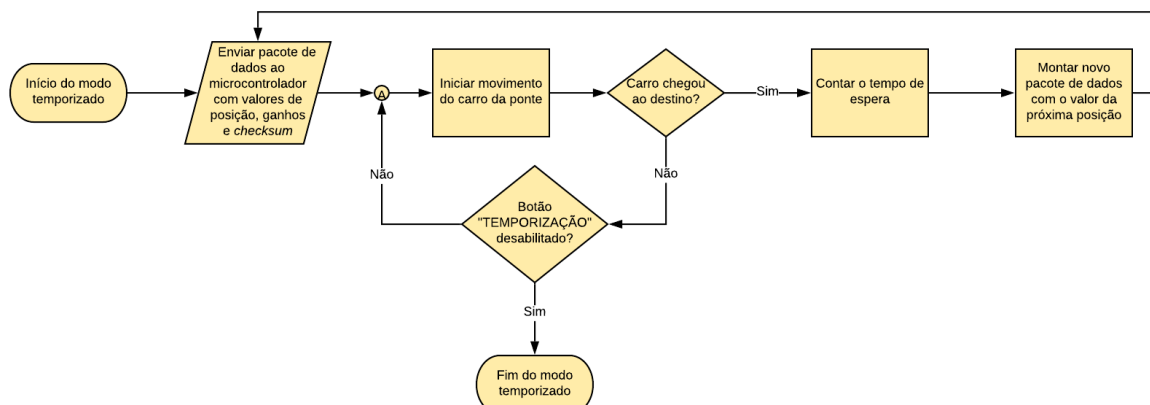
O usuário também pode optar pelo acionamento via comando de voz, o que será visto a seguir.

Figura 18 – Fluxograma do modo de comando manual da GUI.



Fonte: Elaborado pelo autor (2019).

Figura 19 – Fluxograma do funcionamento do modo temporizado.



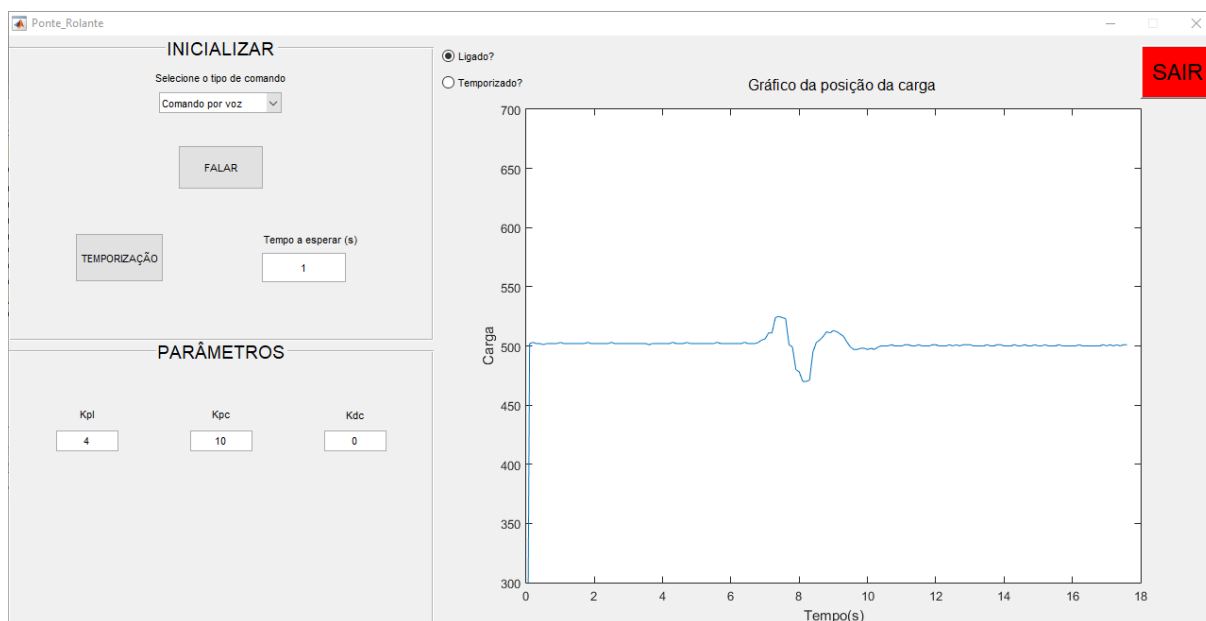
Fonte: Elaborado pelo autor (2019).

### 3.1.2 Comando por voz

Nesse modo, o acionamento do carro se dá através do botão “FALAR”. Ao clicar nesse botão, será solicitado que o usuário fale uma cor relativa à posição desejada para a movimentação. Diferentemente do modo de comando manual, os valores das posições não podem ser alterados pelo usuário; são valores previamente definidos de acordo com a posição dos indicadores de cores na estrutura da ponte, conforme mostrado na Tabela 4 e na Figura 21. Assim que a cor é falada pelo usuário, o programa fará o reconhecimento da posição desejada para a movimentação do carro de acordo com qual cor foi dita (a Seção 3.3 trará mais detalhes sobre esse processo de reconhecimento de voz). A Figura 20 mostra a GUI no seu modo de comando por voz. Tendo a posição desejada, o pacote de dados é enviado ao ATmega 2560 para a movimentação e controle do carro da ponte.

O modo temporizado do comando por voz é semelhante ao do comando manual, com uma exceção: a sequência dos movimentos é definida por voz. Com isso, ao clicar no botão “TEMPORIZAÇÃO” e, em seguida, no botão “FALAR”, a interface irá solicitar que o usuário fale quatro cores, uma de cada vez, de acordo com a sequência desejada dos movimentos temporizados. O fluxograma mostrado na Figura 22 traz um resumo do funcionamento da interface no modo de comando por voz.

Figura 20 – Interface para o modo de comando por voz.



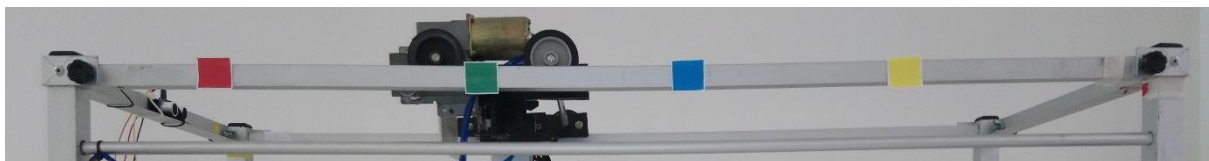
Fonte: Elaborado pelo autor (2019).

Tabela 4 – Posições para o movimento no modo de comando de voz de acordo com as respectivas cores.

Cor	Posição (cm)
Vermelho	10
Verde	35
Azul	55
Amarelo	75

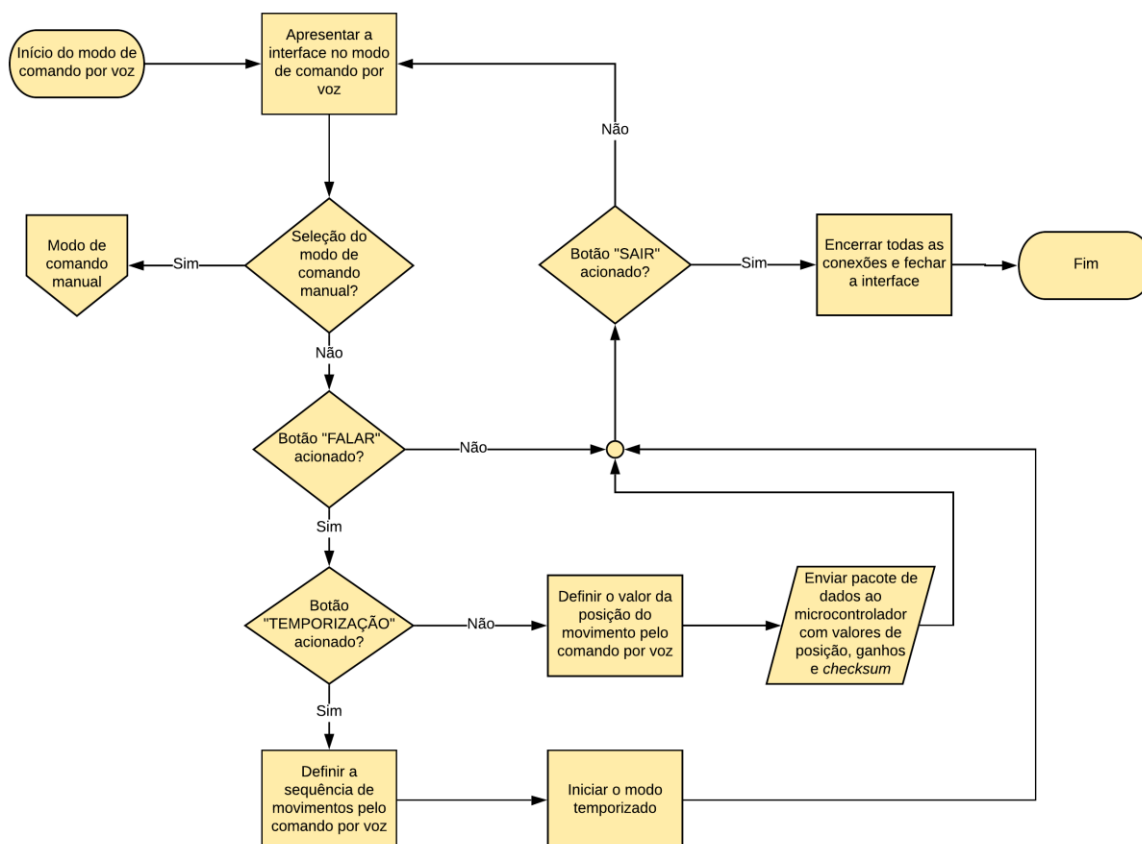
Fonte: Elaborado pelo autor (2019).

Figura 21 – Indicadores de cores na estrutura da ponte rolante para os comandos por VOZ.



Fonte: Elaborado pelo autor (2019).

Figura 22 – Fluxograma do modo de comando por voz da GUI.



Fonte: Elaborado pelo autor (2019).

### 3.1.3 Comunicação dos dados

Para que o controle do carro da ponte pudesse ocorrer, bem como a visualização na interface de algumas informações sobre o protótipo, uma comunicação serial foi estabelecida entre o microcontrolador e a GUI. Ambas as partes envolvidas na comunicação vão enviar e receber dados: o ATmega 2560 recebe os dados de comando e controle provenientes da interface enquanto que a GUI recebe os dados de posições da carga e flag de movimentação do carro provenientes do microcontrolador.

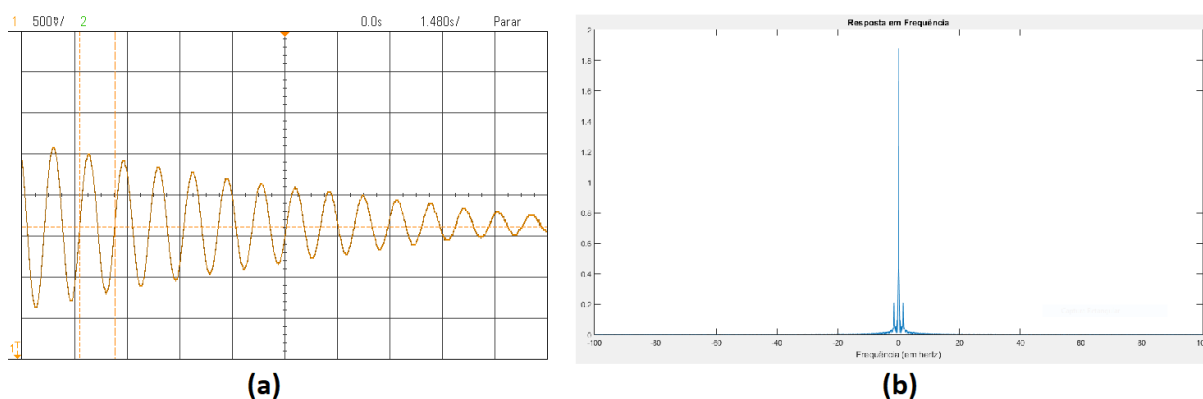
A preocupação com a integridade dos dados provenientes do ATmega 2560 e recebidos pela GUI não é grande, tendo em vista que, devido à forma como foi programado o microcontrolador, os dados esperados pela interface são enviados continuamente e com uma frequência muito grande pelo ATmega 2560. Com isso, ainda que um pacote de dados se perca, outros pacotes enviados logo em seguida



poderão informar, com certa redundância, uma possível mudança no estado da ponte rolante.

Para exemplificar isso, é obtida a resposta em frequência da carga em seu movimento oscilatório. O motivo dessa análise em frequência é para verificar, pelo teorema de Nyquist, a taxa mínima de amostragem do potenciômetro da carga. O teorema de Nyquist diz que um sinal deve ser amostrado em uma frequência de, pelo menos,  $2f_M$  para que não se percam informações relevantes, sendo  $f_M$  a maior frequência do sinal (NYQUIST, 1928). Na Figura 23 podem ser vistos os gráficos das respostas elétrica e em frequência da carga da ponte rolante. Para a obtenção desses gráficos, soltou-se o pêndulo da carga da ponte rolante a partir de uma extremidade, permitindo-o oscilar livremente até que, por efeito do atrito do potenciômetro, as oscilações cessassem.

Figura 23 – Respostas (a) elétrica e (b) em frequência da carga da ponte rolante.



Fonte: Elaborado pelo autor (2019).

A frequência máxima apresentada pelo gráfico de resposta em frequência, isto é, aquela que apresenta lóbulos laterais de magnitudes consideráveis é de, aproximadamente,  $f_M = 1,5$  Hz. Com isso, a frequência mínima de amostragem é de 3 Hz, o que resulta em um tempo máximo de amostragem de 333 milissegundos, aproximadamente. Como a amostragem do potenciômetro feita pelo microcontrolador é de acordo com o tempo de *polling* do programa embarcado (de 0,5 a 1,5 milissegundos, aproximadamente) e o tempo de atualização do temporizador da interface responsável pela leitura dos pacotes vindos do ATmega 2560 e plotagem do gráfico de posição da carga é de, aproximadamente, 5 milissegundos, conclui-se que o teorema de Nyquist é atendido, permitindo até mesmo que um pacote seja perdido já que a frequência de amostragem efetiva é muito maior que a frequência mínima

requerida. Ainda assim, na programação da interface, buscou-se tratar os dados recebidos, a fim de eliminar valores espúrios e erros de comunicação.

Já a preocupação com a integridade dos dados provenientes da GUI e recebidos pelo ATmega 2560 é bem grande, tendo em vista que, ao se realizar um comando na interface, um único pacote de dados é enviado pelo programa, ou seja, não acontece um envio contínuo de pacote. Devido a isso, garantir a integridade do pacote recebido é fundamental porque a perda de um pacote significa a perda de uma instrução executada pelo usuário. O método abordado nesse trabalho para validação do recebimento de dados por parte do ATmega 2560 é a implementação do *checksum*. O pacote implementado utilizado no envio de dados da interface para o microcontrolador é formatado segundo o padrão de comunicação serial. Na Figura 24, pode-se ver a estruturação do pacote de dados.

Figura 24 – Estruturação e exemplo do pacote de comunicação de dados.

STX	Distância	;	kP Lateral	;	kP Carga	;	kD Carga	:	Checksum	ETX
<	30	;	5	;	10	;	1	:	214	>

Fonte: Elaborado pelo autor (2019).

O pacote começa com um *byte* de início, STX, representado pelo caractere "<". Em seguida, o pacote aglutina *bytes* correspondentes à distância do movimento desejado,  $K_p$  para o controle do movimento lateral e  $K_p$  e  $K_d$  para o controle antibalanço da carga, sempre com um *byte* de separação, representado pelo caractere ";", entre cada dado. Então, um *byte* representado pelo caractere ":" indica o início dos *bytes* correspondentes ao valor do *checksum*, sendo seguido pelo *byte* de fim de pacote, ETX, representado pelo caractere ">".

Vale ressaltar que, como o pacote é enviado como um vetor de caracteres, o número de *bytes* para cada dado pode variar, tendo em vista que cada caractere e dígito do pacote representa um único *byte* (por exemplo, o dado referente à distância pode variar entre números com um ou dois dígitos, ou seja, com um ou dois *bytes* na comunicação); além disso, as constantes dos controles podem admitir valores decimais. Tudo isso fará com que o número de *bytes* de cada pacote seja dinâmico.

Por essa causa é que se escolheu fazer uso dos caracteres de separação, além também de facilitar a leitura e interpretação do pacote por parte do microcontrolador. Ao receber o pacote de dados, o ATmega 2560 extrai os *bytes* de informação compreendidos entre os *bytes* de início e fim e calcula o *checksum* do pacote recebido. O cálculo do *checksum* considera a soma truncada de 8 bits dos caracteres de todas as constantes do pacote; não são incluídos nessa soma os caracteres de início e fim de pacote e os caracteres de separação. Caso a soma entre o *checksum* calculado e o enviado resulte em zero, o programa prossegue com suas rotinas; do contrário, esse é um indicativo que houve corrupção dos bytes enviados e, com isso, é enviado de volta um código de erro de comunicação serial definido como “1111” à interface que, por sua vez, envia novamente o pacote ao microcontrolador. Esse processo pode ocorrer até 3 vezes consecutivamente; após isso, devido à impossibilidade de se estabelecer uma comunicação estável entre a interface e o microcontrolador, encerra-se a ligação serial e o programa é finalizado.

Com a interface concluída, o próximo passo é iniciar o projeto de controle híbrido para o sistema.

## 3.2 PROJETO DE CONTROLE HÍBRIDO

Conforme visto na Seção 2.4, o controle híbrido para esse trabalho consiste em um controlador operando na forma clássica PD para o controle antibalço da carga e na forma inteligente *fuzzy* para o controle do movimento lateral do carro da ponte rolante. Como o movimento lateral influencia diretamente na análise do balanço da carga e em seu projeto de controle, decidiu-se por iniciar o projeto do controlador híbrido pelo controle *fuzzy* do movimento lateral. Contudo, antes mesmo de começar essa parte do controle, identificou-se que o motor do carro possui uma zona morta; com isso, é necessário contornar essa limitação imposta à movimentação do sistema.

### 3.2.1 Zona morta do motor

A zona morta é um tipo de não-linearidade em que, para uma dada região de funcionamento do sistema, a saída permanece nula mesmo com variações na

entrada; a saída apenas apresentará alguma magnitude quando a entrada ultrapassar um certo limite (BAI; CHO, 1995).

O motor DC usado neste trabalho possui uma região de funcionamento em que permanece com a sua saída nula (parado) mesmo com variações na tensão de entrada. Essa zona morta foi identificada para alterações nos valores de PWM entrada de até 55, considerando o PWM de resolução de 8 bits utilizado neste trabalho. Ou seja, variações no PWM de entrada, fornecido pelo *drive* de acionamento, na faixa de 0 a 55 não faz o carro se movimentar; para valores de PWM maiores que 55, o carro passa a se movimentar normalmente. Essa condição de zona morta é altamente prejudicial para o controle tendo em vista que, ao se tentar corrigir um pequeno erro na variável de controle, o carro não se movimentaria porque o valor de PWM associado a essa correção estaria dentro da zona morta do motor.

Com isso, para contornar essa situação, foi realizada uma linearização no programa embarcado no microcontrolador de forma que os valores de PWM calculados que estivessem na zona morta pudessem resultar em algum movimento efetivo do carro da ponte rolante. A linearização foi feita conforme a Equação 13:

$$s_{PWM} = \frac{255 - 55}{255} \cdot s_c + 55 = 0,784 \cdot s_c + 55 \quad (13),$$

em que  $s_c$  e  $s_{PWM}$  representam, respectivamente, os valores dos sinais de controle calculado e aplicado na saída PWM após a linearização.

Com essa solução para o problema da zona morta implementada, é possível iniciar o projeto de controle *fuzzy* do movimento lateral.

### 3.2.2 Controle *fuzzy* do movimento lateral

O controle do movimento lateral visa fazer o carro chegar até as posições finais desejadas pelo usuário, otimizando o tempo de excursão e o impacto de oscilação sobre a carga. Para o controle de tal movimento lateral, como já foi definido antes, escolheu-se pela utilização de um controle *fuzzy*. Essa escolha evitará complicadas e imprecisas modelagens matemáticas para esse sistema, permitindo que o projetista use sua experiência para definir todos os parâmetros do controlador.

Para que seja possível compreender o projeto do controlador, é fundamental, antes de tudo, definir algumas configurações do controlador *fuzzy*.

### 3.2.2.1 Configuração do controlador *fuzzy*

Conforme visto na Seção 2.4.3, existem dois tipos principais de métodos de inferência: Mandani (baseada exclusivamente na experiência do projetista) e Takagi-Sugeno (abordagem orientada a dados). Como não existem dados úteis disponíveis para o projeto do controlador, optou-se pela utilização do método Mandani.

Com isso, e tendo em vista que toda forma de controle estará embarcada no ATmega 2560, fez-se necessária a busca por uma biblioteca disponível ao microcontrolador, que facilitaria a implementação do controlador *fuzzy*. A biblioteca aplicada nesse trabalho é a eFLL (*embedded Fuzzy Logic Library*). A eFLL foi desenvolvida pelo *Robotic Research Group* na Universidade Estadual do Piauí e está disponível no GitHub® (ALVES, 2019). Como foi escrita em C++/C e usa apenas a biblioteca padrão da linguagem C, *stdlib.h*, a eFLL é uma opção versátil, leve e eficiente de trabalhar com lógica e controle *fuzzy* em qualquer sistema embarcado que tenha seus comandos escritos na linguagem C.

Essa biblioteca usa o método de inferência Mandani e o processo de defuzzificação por centro de área, ou centroide, método esse que entrega os melhores resultados para aplicações de controle já que evita casos em que o controlador se satura mesmo com alterações na variável de entrada. Com essa biblioteca, é possível criar o controlador *fuzzy*, entradas e saídas e suas funções de pertinência e conjunto de regras *fuzzy*, além de permitir realizar todo o processo de fuzzificação e defuzzificação.

Com essa biblioteca devidamente embarcada no microcontrolador, é possível a montagem das funções de pertinência de acordo com o projeto do controlador.

### 3.2.2.2 Montagem das funções de pertinência

A variável referente à função de pertinência de entrada, chamada também de variável linguística de entrada, é o erro da distância do carro da ponte, isto é, a diferença entre a referência do movimento lateral e o valor da distância do carro lido pelo sensor de ultrassom. Como os valores das distâncias para o início e final de percurso da ponte são 4 e 80 centímetros, respectivamente, o erro da distância pode variar entre -76 a +76.

Já a variável referente à função de pertinência de saída, chamada também de variável linguística de saída, é interpretada como sendo a velocidade de movimentação do carro. De forma mais técnica, seria o valor do sinal de controle aplicado ao *drive* de acionamento do carro da ponte; quanto maior o valor desse sinal, maior a velocidade de movimentação do carro. A excursão desse sinal pode variar conforme a mudança no número de funções de pertinência, como será visto mais à frente.

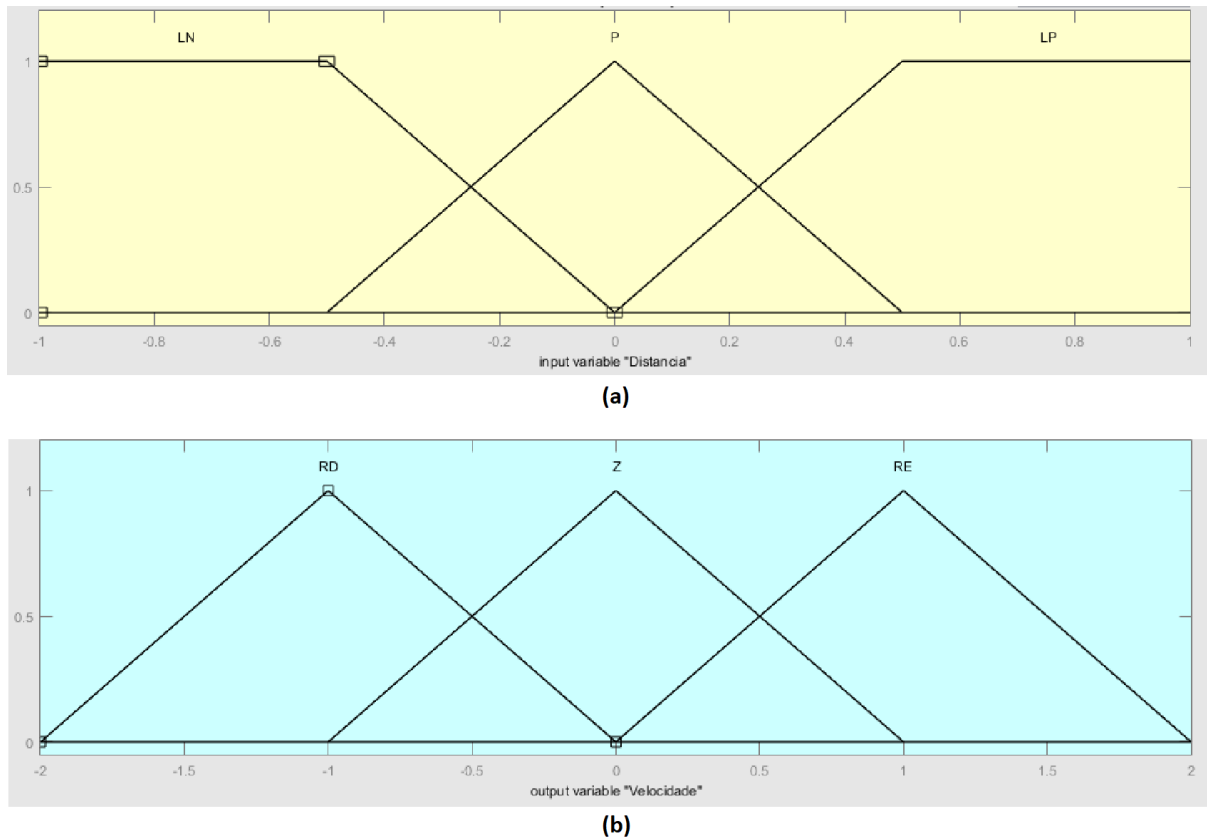
O número de conjuntos *fuzzy*, isto é, funções de pertinência, influencia diretamente no desempenho do controle. Para o caso de um número pequeno de FPs, o processamento e, conseqüentemente, a atuação do controlador serão mais velozes; contudo, o controlador pode não apresentar um resultado muito bom na correção da variável de controle, além de tornar o sistema abrupto demais. Já para o caso de muitas funções de pertinência, o controlador possivelmente corrigirá melhor a variável de controle, tendo em vista que, por conta do grande número de FPs, a resolução da variável *fuzzy* de entrada é aumentada, possibilitando uma correção mais detalhada por parte da saída; contudo, por conta da existência de muitas funções de pertinência, o processamento e, conseqüentemente, a atuação do controlador serão mais lentos, podendo prejudicar consideravelmente o desempenho do controlador, apesar do sistema utilizado nesse trabalho apresentar uma dinâmica considerada rápida.

Portanto, para se definir qual o número ideal de funções de pertinência do controlador *fuzzy*, decidiu-se fazer testes com três diferentes configurações: controlador com três, cinco e sete conjuntos *fuzzy*. Além da variação no número de FPs, admitiu-se também uma variação no ganho do movimento proporcional.

A montagem dos conjuntos *fuzzy* de entrada e saída para a configuração com três funções de pertinências é mostrada na Figura 25.

Nessa configuração, a entrada é composta pelos conjuntos LN (Longe-Negativo), P (Perto) e LP (Longe-Positivo); e a saída pelos conjuntos RD (Rápido-Direita), Z (Zero) e RE (Rápido-Esquerda). Percebe-se que esses valores linguísticos tendem a representar a natureza da variável *fuzzy* de entrada, bem como a forma de aplicação da saída.

Figura 25 – Conjuntos *fuzzy* de (a) entrada e (b) saída para a configuração com três funções de pertinência.



Fonte: Elaborado pelo autor (2019).

Para a configuração com três FPs, a entrada pode variar na faixa de valores de -1 a +1, como se pode ver na Figura 25. Com isso, a variável linguística de entrada, que é o erro da distância do carro da ponte, deverá passar por uma normalização, uma vez que ela varia entre -76 a +76. A normalização ocorre pela Equação 14:

$$erro_{NORM} = erro / 76 \quad (14).$$

Já a saída, para esse caso, varia de -1 a +1. Contudo, como a variável linguística de saída, que é a velocidade de movimentação do carro, representa o valor do sinal de controle aplicado ao *drive* de acionamento da ponte, será necessária uma nova normalização, tendo em vista que a saída de PWM do ATmega 2560, responsável pelo acionamento do *drive*, tem uma resolução de 8 bits, ou seja, assume valores de 0 a 255. A normalização ocorre pela Equação 15:

$$velocidade_{NORM} = velocidade \cdot 255 \quad (15).$$

O conjunto de base de regras para o controlador com três conjuntos *fuzzy* pode ser

visto no Quadro 1.

Quadro 1 – Base de regras para controlador com três conjuntos *fuzzy*.

Entrada	Saída
SE erro é LN	ENTÃO velocidade é RE
SE erro é P	ENTÃO velocidade é Z
SE erro é LP	ENTÃO velocidade é RD

Fonte: Elaborado pelo autor (2019).

A montagem dos conjuntos *fuzzy* de entrada e saída para a configuração com cinco funções de pertinências é mostrada na Figura 26.

Nessa configuração, a entrada é composta pelos conjuntos LN, PN (Perto-Negativo), MP (Muito Perto), PP (Perto-Positivo) e LP; e a saída pelos conjuntos RD, LD (Lento-Direita), Z, LE (Lento-Esquerda) e RE.

Para a configuração com cinco FPs, para que as funções de pertinência mantenham o mesmo tamanho de base que no exemplo anterior, a entrada pode variar na faixa de valores de -2 a +2, como se pode ver na imagem mostrada anteriormente. Com isso, a variável linguística de entrada deverá passar por uma normalização também nesse caso. A normalização ocorre pela Equação 16:

$$erro_{NORM} = 2 \cdot erro / 76 \quad (16).$$

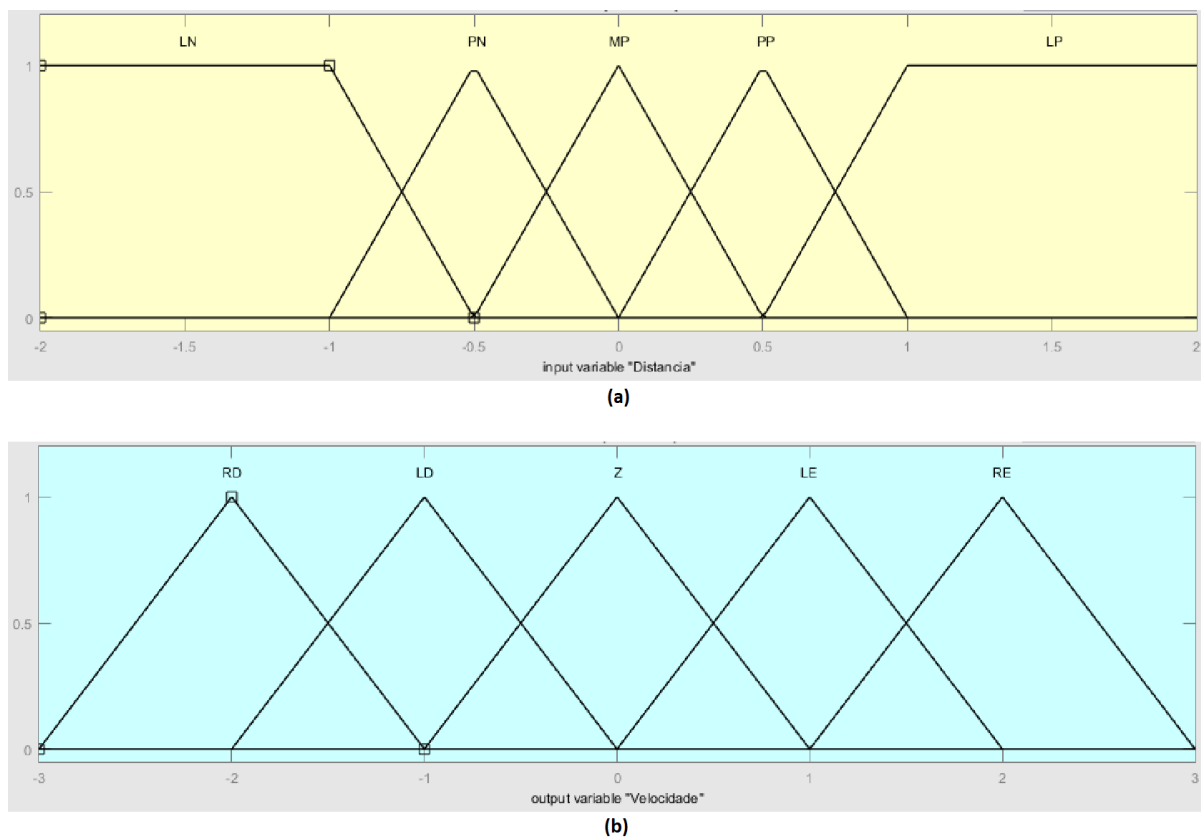
Já a saída, para esse caso, varia de -2 a +2. Portanto, nesse caso também será necessária uma normalização. A normalização ocorre pela Equação 17:

$$velocidade_{NORM} = velocidade \cdot 255 / 2 \quad (17).$$

O conjunto de base de regras para o controlador com cinco conjuntos *fuzzy* pode ser visto no Quadro 2.



Figura 26 – Conjuntos *fuzzy* de (a) entrada e (b) saída para a configuração com cinco funções de pertinência.



Fonte: Elaborado pelo autor (2019).

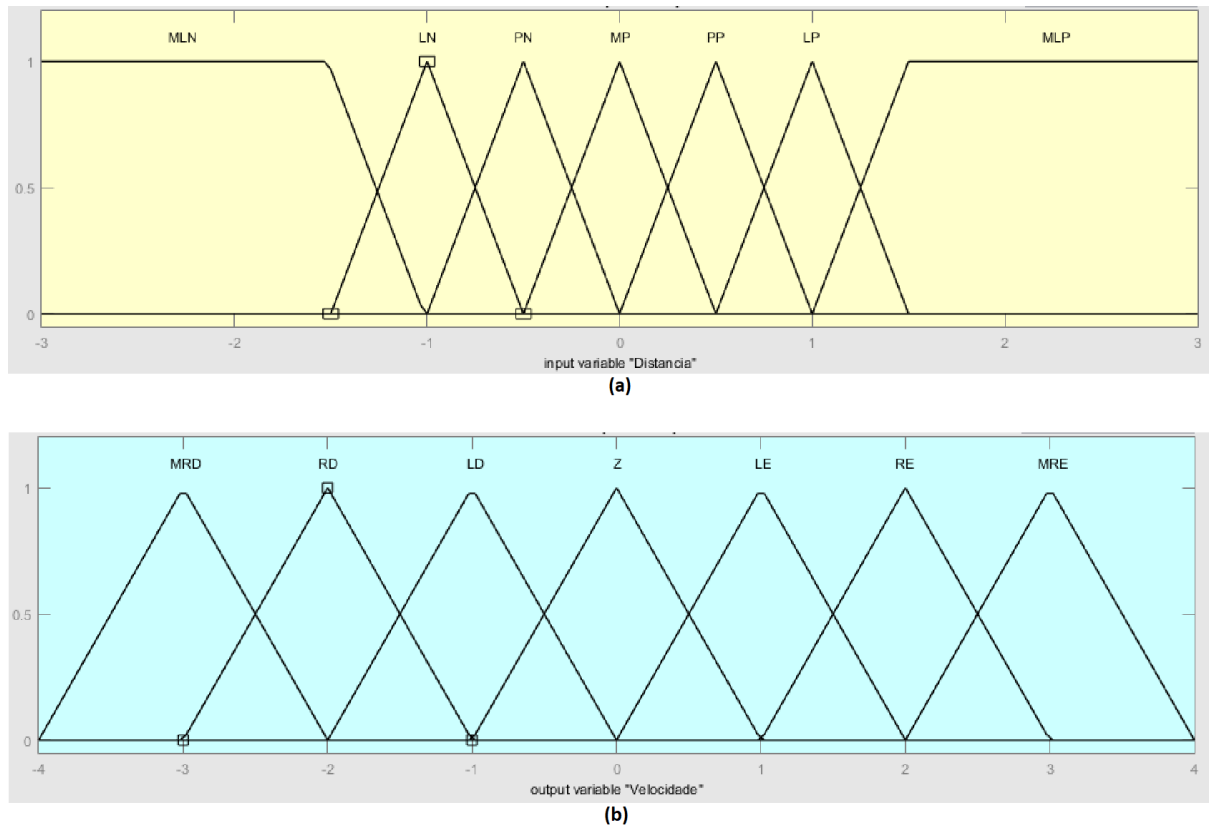
Quadro 2 – Base de regras para controlador com cinco conjuntos *fuzzy*.

Entrada	Saída
SE erro é LN	ENTÃO velocidade é RE
SE erro é PN	ENTÃO velocidade é LE
SE erro é MP	ENTÃO velocidade é Z
SE erro é PP	ENTÃO velocidade é LD
SE erro é LP	ENTÃO velocidade é RD

Fonte: Elaborado pelo autor (2019).

A montagem dos conjuntos *fuzzy* de entrada e saída para a configuração com sete funções de pertinências é mostrada na Figura 27.

Figura 27 – Conjuntos *fuzzy* de (a) entrada e (b) saída para a configuração com sete funções de pertinência.



Fonte: Elaborado pelo autor (2019).

Nessa configuração, a entrada é composta pelos conjuntos MLN (Muito Longe-Negativo), LN, PN, MP, PP, LP e MLP (Muito Longe-Positivo); e a saída pelos conjuntos MRD (Muito Rápido-Direita), RD, LD, Z, LE, RE e MRE (Muito Rápido-Esquerda).

Para a configuração com sete FPs, para que as funções de pertinência mantenham o mesmo tamanho de base que no exemplo anterior, a entrada pode variar na faixa de valores de -3 a +3, como se pode ver na imagem mostrada anteriormente. Com isso, a variável linguística de entrada deverá passar por uma normalização também nesse caso. A normalização ocorre pela Equação 18:

$$erro_{NORM} = 3 \cdot erro / 76 \quad (18).$$

Já a saída, para esse caso, varia de -3 a +3. Portanto, nesse caso também será necessária uma normalização. A normalização ocorre pela Equação 19:

$$velocidade_{NORM} = velocidade \cdot 255 / 3 \quad (19).$$

O conjunto de base de regras para o controlador com sete conjuntos *fuzzy* pode ser visto no Quadro 3.

Quadro 3 – Base de regras para controlador com sete conjuntos *fuzzy*.

<b>Entrada</b>	<b>Saída</b>
SE erro é MLN	ENTÃO velocidade é MRE
SE erro é LN	ENTÃO velocidade é RE
SE erro é PN	ENTÃO velocidade é LE
SE erro é MP	ENTÃO velocidade é Z
SE erro é PP	ENTÃO velocidade é LD
SE erro é LP	ENTÃO velocidade é RD
SE erro é MLP	ENTÃO velocidade é MRD

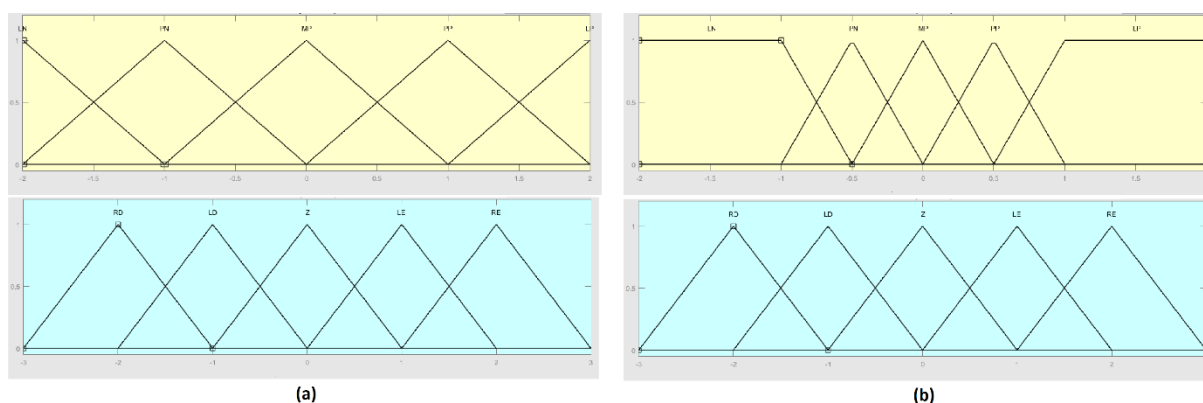
Fonte: Elaborado pelo autor (2019).

Como o controlador *fuzzy* aplicado é do tipo proporcional, deve-se admitir mudanças no ganho. Isso irá refletir diretamente nas funções de pertinência do controlador *fuzzy*, como será visto na próxima seção.

### 3.2.2.3 Ganho proporcional

O ganho proporcional do controlador *fuzzy* do tipo Mandani está diretamente ligado à forma como as funções de pertinência são montadas. Como foi visto na Seção 2.4.3.1, o ganho proporcional se dá pela razão entre as larguras das bases de saída e entrada. Portanto, caso se queira aumentar o ganho proporcional do controlador, as larguras das bases das FPs de entrada deverão ser diminuídas. Na Figura 28, podemos ver a diferença entre os controladores com ganho unitário e ganho 2.

Figura 28 – Comparação dos conjuntos *fuzzy* de entrada e saída para controladores com (a) ganho proporcional unitário e (b) ganho proporcional 2.



Fonte: Elaborado pelo autor (2019).

A principal diferença entre os dois casos mostrados anteriormente está no tamanho das bases das FPs de entrada: para o caso do controlador com ganho proporcional unitário, o tamanho das bases das FPs de entrada é igual ao das bases das FPs de saída, isto é, tamanho 2; já para o caso do controlador com ganho proporcional 2, o tamanho das bases das FPs de entrada diminui para 1, proporcionando um aumento no ganho. Percebe-se que, para ambos os casos, os conjuntos *fuzzy* de saída permanecem inalterados.

Como pôde ser visto nos exemplos anteriores, os conjuntos *fuzzy* tem como base funções de pertinência triangulares, muito utilizadas em aplicações de controle por evitar saturação do controlador e, assim, possibilitar o controle efetivo do sistema em qualquer região. A única exceção é com relação aos conjuntos dos extremos da variável de entrada para casos com ganhos superiores à unidade, em que é usada uma FP trapezoidal. A FP trapezoidal proporciona 100% de pertinência da variável de entrada a um determinado conjunto, o que manterá a saída em um único valor apesar das mudanças na entrada, podendo saturar o controlador. O motivo pelo qual se decidiu pela utilização da FP trapezoidal nos extremos é que, nesses casos, a magnitude do erro da distância é grande, isto é, o carro está muito longe do seu *setpoint*; com isso, para agilizar a movimentação do carro, é aceitável que o controlador assuma seu valor máximo de atuação enquanto o valor da variável *fuzzy* de entrada estiver nos seus extremos. A duração da ocorrência dessa condição dependerá do valor do ganho do controlador. Quanto maior o ganho, mais bruscas serão as mudanças entre conjuntos adjacentes, além de elas ocorrerem mais próximo

ao *setpoint*; com isso, as FPs trapezoidais terão uma extensão maior, mantendo o controlador em seu valor máximo de saída por mais tempo, caracterizando também o aumento de ganho.

Testes com diferentes configurações de valores de ganho e números de conjuntos *fuzzy* foram realizados a fim de se definir a configuração ideal. Os resultados são mostrados na Seção 4.1.

Tendo o controle *fuzzy* do movimento lateral sido finalizado, é possível avançar para o projeto do controle PD do balanço da carga.

### 3.2.3 Controle PD antibalanço da carga

Optou-se por trabalhar com a sintonia dos parâmetros do controlador via Ziegler-Nichols. O motivo para tal escolha se dá no desejo de evitar trabalhar com modelos matemáticos do sistema, os quais trariam em si alta complexidade e não necessariamente melhorariam os resultados, tendo em vista que muitas simplificações e ajustes seriam necessários a fim de se obter um resultado plausível. Além disso, a escolha do método de sintonia Ziegler-Nichols foi contribuída por outros fatores como boa conceituação do método e a possibilidade de se obter os parâmetros do controlador por meio de experimentos sobre o sistema em si.

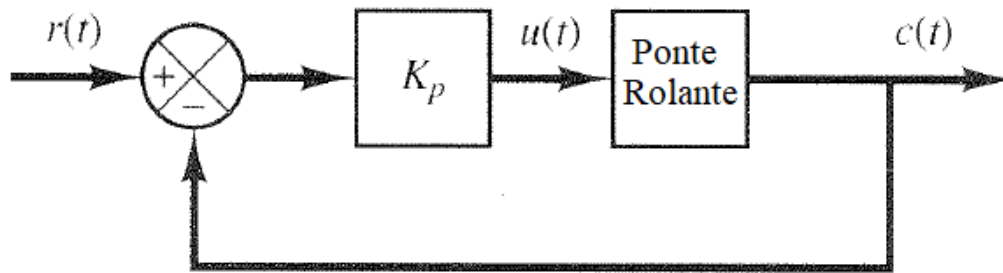
Então, o projeto do controlador PD antibalanço da carga teve como base o segundo método de sintonia Ziegler-Nichols. O primeiro método não pôde ser utilizado já que o sistema apresenta integradores, conforme demonstrado por Baiôco (2012) em sua modelagem do mesmo protótipo de ponte rolante didática utilizado neste trabalho.

#### 3.2.3.1 Aplicação do segundo método de sintonia Ziegler-Nichols

Para se definir os valores das constantes ideais para o controlador PD, utilizou-se do segundo método de sintonia Ziegler-Nichols como um referencial. Isso porque o método de Ziegler-Nichols não fornece valores exatos dos ganhos para o controlador, mas sim sugestões de valores iniciais a partir dos quais se chegará aos valores ideais para o sistema após ajustes em testes práticos.

Portanto, para a aplicação do segundo método de sintonia Ziegler-Nichols, fechou-se a malha do sistema e ajustou-se os valores de ganho integral e derivativo para zero, ficando-se apenas com a ação proporcional, conforme demonstrado na Figura 29.

Figura 29 – Sistema malha fechada para ensaio de sintonia pelo segundo método de Ziegler-Nichols.



Fonte: Adaptado de Ogata (2001, p. 546)

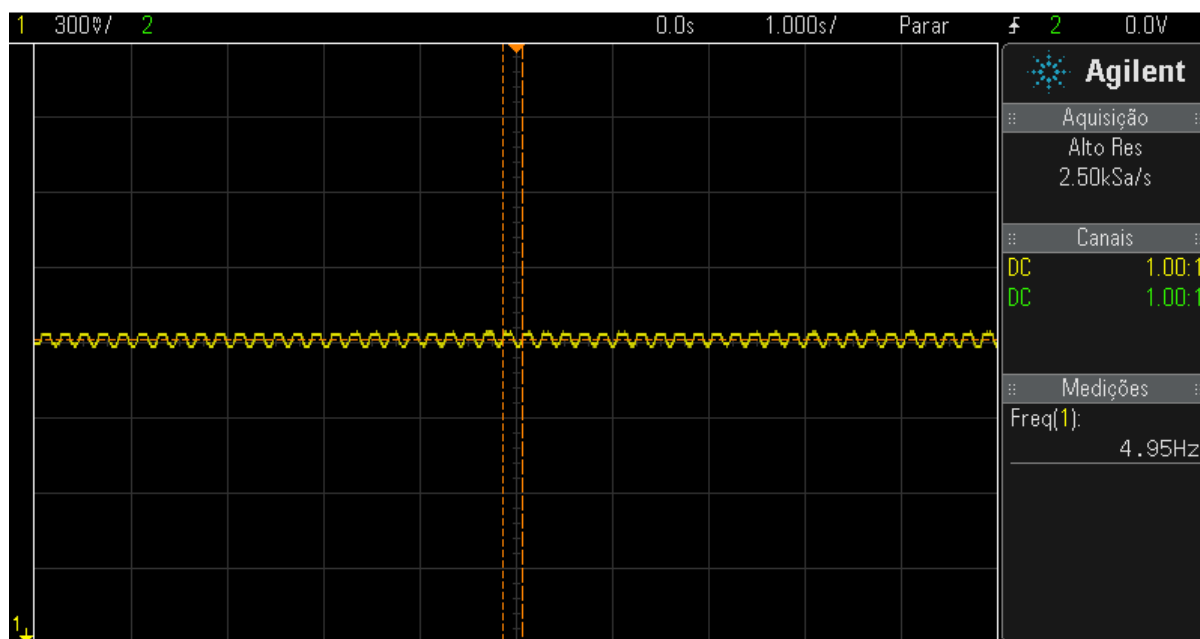
Então, o ganho proporcional  $K_p$  começou a ser aumentado até que atingisse o valor de ganho proporcional crítico  $K_{cr}$ , em que o sistema apresenta oscilações mantidas. O  $K_{cr}$  encontrado para esse caso foi de 37. Na Figura 30 pode-se observar as oscilações críticas apresentadas pelo sistema, cujo período crítico de oscilação  $P_{cr}$  é igual ao inverso da frequência crítica de oscilação  $f_{cr}$ , isto é,

$$P_{cr} = 1/f_{cr} = 1/4,95\text{Hz} = 202,02\text{ ms} \quad (20).$$

Com isso, utilizando a Tabela 2, pôde-se obter os valores das constantes sugeridas pelo método Ziegler-Nichols para o controlador PD em questão, isto é, um  $K_p$  de 22,2 e um  $T_d$  de 0,0253 segundos, resultando em um  $K_d$  de 0,561.

Optou-se por não utilizar a ação integral neste trabalho. O motivo é que, quando inserida, a ação integral deixava o sistema muito instável, como foi constatado em alguns testes preliminares. Com isso, uma adequação da ação integral a este trabalho não necessariamente seria uma garantia de melhora no desempenho do controle. Além disso, não há a necessidade de eliminação de erro em regime permanente para esse sistema porque, como a variável controlada é um pêndulo simples e seu *setpoint* é a sua posição de descanso, que é na vertical, a eliminação de erro em regime permanente é realizada, de certa forma, pela própria ação da gravidade sobre o pêndulo.

Figura 30 – Oscilações mantidas com  $f_{cr}$  de 4,95 Hz em ensaio de sintonia Ziegler-Nichols.



Fonte: Elaborado pelo autor (2019).

Um outro ponto importante de se elucidar é a diferença entre as abordagens por modelo matemático e por Ziegler-Nichols experimental, visto na próxima seção.

### 3.2.3.2 Comparação com abordagem por modelo matemático

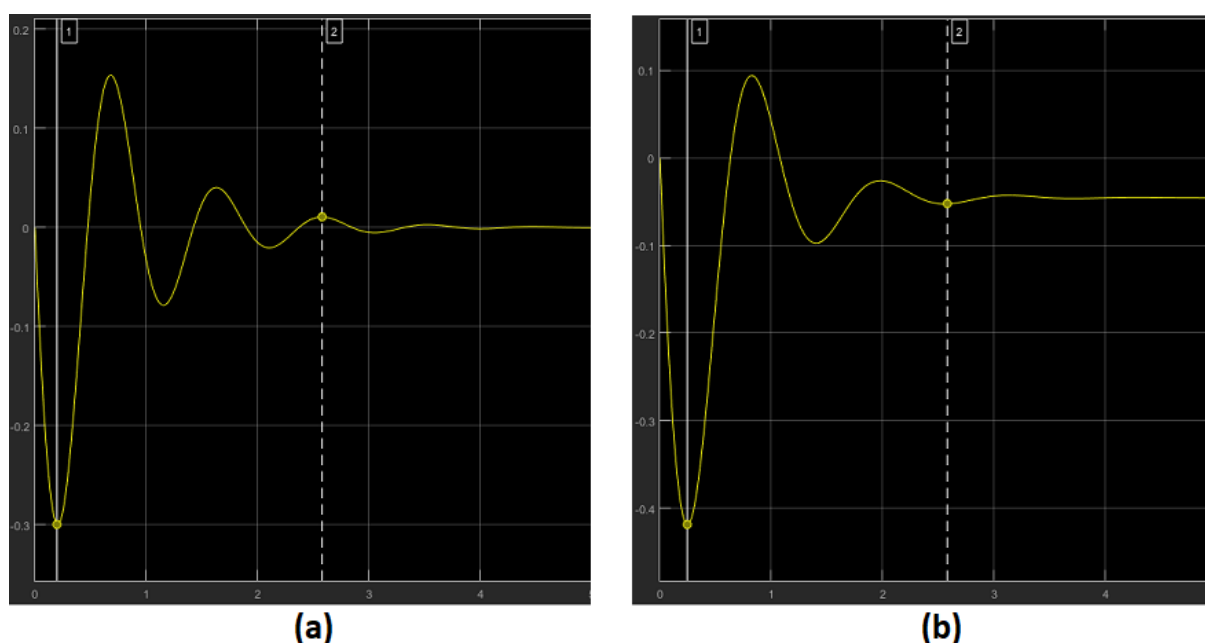
Em seu trabalho, Baiôco (2012) formula um modelo matemático do sistema e, em seguida, utilizando o critério de estabilidade de Routh, encontra os valores de  $K_{cr}$  e  $P_{cr}$  teóricos de 0,8063 e 0,9 segundos, respectivamente, resultando, pelo método de Ziegler-Nichols, em um  $K_p$  de 0,4838 e um  $K_d$  de 0,0538. Percebe-se que esses valores são bem diferentes dos encontrados neste trabalho, cuja abordagem é via Ziegler-Nichols experimental, sem modelos matemáticos.

Contudo, o próprio autor afirma que alterações foram necessárias nesses valores, principalmente no ganho proporcional, já que o tempo de acomodação do sistema na simulação estava muito grande. Isso ocorreu porque, como dito anteriormente, além do método Ziegler-Nichols fornecer apenas uma sugestão de valores iniciais para as constantes do controlador, um modelo matemático é cercado de aproximações, gerando incertezas nos resultados. Após alguns testes, o  $K_p$  foi alterado para 40, valor

bem diferente do encontrado inicialmente de forma teórica e mais próximo do encontrado neste trabalho, que é de 22,2.

Por fins de demonstração, simulou-se a resposta ao degrau do sistema para os ganhos encontrados neste trabalho, utilizando o modelo matemático fornecido no trabalho de Baiôco (2012). Então, comparou-se o resultado com os ganhos ajustados encontrados no trabalho de Baiôco (2012), como visto na Figura 31.

Figura 31 – Comparação entre as simulações da resposta ao degrau do sistema para (a) ganhos encontrados neste trabalho e (b) ganhos ajustados encontrados por Baiôco (2012) em seu trabalho.



Fonte: Elaborado pelo autor (2019).

Analisando a Figura 31, é possível observar que os resultados estão próximos, sendo a resposta obtida por Baiôco (2012) um pouco menos oscilatória. Isso indica que algumas mudanças podem ser realizadas nas constantes encontradas anteriormente a fim de reduzir as oscilações e o tempo de acomodação.

Para a definição das constantes ideais para o controlador PD, alguns testes deveriam ser realizados com diferentes valores de ganhos; contudo, antes da realização desses testes, a leitura do potenciômetro do pêndulo da carga, que é a variável de controle para o antibalço, deveria passar por alguns condicionamentos, como será visto na próxima seção.



### 3.2.3.3 Condicionamento da leitura do potenciômetro da carga

Ao se realizar alguns testes práticos com o controlador PD na ponte rolante, constatou-se uma absoluta dificuldade do controlador em estabilizar o sistema, seja qual fosse os valores dos ganhos. Isso estava sendo causado pela forma como a variável de controle estava sendo tratada. Para entender melhor essa questão, faz-se necessário antes uma explicação sobre a relação entre a variação dos ângulos da carga e os valores de tensão do potenciômetro ao se movimentar o pêndulo.

Com a ajuda de um osciloscópio, associou-se a leitura de alguns valores de ângulos da carga aos seus respectivos valores de tensão. O objetivo dessa abordagem seria saber quantos graus de variação no ângulo da carga correspondem a um degrau de resolução da conversão A/D de leitura do potenciômetro por parte do microcontrolador. Obteve-se a medida para os ângulos: 30°, 60°, 90°, 120° e 150°. Para se ter um bom grau de certeza, foram realizadas três medições desse tipo, obtendo-se a média dessas medições ao final. O resultado é visto na Tabela 5.

Tabela 5 – Relação entre leitura dos ângulos e valores de tensão da carga.

Ângulo (°)	Tensão média (V)
30	3,32
60	2,84
90	2,31
120	1,88
150	1,39

Fonte: Elaborado pelo autor (2019).

Com isso, como os intervalos entre ângulos são de mesmo tamanho (30°), para cada intervalo obteve-se a relação de graus por Volt. Ao fim, tirou-se a média de todas essas relações, nos dando uma resolução de, aproximadamente, 0,0625°/mV. Como a resolução da conversão A/D da leitura do potenciômetro na entrada analógica de 10 bits do microcontrolador é de  $5V/2^{10} = 4,883\text{ mV}$ , um degrau de variação na

conversão A/D do valor lido do potenciômetro da carga corresponde a uma variação de  $0,0625^\circ/\text{mV} \cdot 4,833 \text{ mV} = 0,302^\circ$ .

De volta à análise da forma de tratamento da variável de controle, o que se estava fazendo era admitir um valor único de *setpoint*, sem nenhuma margem para variações positivas e negativas. Contudo, percebeu-se que o sistema tinha severas dificuldades em se estabilizar nessa condição porque, como não existia margem para os valores de *setpoint* lidos, uma variação tão diminuta de  $0,302^\circ$ , para mais ou para menos, já implicava em uma ação corretiva do controlador; essa ação era forte o suficiente para balançar a carga de um lado a outro gerando uma mínima variação angular, o que mantinha o erro não nulo e o controlador sempre em ação corretiva. Essa condição perdurava por muito tempo até que o controlador conseguisse encontrar o equilíbrio, principalmente se ele tivesse que corrigir uma grande variação de ângulo da carga. Além disso, ainda que o sistema estivesse estabilizado, uma pequena flutuação na tensão da rede ou até mesmo um erro na conversão A/D por parte do microcontrolador seria o bastante para mudar o valor lido do potenciômetro em um degrau, gerando erro na variável de controle e desestabilizando o sistema novamente.

Tendo em vista essa dificuldade, optou-se por admitir que o *setpoint* assumisse uma margem de valores a partir de um referencial, criando uma espécie de zona de *setpoint*. Essa solução resolveria o problema da desestabilização do sistema, possibilitando um controle viável. Alguns valores foram testados e aceitou-se uma margem no valor lido de 10 valores de conversão A/D para ambos os sentidos. Ou seja, ao invés do *setpoint* ser representado por um valor único decorrente da conversão A/D da leitura do potenciômetro (por exemplo, 475), ele seria representado por uma margem de  $\pm 10$  valores considerando um referencial (por exemplo, de 465 até 485). Isso quer dizer que a carga poderia sofrer uma variação angular de, aproximadamente,  $3^\circ$  em ambos os sentidos e o controlador não entraria em ação. Essa variação é plenamente aceitável por ser bem pequena, não causando danos à carga caso ela se movimente nessa zona de *setpoint* (AVILA et.al., 2008).

Com esses ajustes, é possível realizar os testes necessários para a definição dos valores ideais de constantes para o controlador PD antibalanco da carga. Vários testes foram realizados com diversos valores de ganhos a fim de definir o melhor valor. Os resultados são mostrados na Seção 4.2.

Esse condicionamento da leitura do potenciômetro é realizado no programa embarcado no microcontrolador, responsável por integrar toda a parte de controle do sistema.

### 3.2.4 Integração dos controles no microcontrolador

As duas formas de controle apresentadas são integradas em um único sistema embarcado. Com isso, o controlador embarcado é responsável por comutar entre duas formas de controle: *fuzzy* para o movimento lateral, caso a carga esteja estável, e PD para antibalanco da carga. Ou seja, se a carga balançar, o controlador se ocupará por remover o balanço; com a carga de volta à estabilidade, ele pode também voltar a controlar o carro da ponte rolante no seu movimento lateral.

O sensor responsável por indicar o balanço da carga é um potenciômetro, como já dito. Já o sensor responsável por indicar a posição do carro em seu percurso é um sensor ultrassom HC-SR04. Esse sensor irá emitir pulsos ultrassônicos e, de acordo com a diferença de tempo entre os pulsos enviados e recebidos, é possível saber qual a posição do carro. A leitura desses sensores por parte do microcontrolador sofre uma filtragem do tipo média móvel. Essa filtragem foi necessária porque estavam sendo identificadas muitas leituras erradas e valores espúrios, inviabilizando o controle.

O programa embarcado no microcontrolador apresenta também rotinas de comunicação serial, permitindo o recebimento de pacotes vindos da interface, identificando-os e extraíndo os dados úteis, além de realizar a verificação da integridade do pacote por meio do *checksum*.

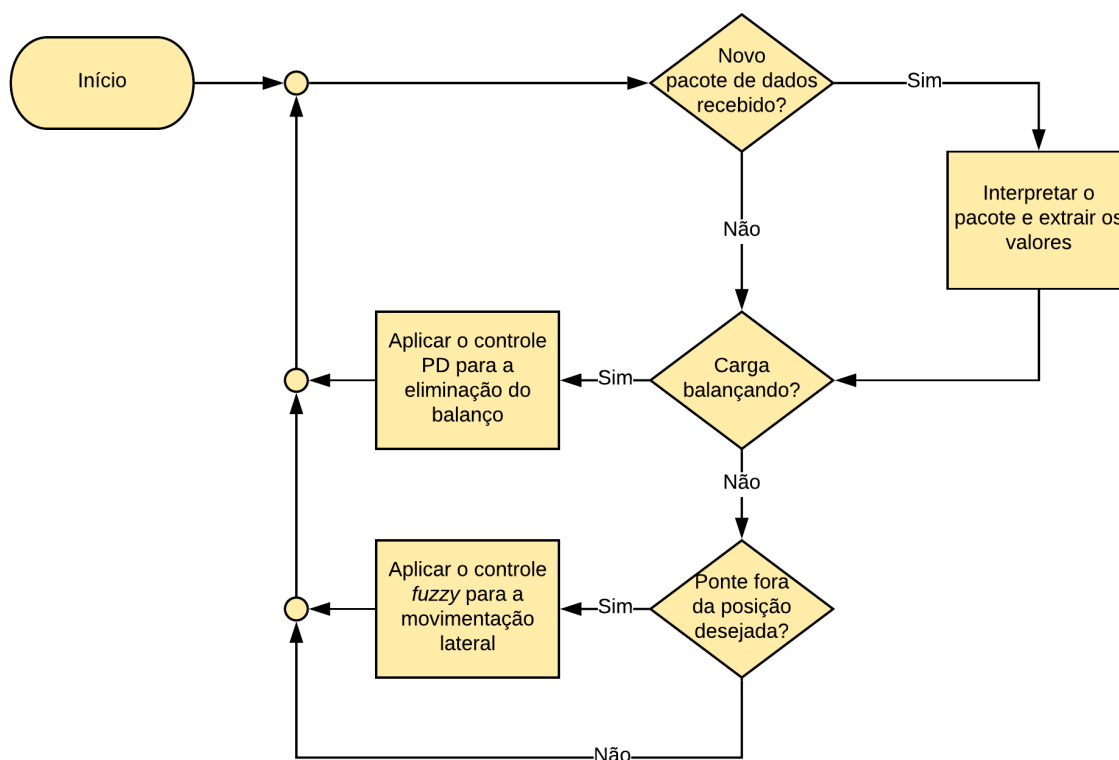
A Figura 32 mostra um fluxograma que resume o processo de controle embarcado no microcontrolador.

## 3.3 APLICAÇÃO DE UM MÉTODO DE ACIONAMENTO A PARTIR DE COMANDOS DE VOZ

Esse trabalho implementa um método de acionamento a partir de comandos de voz cujo foco está na utilização dos coeficientes MFCC. Conforme mencionado na fundamentação teórica, a escolha dos coeficientes cepstrais é devido à sua ampla

utilização e boa conceituação em projetos de reconhecimento de voz apresentados na literatura.

Figura 32 – Fluxograma do processo de controle embarcado ao microcontrolador.

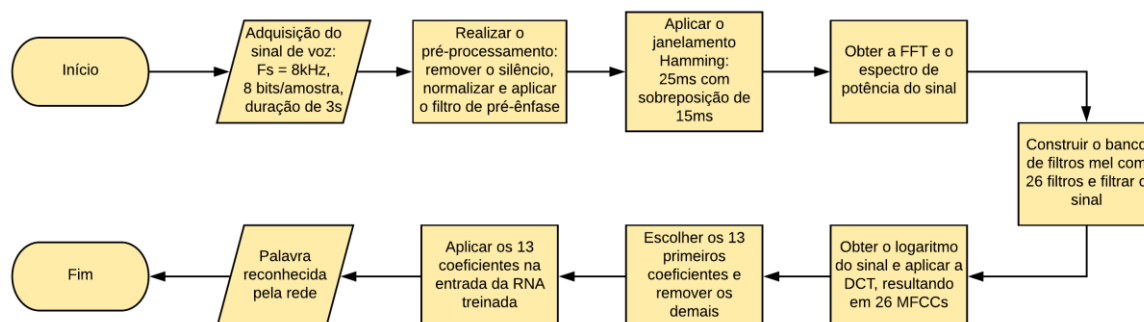


Fonte: Elaborado pelo autor (2019).

Além disso, vale ressaltar que o sistema RAV implementado neste trabalho é do tipo dependente de locutor, ou seja, o funcionamento é garantido para um único locutor, que no caso é aquele que forneceu amostras de sua própria voz. Caso seja necessário garantir o funcionamento para outro locutor, dever-se-ão realizar a coleta de novas amostras e novo treinamento. O motivo dessa escolha foi o pouco tempo disponível para a implementação de um sistema independente de locutor; para esse sistema, diversas amostras deveriam ser colhidas de várias pessoas de diferentes sexos e idades.

A Figura 33 apresenta um panorama do processo de reconhecimento de voz utilizado neste trabalho. Nas próximas seções, cada uma dessas etapas será abordada de forma mais aprofundada.

Figura 33 – Fluxograma do processo de reconhecimento de voz utilizado.



Fonte: Elaborado pelo autor (2019).

A implementação do sistema RAV responsável pelo acionamento por comando de voz do carro da ponte inicia-se pela aquisição do sinal de voz e seu pré-processamento.

### 3.3.1 Aquisição do sinal de voz e pré-processamento

A aquisição do sinal de voz foi feita via MatLab®. Para cada cor, realizou-se 100 gravações de áudio de três segundos cada; portanto, como quatro cores foram utilizadas no trabalho, o número total de gravações de áudio foi de 400. O número de gravações foi igualmente dividido em áudios com diferentes velocidades de fala: rápida, normal e lenta; o motivo é aumentar a robustez do sistema, de forma que ele possa fazer o reconhecimento independentemente da velocidade da fala do locutor. Os áudios foram gravados com um microfone e convertidos digitalmente a áudios “.wav” mono com 8 bits por amostra e uma taxa de amostragem de 8kHz, valor bem utilizado para aplicações desse tipo (MARTINS, 2014; THIAGO, 2017). Tendo os áudios na sua forma digital, é possível manipulá-los com o MatLab® de forma simples, já que eles serão representados como matrizes.

O próximo passo foi a remoção de alguns blocos de silêncio. Para essa etapa, dividiu-se o sinal de voz em blocos de tamanhos iguais e removeu-se os blocos com energia menor que 1% da máxima. Isso reduz drasticamente o tamanho do sinal de voz, tornando o processamento mais rápido. Por exemplo, para uma gravação da palavra “amarelo”, o sinal de voz original, que continha 24.000 amostras, foi reduzido para um sinal de voz com 3.400 amostras, uma redução de, aproximadamente, 85%. Além disso, como esses blocos removidos representam períodos de silêncio no áudio,

ficarão apenas com os dados que contém de fato alguma informação e, portanto, úteis no reconhecimento de voz.

Tendo esses blocos sido removidos, o sinal de voz foi normalizado para valores de -1 a 1. Isso é importante porque a intensidade do áudio pode variar para cada gravação. Portanto, o processo de normalização irá sanar esse problema, restringindo esses valores de intensidade de todos os áudios a uma mesma faixa.

Então, com o sinal de voz normalizado, o filtro de pré-ênfase é aplicado a fim de equalizar o espectro. O coeficiente de pré-ênfase  $\alpha_{pre}$  utilizado foi de 0,97 por ser um valor bem utilizado em sistemas RAV para se obter bons resultados de reconhecimento de voz. A etapa seguinte é a análise espectral.

### 3.3.2 Análise espectral

Nesta etapa, iniciou-se pelo janelamento do sinal. Para isso, dividiu-se o sinal em janelas de 25 milissegundos cada com sobreposição de 15 milissegundos, valores esses dentro do recomendado por Rabiner e Juang (1993), sendo aplicada a janela de Hamming a fim de suavizar os efeitos das transições entre janelas adjacentes.

Então, é realizada a FFT e obtido o espectro de magnitude de cada janela. Com esse valor, é possível obter o espectro de potência para cada janela, de acordo com a Equação 21:

$$P(i) = \frac{M(i)^2}{n_{FFT}} \quad (21),$$

em que  $M(i)$  é o espectro de magnitude de uma dada janela  $i$ ,  $P(i)$  é o espectro de potência dessa mesma janela e  $n_{FFT}$  é o número de pontos da FFT, definido como 512 por ser um valor comum nesse tipo de aplicação, além de ser maior que o número de amostras de cada quadro, o que é requerido (LYONS, 2013).

O espectro de potência de cada janela é fundamental na etapa de extração de características, abordada na próxima seção.

### 3.3.3 Extração de características

Nessa etapa, o primeiro passo é a construção do banco de filtros triangulares mel. Como o recomendado é que o banco possua um número de filtros de 24 a 40, decidiu-

se por usar 26 filtros, valor padrão em algumas aplicações (HUANG; ACERO; HON, 2001; LYONS, 2013). Então, com o banco de filtros construído, realizou-se a filtragem de cada janela do sinal de voz, resultando em uma matriz com o número de linhas igual ao número de filtros, isto é, 26, e o número de colunas igual ao número de janelas, valor esse que varia para cada sinal de voz devido à sua característica, tendo em vista que o número de blocos de silêncio removidos na etapa de pré-processamento pode variar para cada caso.

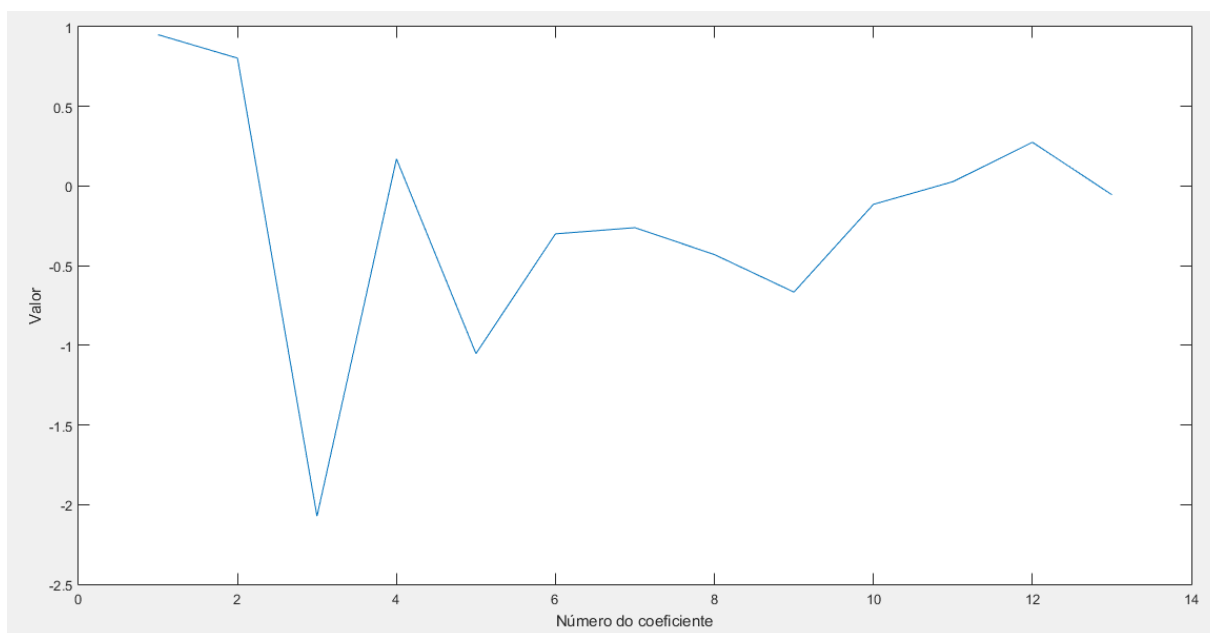
Com o sinal filtrado, a DCT foi aplicada ao logaritmo do sinal, entregando 26 coeficientes mel-cepstrais para cada janela. Então, descartou-se os coeficientes de ordem superior, ficando-se apenas com os 13 primeiros, número aproximado de MFCCs bem utilizado em muitos casos encontrados (SIDDHANT; CHEERAN, 2014; BENBA; JILBAB; HAMMOUCH, 2015). Dentre os coeficientes escolhidos, não está incluso o coeficiente de ordem zero, que representa o componente de energia do sinal, não utilizado nessa aplicação.

Portanto, tem-se um sinal representado agora por uma matriz cujo número de linhas é igual ao número de MFCCs, isto é, 13, e o número de colunas é igual ao número de janelas. Contudo, seria problemático, principalmente no momento de treinar a rede neural responsável pelo reconhecimento, ter o número de colunas variando para cada sinal. Então, para facilitar a análise e, posteriormente, o treinamento da rede neural, obteve-se a média de cada coeficiente para todas as janelas. Com isso, o resultado final seria um vetor com 13 coeficientes que seriam capazes de caracterizar um sinal de voz como um todo.

A Figura 34 mostra um exemplo de um sinal de voz correspondente à palavra “amarelo” caracterizado pelos seus 13 coeficientes mel-cepstrais. Já a Figura 35 mostra os MFCCs correspondentes a todas as gravações relativas à palavra “amarelo”, sendo diferenciados no gráfico por cores de linhas diferentes. É interessante observar nessa figura que, embora alguns conjuntos de coeficientes apresentem diferentes magnitudes, eles seguem uma mesma forma, o que serve para caracterizar e possibilitar a classificação do sinal de voz. A Figura 36 ilustra como que os MFCCs são úteis nesse processo de caracterização de um determinado sinal de voz: nela são comparados os coeficientes mel-cepstrais para sinais de voz correspondentes a duas palavras distintas, “amarelo” e “verde”. Percebe-se que, claramente, os coeficientes apresentam formas distintas para as duas palavras, confirmando que seriam possíveis o treinamento e a utilização de uma Rede Neural

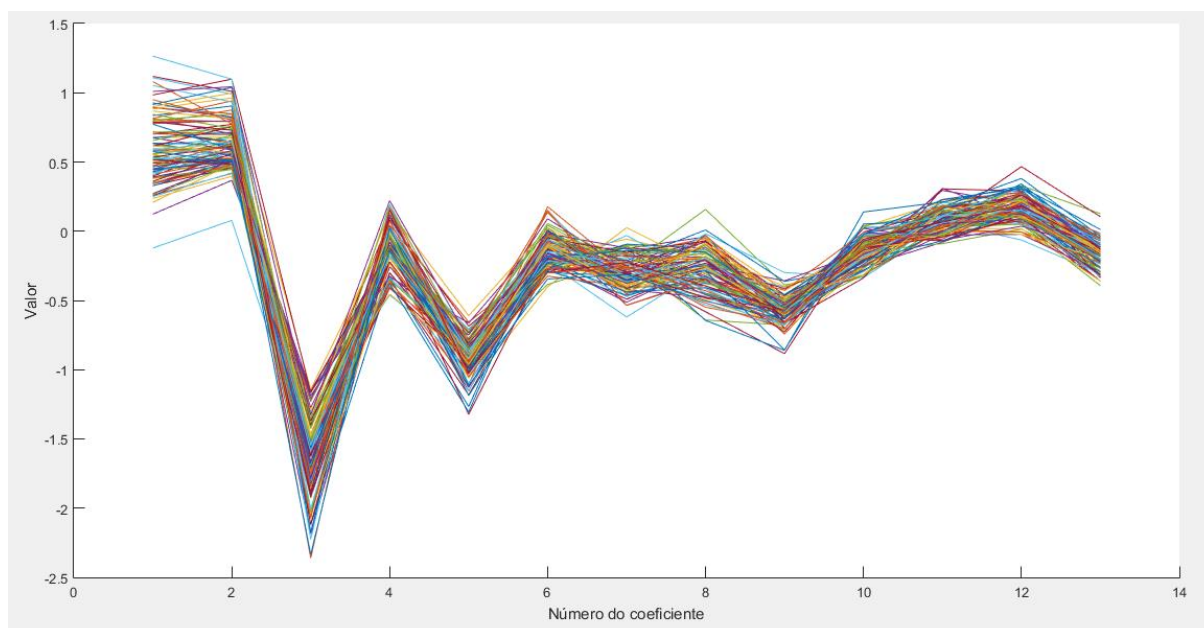
Artificial a fim de realizar a classificação de um determinado sinal de voz. Essa ideia é reforçada através da Figura 37, que apresenta a comparação dos coeficientes mel-cepstrais para os sinais de voz correspondentes às palavras “amarelo” e “verde”.

Figura 34 – Exemplo de MFCCs para o sinal de voz correspondente à palavra "amarelo".



Fonte: Elaborado pelo autor (2019).

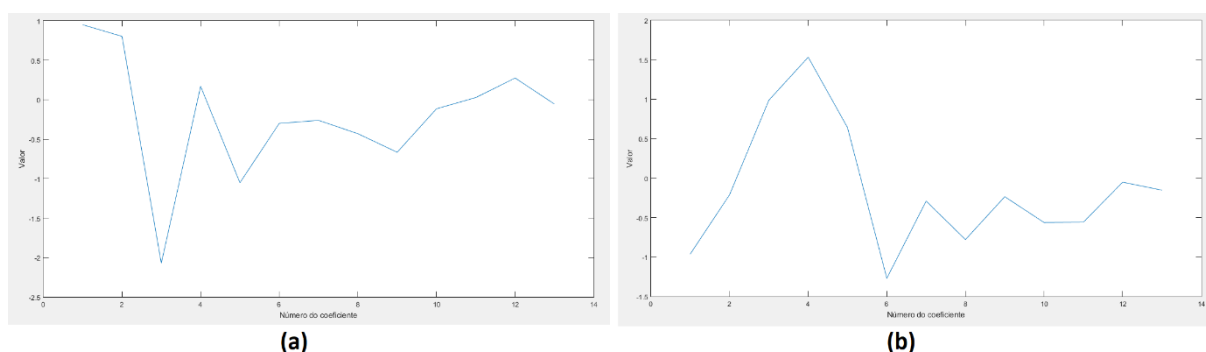
Figura 35 – MFCCs para todas as 100 diferentes gravações correspondentes à palavra "amarelo".



Fonte: Elaborado pelo autor (2019).

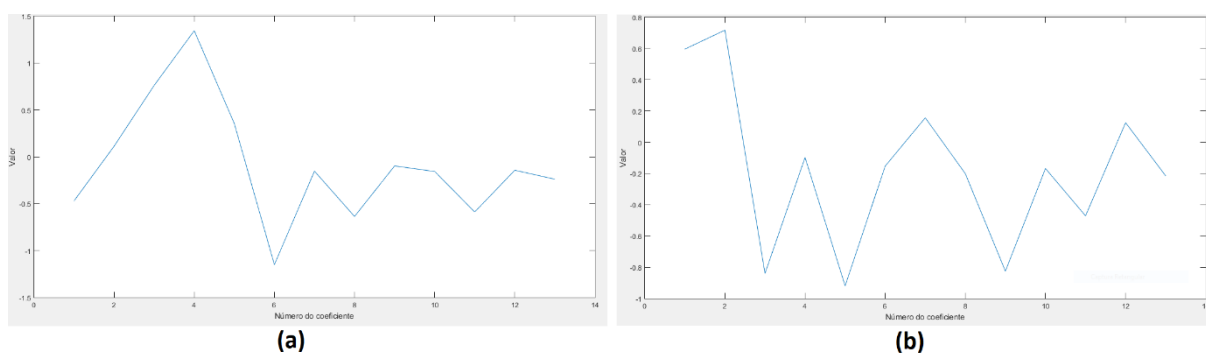


Figura 36 – Comparação de MFCCs para sinais de voz correspondentes às palavras (a) "amarelo" e (b) "verde".



Fonte: Elaborado pelo autor (2019).

Figura 37 – Comparação de MFCCs para sinais de voz correspondentes às palavras (a) "vermelho" e (b) "azul".



Fonte: Elaborado pelo autor (2019).

### 3.3.4 Classificação por Rede Neural Artificial

Para a criação e treinamento da RNA utilizada como classificador neste trabalho, utilizou-se a *toolbox* do MatLab® para redes neurais, *Neural Network Start*, ou *nnstart*. Nessa ferramenta, é possível criar redes neurais para vários fins, com suas diversas topologias. Para o sistema RAV desenvolvido neste trabalho, utilizou-se da seção de reconhecimento de padrões e classificação do *nnstart*.

Antes que a rede fosse efetivamente criada e treinada, foi necessário que as amostras de teste fossem apresentadas de forma aleatória à rede. Conforme explicado por Desai et al. (2010) em seu trabalho, a não-aleatoriedade das amostras de entrada pode tornar a rede tendenciosa para uma categoria de saída devido à ordem

sequencial em que as amostras são apresentadas à RNA. Portanto, para evitar esse problema, as amostras são aleatorizadas antes de serem entregues à rede neural.

A rede criada é do tipo *perceptron* multicamadas com uma camada oculta, função de ativação sigmoide e algoritmo de treinamento *backpropagation*. A rede possui 13 neurônios na camada de entrada, referentes aos 13 coeficientes mel-cepstrais característicos de cada sinal, e 4 neurônios na camada de saída, referentes às quatro cores possíveis de serem identificadas (vermelho, verde, azul e amarelo); a classificação do sinal de voz se dá pela cor representada pelo neurônio com maior valor na saída.

Uma grande questão em um problema que envolve redes neurais com camadas ocultas é quantos neurônios devem existir nessa camada. Foram testadas quatro topologias com diferentes números de neurônios na camada oculta, sendo definidos por diferentes critérios. O primeiro é a Regra Oja, que define o número de neurônios na camada oculta como:

$$H = T / 5(N + M) \quad (22),$$

sendo H o número de neurônios na camada oculta, T o número de amostras de treinamento, N o número de neurônios na camada de entrada e M o número de neurônios na camada de saída (GEVAERT et al., 2010). O segundo é a Regra Hecht-Nielsen, que defende que o número de neurônios na camada oculta pode ser definido como  $2N + 1$  (HECHT-NIELSEN, 1987). Os dois últimos são critérios empíricos, definindo o número de neurônios na camada oculta como  $N/2$  e  $2N/3 + M$  (THIAGO, 2017).

Testes foram realizados com essas redes a fim de se definir qual a melhor topologia para o caso em estudo. Nesses testes, criou-se a rede com sua respectiva topologia e, das 400 amostras de áudio disponíveis, 300 amostras (75%) foram utilizadas para treinamento, 60 amostras foram utilizadas para validação da rede (15%) e as 40 amostras restantes foram utilizadas como amostras de teste (10%). Os resultados dos testes com cada uma dessas redes são mostrados na Seção 4.3.

O sistema RAV desenvolvido é utilizado sempre que o usuário utiliza a modalidade de comando por voz na interface do programa. Os MFCCs são extraídos do sinal de voz do usuário obtido pela GUI, sendo aplicados na rede neural desenvolvida, classificando a fala como alguma das cores disponíveis.

## 4 RESULTADOS

Nas próximas seções serão apresentados os resultados referentes aos testes realizados no sistema a fim de se definir as configurações do controlador e topologia da rede neural, além de permitir uma análise do funcionamento do sistema como um todo.

### 4.1 CONFIGURAÇÃO DO CONTROLADOR *FUZZY*

Os testes realizados para a definição ideal da configuração do controlador *fuzzy* tem como objetivo verificar o desempenho do sistema com relação ao tempo de excursão, sobressinais e subsinais máximos da carga e tempo de acomodação no movimento do carro da ponte para controladores com diferentes ganhos e números de funções de pertinência. A relação entre esses fatores avaliados tende a apontar para a configuração ótima a ser implementada no sistema.

Para a realização desses testes, controladores *fuzzy* com 3, 5 e 7 conjuntos *fuzzy* foram embarcados no microcontrolador. Então, realizou-se o controle do movimento lateral do carro em um trajeto de 60 centímetros (de 15 até 75 centímetros), variando os valores de ganho e analisando as variáveis necessárias para cada situação. Vale lembrar que apenas o movimento lateral é controlado nesses testes; a carga não está submetida a controle antibalanco e, portanto, os dados de sobressinais e subsinais máximos são fruto tão somente da interferência do movimento lateral sobre a carga, permitindo avaliar o impacto sobre a sua oscilação. Além disso, por meio de ensaios prévios, percebeu-se que o sistema apresenta uma dinâmica diferente com relação aos trajetos de ida e volta. Portanto, essas análises foram discriminadas para os dois sentidos a fim de proporcionar uma avaliação mais completa e precisa.

A seguir são apresentados os resultados dos testes: a Tabela 6 apresenta os dados comparativos para o percurso de ida enquanto que a Tabela 7 apresenta para o percurso de volta. Já a Figura 38 mostra e compara exemplos de formas de onda da carga para os percursos de ida e volta para controlador com 3 conjuntos *fuzzy* e ganho 5.

Tabela 6 – Dados comparativos de controladores *fuzzy* com 3, 5 e 7 conjuntos para excursão de ida.

Nº Conjuntos fuzzy	Kp	Tempo de excursão (s)	Velocidade (cm/s)	Sobressinal máximo (%)	Sobressinal máximo (°)	Subsinal máximo (%)	Subsinal máximo (°)	Tempo de acomodação (s)
3	1	6,36	9,43	2,51	3,64	-4,63	-6,7	1,255
	2	4,67	12,85	3,42	4,98	-6,69	-9,75	1,715
	3	4,11	14,6	3,19	4,64	-6,77	-9,84	1,395
	5	3,64	16,48	3,4	4,92	-6,6	-9,57	1,37
	10	3,25	18,46	3,28	4,75	-6,65	-9,64	1,4
	20	3,01	19,93	4,32	6,26	-6,68	-9,69	1,395
5	1	6,87	8,73	2,81	4,09	-4,72	-6,87	1,305
	2	4,94	12,15	3,97	5,75	-5,95	-8,62	1,35
	3	4,33	13,86	4,36	6,33	-5,62	-8,14	1,34
	5	3,7	16,22	4,07	5,91	-5,93	-8,62	1,395
	10	3,28	18,29	4,49	6,51	-5,54	-8,04	1,32
	20	2,98	20,13	4,8	6,98	-6,3	-9,18	1,4
7	1	7,09	8,46	3,5	5,1	-4,98	-7,26	1,35
	2	4,96	12,1	4,06	5,9	-5,82	-8,47	1,325
	3	4,33	13,86	3,96	5,75	-5,98	-8,71	1,38
	5	3,69	16,26	4,02	5,83	-5,92	-8,59	1,325
	10	3,34	17,96	3,41	4,97	-6,52	-9,5	1,395
	20	3,01	19,93	4,83	7,04	-6,27	-9,13	1,375

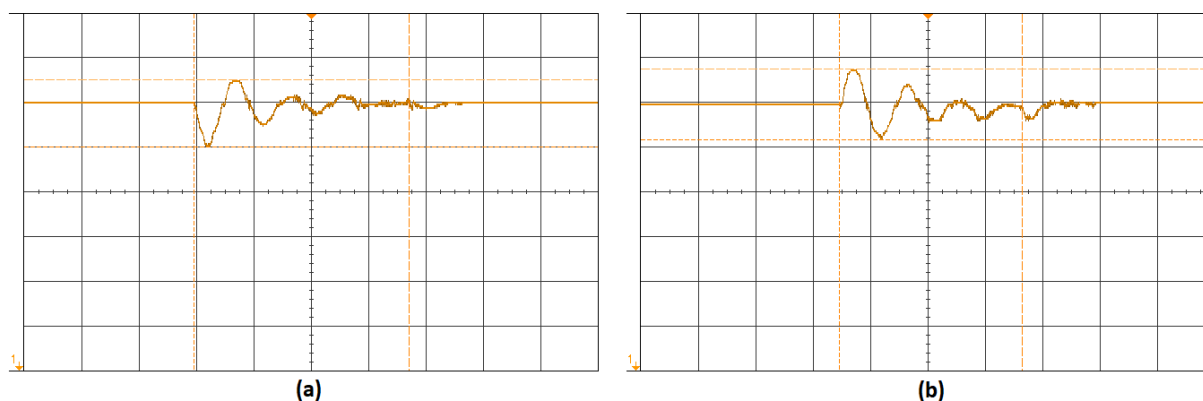
Fonte: Elaborado pelo autor (2019).

Tabela 7 – Dados comparativos de controladores *fuzzy* com 3, 5 e 7 conjuntos para excursão de volta.

Nº Conjuntos <i>fuzzy</i>	Kp	Tempo de excursão (s)	Velocidade (cm/s)	Sobressinal máximo (%)	Sobressinal máximo (°)	Subsinal máximo (%)	Subsinal máximo (°)	Tempo de acomodação (s)
3	1	5,86	10,24	4,39	6,39	-4,21	-6,13	1,68
	2	4,37	13,73	5,86	8,52	-4,92	-7,15	1,795
	3	3,8	15,79	5,78	8,39	-5,12	-7,43	1,915
	5	3,19	18,81	5,36	7,76	-5,17	-7,48	2,595
	10	2,9	20,69	5,23	7,56	-4,97	-7,19	3,005
	20	2,65	22,64	5,22	7,56	-6,64	-9,62	3,93
5	1	6,15	9,76	4,55	6,61	-4,05	-5,89	1,42
	2	4,56	13,16	5,39	7,78	-4,9	-7,09	1,78
	3	3,86	15,54	6,02	8,74	-5,11	-7,42	1,865
	5	3,36	17,86	5,89	8,57	-5,32	-7,74	2,725
	10	3	20,00	6,3	9,15	-5,05	-7,34	2,93
	20	2,68	22,39	6,15	8,92	-5,68	-8,24	4,405
7	1	6,16	9,74	4,92	7,15	-4,33	-6,31	1,765
	2	4,48	13,39	5,67	8,24	-5,36	-7,76	1,86
	3	3,89	15,42	5,32	7,74	-5,56	-8,11	1,91
	5	3,37	17,80	5,62	8,15	-5,41	-7,85	2,675
	10	3,06	19,61	5,52	8,04	-5,22	-7,56	2,995
	20	2,66	22,56	5,34	7,75	-6,47	-9,42	4,7

Fonte: Elaborado pelo autor (2019).

Figura 38 – Comparação das formas de onda da carga para os testes nos percursos de (a) ida e (b) volta da ponte para controlador com 3 conjuntos *fuzzy* e ganho 5.



Fonte: Elaborado pelo autor (2019).

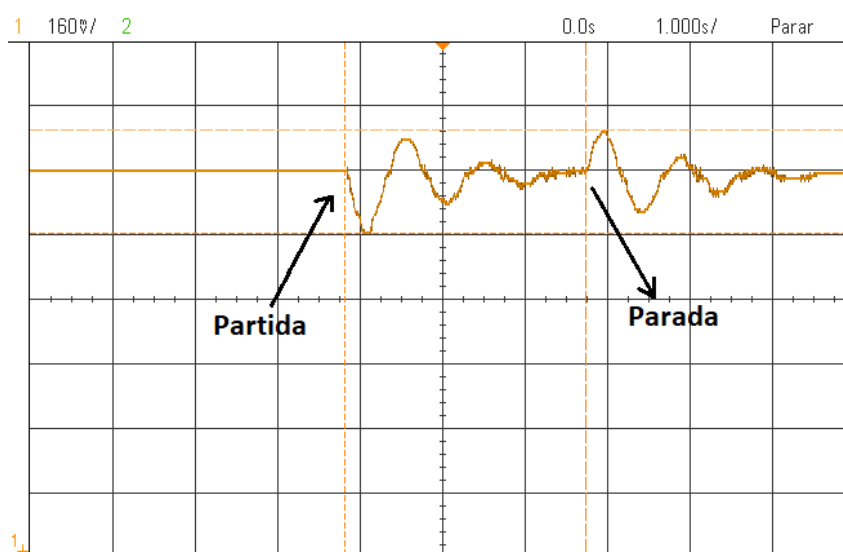
É possível observar na Figura 38 que a maior oscilação que a carga sofre é no momento da partida, representado pelos picos iniciais. Isso se dá pelo fato de que, na partida, o carro da ponte rolante movimenta-se com velocidade máxima. Além disso, os balanços máximos ocorrem em sentidos diferentes de acordo com o sentido de movimentação da ponte: se o carro está na excursão de ida, a maior oscilação da carga ocorre no sentido negativo; se o carro está na excursão de volta, a maior oscilação da carga ocorre no sentido positivo. Esses sentidos demonstram claramente o efeito da inércia da carga. Assim que o carro para na posição desejada, devido à sua inércia a carga sofre uma oscilação no sentido oposto ao da oscilação de partida; contudo, como o carro está em uma velocidade bem mais baixa, as oscilações são bem menores que na partida.

A análise das Tabela 6 e Tabela 7 permite não somente ratificar essas conclusões, como também tirar algumas outras. Primeiramente, o tempo de excursão e a velocidade do carro variam conforme a variação do ganho, isto é, quanto maior o ganho, menor o tempo de excursão e maior a velocidade. Mas não apenas a variação de ganho interfere no tempo de excursão e na velocidade: a variação de números de conjuntos *fuzzy* também intervém nessas variáveis. Para verificar isso, basta analisar os tempos e velocidades para ganhos unitários. Em todos os casos, o controlador com 3 conjuntos apresenta tempos inferiores ao dos outros controladores, chegando a apresentar uma diferença de até meio segundo. Já o controlador com 7 conjuntos é o mais lento de todos. Isso ocorre porque, como havia sido explicado na Seção 3.2.2.2, quanto maior o número de conjuntos, maior é a resolução da variável na entrada,

ocasionando uma variação na saída de forma mais gradativa, tornando o sistema mais lento. Essa diferença nos tempos de excursão e velocidades tende a diminuir conforme o valor do ganho é aumentado já que isso restringe a área efetiva de controle, fazendo com que o controlador permaneça em saturação por mais tempo. É observada também uma diferença com relação aos tempos de excursão e velocidades para os dois percursos. Claramente, o percurso de volta apresenta tempos de excursão menores e velocidades maiores, ratificando que as dinâmicas dos sistemas são, de fato, distintas para os dois trajetos e que a dinâmica para o percurso de volta é mais veloz. Isso influenciará diretamente na escolha da configuração do controlador *fuzzy* mais apropriado.

Visualiza-se nas tabelas que o aumento do ganho tende a aumentar também a intensidade das oscilações, principalmente na partida. Isso se torna crítico para os controladores com ganho 20. Foi dito que as maiores oscilações que a carga sofre são no momento da partida; isso é verdadeiro para todas as outras configurações, exceto para as configurações com ganho 20. Para o percurso de ida, por exemplo, os valores de sobressinal máximo apresentados são referentes ao momento de parada, e não partida. Idem para o sentido de volta, para os valores de subsinal máximo. Isso é um indicativo de que o valor de ganho 20 é muito alto para o sistema, ocasionando oscilações muito intensas não apenas na partida, como na parada. É possível observar esse fenômeno na Figura 39.

Figura 39 – Forma de onda da carga para percurso de ida e controlador com 3 conjuntos *fuzzy* e ganho 20, permitindo observar sobressinal máximo na parada.



É possível também analisar nas tabelas os dados sobre tempo de acomodação. Esses dados foram obtidos para valores no interior da faixa especificada por uma porcentagem absoluta do valor final de 2% (variação angular correspondente de  $2,9^\circ$ ) porque é nesse intervalo que a carga tem liberdade para se movimentar devido à margem admitida na leitura do potenciômetro (Seção 3.2.3.3). Além disso, essa análise foi feita para as oscilações de partida. Observa-se que, para o percurso de ida, os valores de tempo de acomodação são bem semelhantes para todas as configurações; já para o percurso de volta, os tempos são bem maiores. Isso pode ser explicado pelo fato de o sistema possuir uma dinâmica veloz no percurso de volta, ocasionando perturbações que se manterão por mais tempo devido à intensidade do movimento do carro. Isso se torna crítico para os controladores com ganho 20: nesses casos, quando o carro chega à posição desejada, as oscilações iniciais ainda não haviam chegado à faixa de 2% do valor final absoluto, isto é, o carro chega no destino com a carga ainda em grande oscilação.

A escolha da configuração ideal deverá considerar a análise conjunta das duas tabelas e seus respectivos dados. Com base nisso, ao analisar os valores para excursão de ida, percebe-se que a configuração 5 conjuntos *fuzzy* e ganho 10 apresenta um bom desempenho. Essa configuração alia um bom tempo de excursão a um nível reduzido de oscilações na partida (subsinal máximo). Já para a excursão de volta, a configuração 3 conjuntos *fuzzy* e ganho 10 seria a ideal pelos mesmos motivos, apesar de um tempo de acomodação um pouco elevado. O fato de haver configurações ideais distintas para as duas excursões confirma que o sistema apresenta uma dinâmica diferente para cada sentido, como fora comprovado por intermédio de testes preliminares.

Tendo em vista essas análises e considerando que o controlador implementado não terá o seu número de conjuntos *fuzzy* alterado durante a operação, optou-se por escolher uma configuração que apresente um número de conjuntos que tenha um bom desempenho tanto na ida, quanto na volta. Apesar da configuração 5 conjuntos *fuzzy* e ganho 10 ser muito boa para a ida, apresenta resultados ruins para a volta, além de entregar uma oscilação na parada para o percurso de ida um pouco acima das demais. Já a configuração 3 conjuntos *fuzzy* e ganho 10, além de ser a ideal para a volta, apresenta bom desempenho na ida também, apesar da oscilação um pouco elevada na partida, sendo compensada pelo baixo balanço na parada. Isso faz dessa configuração a ideal para a implementação no sistema. Ainda, será visto na próxima



seção que a presença do controle antibalanco irá mitigar consideravelmente o tempo de acomodação do sistema.

Em projetos futuros, pode-se trabalhar com uma configuração dinâmica do controlador, isto é, a aplicação das configurações mais apropriadas para ida e volta. Isso admitirá uma flexibilidade maior quanto à implementação do controlador *fuzzy*, bem como uma melhora no desempenho.

## 4.2 GANHOS IDEAIS PARA O CONTROLADOR ANTIBALANÇO

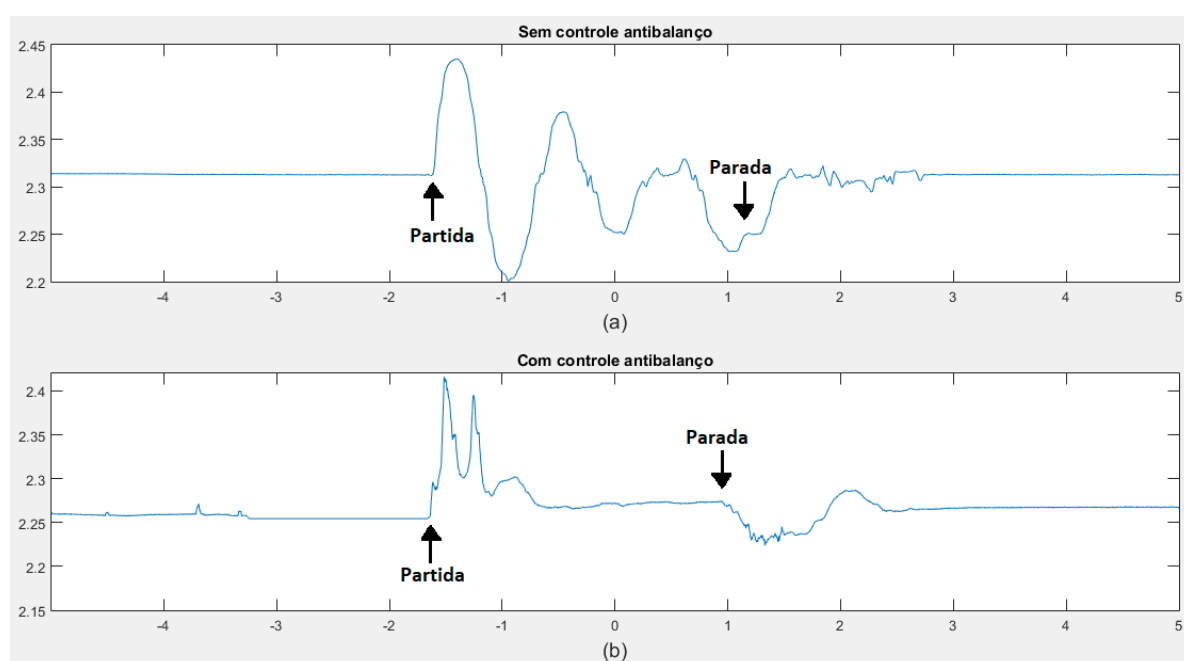
Com a configuração para o controlador responsável pelo movimento lateral definida, a próxima etapa consiste na definição dos ganhos ideais para o controlador antibalanco. Para isso, foram realizados testes semelhantes aos realizados para definição do controlador *fuzzy*, isto é, fez-se o carro da ponte se movimentar em um trajeto de 60 centímetros, analisando-se variáveis como tempo de acomodação e sobressinal máximo da carga. A diferença é que, além do movimento lateral estar sendo unicamente controlado pelo controlador *fuzzy* definido anteriormente (3 conjuntos + ganho 10), o controle antibalanco é adicionado ao sistema. Com isso, os testes foram realizados de forma que os ganhos proporcional e derivativo do controlador antibalanco assumissem diversos valores; por exemplo, o  $K_p$  assumiu os valores 1, 2, 5, 10 e 22,2 (valor estabelecido pela sintonia Ziegler-Nichols), enquanto que o  $K_d$  assumiu os valores 0, 0,25, 0,56 (valor estabelecido pela sintonia Ziegler-Nichols), 1 e 2.

Nesses testes, constatou-se que a resposta é bem semelhante para todos os pares de ganhos avaliados. Isso acontece porque o controlador antibalanco atua basicamente nas oscilações de parada e partida, sendo esta última a mais significativa. Todavia, essas oscilações são de pequena magnitude, como pôde ser constatado na Seção 4.1, em que o caso mais crítico sem controle antibalanco apresentou oscilações que não chegaram nem a  $10^\circ$ . Devido a isso, o comportamento da carga submetida a um controlador antibalanco não varia com a mesma intensidade da variação dos ganhos. Para que fosse possível observar grandes distinções entre os valores, seria necessária uma análise para oscilações muito maiores que  $10^\circ$ , o que foge não apenas da realidade de um controlador antibalanco aplicado a pontes rolantes como também dos modelos e aproximações assumidos para movimento pendular (Seção 2.1).

Apesar de não haver grandes distinções para respostas de diferentes configurações do controlador antibalço, existe uma grande melhora nas oscilações da carga em comparação com os testes sem controle. Para ilustrar isso, serão considerados os testes para o controlador PD estabelecido pela sintonia Ziegler-Nichols, isto é,  $K_p = 22,2$  e  $K_d = 0,56$ .

Nos testes sem controle antibalço, constatou-se que o tempo de acomodação foi de 1,4 segundos para o percurso de ida e 3 segundos para a volta, para a configuração escolhida de controlador *fuzzy*. Esperava-se que o tempo de acomodação fosse melhorado com a presença do controle antibalço, e de fato foi. Para os testes com controle antibalço, o tempo de acomodação caiu drasticamente para 0,475 segundos para o percurso de ida e 0,47 segundos para o percurso de volta. Isso comprova a importância do controle antibalço para estabilização do sistema. Na Figura 40 – Comparação das respostas da carga para movimento lateral do carro (a) sem controle antibalço e (b) com controle antibalço da carga. é possível observar a comparação entre as respostas sem e com controle antibalço para o percurso de volta; é visível como que, desde a partida, a carga oscila por muito mais tempo sem controle, enquanto que para o caso com controle existe uma grande redução no tempo das oscilações e, conseqüentemente, no tempo de acomodação.

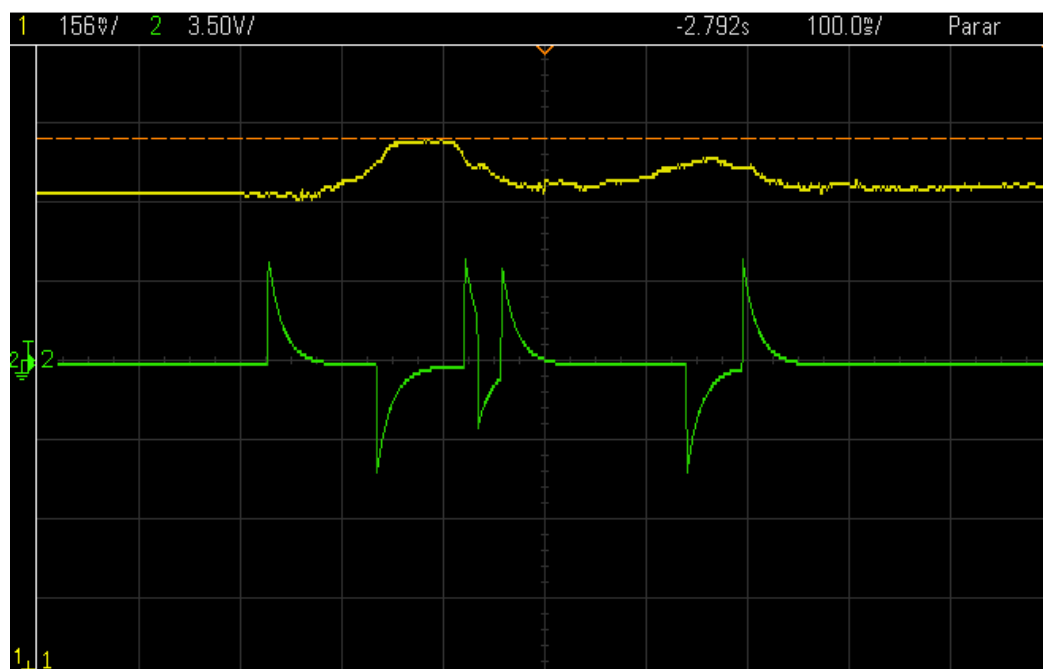
Figura 40 – Comparação das respostas da carga para movimento lateral do carro (a) sem controle antibalço e (b) com controle antibalço da carga.



Fonte: Elaborado pelo autor (2019).

Analisando o gráfico da resposta com controle é possível observar que, após a partida, a carga apresenta três oscilações até alcançar a estabilidade. A explicação é a seguinte: assim que ocorre a partida, a carga irá sofrer sua maior oscilação, representada pelo primeiro pico. Então, o controlador entrará em ação, fazendo com que o carro se mova na direção da oscilação a fim de neutralizá-la; por isso é possível observar uma queda no valor lido do potenciômetro da carga após o primeiro pico. O valor lido do potenciômetro da carga cai até entrar na margem da leitura do potenciômetro, levando o microcontrolador interpretar que a carga está estabilizada, sendo possível continuar com o movimento lateral. Com isso, assim que o carro arranca novamente para continuar seu movimento, uma segunda oscilação é gerada, agora de valor de pico reduzido em comparação com a primeira por conta da atenuação pela ação anterior de controle. O mesmo processo ocorre mais uma vez, gerando a terceira oscilação. Após isso, a carga se estabiliza dentro da margem aceitável. A atuação do controlador pode ser exemplificada através da Figura 41.

Figura 41 – Gráfico da atuação do controlador (em verde) de acordo com a movimentação da carga (em amarelo).



Fonte: Elaborado pelo autor (2019).

A Figura 41 foi obtida para uma movimentação do carro em um percurso de volta e apresenta a atuação do controlador, mostrada pelo gráfico verde, de acordo com a

movimentação da carga, mostrada pelo gráfico amarelo. Os picos positivos no gráfico do controlador representam a movimentação do carro para a esquerda, enquanto que os picos negativos representam a movimentação para a direita. Já para o gráfico da carga, uma oscilação positiva representa uma oscilação para a direita por parte da carga. Com base nisto, algumas análises podem ser feitas. Primeiramente, o pico inicial de atuação do controlador é positivo, confirmando que o carro está no sentido de volta, isto é, se movimentando para a esquerda. Então, pouco tempo após a partida do carro, devido à inércia, a carga apresenta uma oscilação para a direita. Para corrigir isto, é necessário que o carro interrompa o movimento anterior (para a esquerda) e se mova no sentido da oscilação da carga (para a direita) a fim de eliminar o balanço; isto é exemplificado pelo primeiro pico negativo de atuação do controlador. Vale ressaltar que este pico negativo se deu apenas no momento que o valor lido do potenciômetro da carga saiu da margem de *setpoint* para o controle antibalanço, definida na Seção 3.2.3.3. Então, a oscilação da carga é mitigada, permitindo que o carro volte a se movimentar no sentido da excursão de volta. Contudo, assim que o carro volta a se movimentar, a carga apresenta um novo balanço, agora de amplitude reduzida. Então, o processo anterior de eliminação do balanço se repete, como pode ser visto pelos picos em sucessão, até que a carga se estabilize e o carro possa se movimentar para o seu destino final, sem interrupções, como visto a partir do último pico.

Quanto ao sobressinal máximo, os testes com  $K_p = 22,2$  e  $K_d = 0,56$  apresentaram resultados semelhantes não só para outros diferentes pares de ganhos como também para o caso sem controle. Isso pode ser explicado pelo baixo nível percentual de sobressinal máximo decorrente da partida; como as oscilações não são de altíssima magnitude, elas são atingidas de forma rápida (aproximadamente 12,5 milissegundos). Por isso, quando o controlador entra em ação para eliminar o balanço, a carga já atingira seu sobressinal máximo. Esse comportamento ocorre apesar da presença da ação derivativa, que tem como característica a antecipação do erro. De fato, é possível observar a ação efetiva do derivativo para oscilações maiores; contudo, já foi dito que isso foge do propósito do controlador desenvolvido. Um aumento no ganho derivativo poderia reduzir as oscilações máximas na partida; contudo, para valores de ganho derivativo acima de 2, o maior valor testado, o sistema apresenta trepidações severas, prejudicando o desempenho.

Com esses resultados, optou-se por utilizar os valores de ganho sugeridos pelo método de sintonia Ziegler-Nichols, isto é,  $K_p = 22,2$  e  $K_d = 0,56$ .

#### 4.3 DESEMPENHO DE DIFERENTES TOPOLOGIAS DE REDES NEURAIS

Conforme mencionado na Seção 3.3.4, testes foram realizados com diversas topologias de redes neurais a fim de se definir qual a topologia ideal para a implementação no sistema RAV utilizado neste trabalho.

O teste realizado consistiu em, para cada topologia, submeter amostras de teste às redes criadas via *nstart* e verificar a porcentagem de acertos para cada palavra e o número de épocas que foram necessárias para o treinamento da rede. Três testes desse tipo foram feitos, obtendo ao final as médias dos resultados, sendo esses mostrados na Tabela 8. Vale ressaltar que, para estes testes, 100 amostras de áudio por cor foram utilizadas para o treinamento das redes, resultando em 400 amostras de áudio no total.

Tabela 8 – Resultados dos três testes de desempenho para as diferentes topologias de redes neurais treinadas com 100 amostras de áudio por cor.

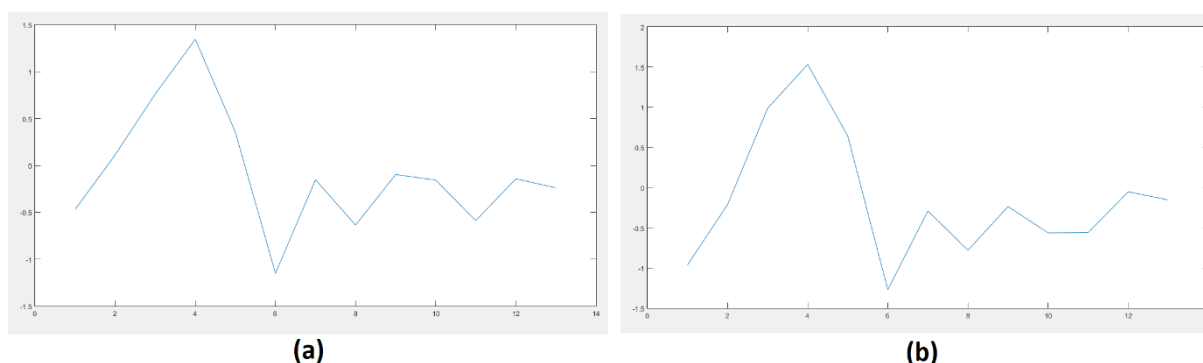
Topologia	H	Acertos (%)				Média de acertos (%)	Número de épocas de treinamento
		Vermelho	Verde	Azul	Amarelo		
Regra Oja	4	77,78	88,89	100	100	91,67	32
Hecht-Nielsen	27	93,33	93,33	100	100	96,67	31
$N/2$	7	82,22	93,33	100	100	93,89	27
$2N/3 + M$	13	91,11	95,55	100	100	96,67	32

Fonte: Elaborado pelo autor (2019).

Analisando a tabela, diversas conclusões podem ser obtidas. Primeiramente, é possível observar a diferença de números de neurônios na camada oculta para cada topologia de rede, tornando-as bem distintas já que essa diferença irá interferir diretamente no treinamento e, portanto, desempenho da rede.

Segundo, é interessante observar que os erros de reconhecimento aconteceram exclusivamente para as palavras “vermelho” e “verde”; para as palavras “azul” e “amarelo” não houveram erros, isto é, todas foram reconhecidas com sucesso por todas as topologias. Além disso, os erros de classificação da palavra “vermelho” aconteceram porque esta palavra foi classificada como “verde”, e vice-versa; estas duas palavras nunca foram confundidas com as palavras “azul” e “amarelo”. Isso pode ser explicado pelo fato de que, enquanto a pronúncias das duas últimas palavras são bem distintas entre todas, facilitando o reconhecimento por parte da rede, as pronúncias das primeiras duas palavras são semelhantes, principalmente nas primeiras sílabas, ocasionando eventuais erros de reconhecimento. A Figura 42 mostra a semelhança entre os coeficientes que caracterizam as palavras “vermelho” e “verde”.

Figura 42 – Comparação de MFCCs para sinais de voz correspondentes às palavras (a) "vermelho" e (b) "verde".



Fonte: Elaborado pelo autor (2019).

Apesar dos erros, as redes possuem um desempenho médio satisfatório, já que a rede com o pior desempenho apresenta uma média de acertos superior a 90% (91,67%). Isso pode ser explicado pela forma de reconhecimento aplicada, isto é, dependente de locutor. Um sistema RAV dependente de locutor tende a ter um desempenho maior quando comparado a um sistema independente de locutor porque as amostras colhidas são todas de uma única fonte, fazendo com que, ainda que existam distinções nos sinais de voz, essas não sejam tão significativas quando comparadas com amostras de outras fontes.

Os números de épocas para treinamento das redes basicamente não variam muito; além disso, esses valores são bem pequenos já que os números de neurônios na

camada oculta também não são grandes, de forma a influenciar diretamente no desempenho computacional de treinamento da rede. Todas as redes criadas tiveram um treinamento bem rápido, com tempos que chegaram a, no máximo, 1 segundo. Analisando a média de acertos das redes, pode-se destacar duas que tiveram um desempenho melhor, sendo essas as redes com  $H = 27$  (Hecht-Nielsen) e  $H = 13$  (empírica com fórmula  $2N/3 + M$ ), ambas com 96,67% de média de acertos. Portanto, a rede aplicada ao sistema RAV seria uma dessas duas. O fator crucial para a escolha de uma delas é o que as diferencia grandemente: número de neurônios na camada oculta. A rede  $2N/3 + M$  possui menos que a metade de neurônios na camada oculta da rede Hecht-Nielsen. Portanto, optou-se por aplicar a rede de topologia  $2N/3 + M$  ao sistema de reconhecimento de voz, já que essa rede obteve um desempenho praticamente igual ao da rede Hecht-Nielsen, porém com um número menor de neurônios, otimizando o sistema.

Os testes foram realizados para redes treinadas com 100 amostras de áudio para cada cor. Escolheu-se essa grande quantidade de amostras para que se pudesse ter uma rede bem treinada e, conseqüentemente, bons resultados de classificação. Contudo, em outras aplicações com um número maior de possíveis classificações, por exemplo, essa quantidade pode ser grande demais, inviabilizando o treinamento. Portanto, outros testes, semelhantes aos realizados anteriormente, foram feitos a fim de se verificar o desempenho das redes quando submetidas a um treinamento com menos amostras de áudio. Verificou-se o desempenho das diferentes topologias de *perceptron* multicamadas quando treinadas com 50 e 20 amostras de áudio; os resultados são vistos nas Tabela 9 e Tabela 10.

Algumas conclusões podem ser obtidas após a análise das Tabela 9 e Tabela 10. Primeiramente, é possível observar uma queda na média de acertos das redes, principalmente para o caso de treinamento com 20 amostras de áudio. Isso indica que, quanto mais amostras disponíveis no treinamento da rede neural, melhor o seu desempenho tende a ser. Para ratificar isto, constata-se que houve erros na classificação das palavras “azul” e “amarelo”, diferentemente dos resultados para redes treinadas com 100 amostras de áudio, em que essas duas palavras foram classificadas corretamente por todas as redes. Os números de épocas de treinamento sofreram uma redução nos dois últimos casos de teste quando comparados aos do

primeiro caso, resultado este que é natural tendo em vista que menos amostras foram utilizadas no treinamento.

Tabela 9 – Resultados dos três testes de desempenho para as diferentes topologias de redes neurais treinadas com 50 amostras de áudio por cor.

Topologia	H	Acertos (%)				Média de acertos (%)	Número de épocas de treinamento
		Vermelho	Verde	Azul	Amarelo		
Regra Oja	4	88,89	77,78	100	97,78	91,11	26
Hecht-Nielsen	27	80	91,11	95,56	100	91,67	28
$N/2$	7	77,78	95,56	100	100	93,33	25
$2N/3 + M$	13	91,11	93,33	100	100	96,11	27

Fonte: Elaborado pelo autor (2019).

Tabela 10 – Resultados dos três testes de desempenho para as diferentes topologias de redes neurais treinadas com 20 amostras de áudio por cor.

Topologia	H	Acertos (%)				Média de acertos (%)	Número de épocas de treinamento
		Vermelho	Verde	Azul	Amarelo		
Regra Oja	4	66,67	95,56	100	100	90,56	21
Hecht-Nielsen	27	82,22	68,89	80	100	82,78	19
$N/2$	7	80	86,67	95,55	86,67	87,22	19
$2N/3 + M$	13	86,67	88,89	93,33	100	92,22	14

Fonte: Elaborado pelo autor (2019).

Nos dois novos casos de teste, a topologia escolhida através do primeiro caso para ser aplicada ao sistema, isto é, 13 neurônios na camada oculta ( $2N/3 + M$ ), se manteve como a que possui o melhor desempenho. A Tabela 11 apresenta a comparação dos desempenhos nos três casos de teste para essa topologia.



É possível verificar uma pequena queda no desempenho geral da rede conforme o número de amostras de treinamento cai. Essa queda de desempenho é fruto de uma redução nas classificações corretas das palavras “azul” e, principalmente, “vermelho” e “verde”, casos mais críticos como já fora relatado.

Tabela 11 – Comparação dos resultados dos testes de desempenho da rede de topologia  $2^N/3 + M$  quando treinada com 100, 50 e 20 amostras de treinamento.

Número de amostras de treinamento	Acertos (%)				Média de acertos (%)	Número de épocas de treinamento
	Vermelho	Verde	Azul	Amarelo		
100	91,11	95,55	100	100	96,67	32
50	91,11	93,33	100	100	96,11	27
20	86,67	88,89	93,33	100	92,22	14

Fonte: Elaborado pelo autor (2019).

Apesar da queda de desempenho, a rede ainda mantém uma boa classificação geral, com médias de acertos acima de 90%. Portanto, para uma outra aplicação que venha requerer uma classificação de mais cores, é válido considerar realizar o treinamento com menos amostras. Para isso, é importante avaliar se esta queda de desempenho, fruto da redução do número de amostras de treinamento, é altamente significativa para a aplicação.

#### 4.4 TESTES COM O SISTEMA INTEGRADO

Tendo os testes anteriores sido feitos, foi possível realizar testes com todo o sistema integrado, fazendo uso dos resultados obtidos até então. Para isso, todos os comandos e definições de valores de ganhos e posições foram realizados via GUI. Não foi necessário realizar outro teste a fim de verificar o comportamento do sistema com relação a balanço da carga, tempo de excursão, tempo de acomodação e afins porque os resultados dos testes da Seção 4.2 representam os resultados do sistema

integrado, considerando as configurações de controladores antibalanco e *fuzzy* escolhidas.

Contudo, fez-se necessária uma análise com relação ao tempo de resposta do sistema diante dos comandos aplicados via interface. Esta análise seria efetuada para dois casos: acionamento por comando de voz e acionamento por comando manual. Para isso, mediu-se o tempo entre o acionamento do carro através da GUI e o início efetivo do seu movimento; no caso do acionamento por comando de voz, o tempo seria contado a partir do momento em que a obtenção do sinal de voz fosse encerrada. Cinco testes foram realizados para cada caso, obtendo-se ao final a média dos resultados.

Os testes para acionamento por comando de voz apresentaram um resultado para tempo de resposta médio de 2,91 segundos, isto é, o carro da ponte rolante demorou 2,91 segundos para iniciar o seu movimento após a obtenção do sinal de voz do usuário. Já os testes para acionamento por comando manual apresentaram um resultado para tempo de resposta médio de 0,2 segundos.

Com base nisto, fica clara a diferença entre os tempos de resposta dos dois modos de acionamento. O acionamento por comando manual permite uma movimentação quase que instantânea do carro, enquanto que o acionamento por comando de voz causa um atraso no início do movimento do carro de quase 15 vezes maior que o caso anterior. Isso pode ser explicado pela simplicidade do processo de acionamento manual, que consiste basicamente em montar o pacote de comunicação e enviar ao microcontrolador, bem como pela complexidade do processo de acionamento por comando de voz, que requer a execução de todas as etapas apresentadas na Seção 3.3.

O funcionamento do sistema integrado é apresentado através de um vídeo, podendo ser acessado pelo link: <https://www.youtube.com/watch?v=Fo2r0BO4p7E&t=1s>. Além disso, todos os códigos fonte utilizados para o funcionamento do sistema foram disponibilizados livremente através do GitHub®, podendo ser acessados pelo link: <https://github.com/ChristoferGB/Controle-de-ponte-rolante>.

## 5 CONCLUSÃO

Este trabalho agregou aos estudos e aplicações de sistemas automatizados já existentes, cumprindo com a proposta de desenvolver um controle antibalanco aplicado a uma ponte rolante didática. O modelo utilizado é de pequena escala se comparado a uma ponte rolante real; contudo, o sucesso da aplicação didática abre o caminho para uma implementação em modelo real.

Comprovou-se que é possível integrar o controle dos movimentos laterais do carro de uma ponte rolante e o controle antibalanco de sua carga sem que se perca desempenho, isto é, mantendo um bom tempo de excursão e acomodação e uma diminuta oscilação na carga.

Ademais, constatou-se a eficácia da lógica e controle *fuzzy* na implementação do controle dos movimentos laterais do carro, mostrando-se uma boa alternativa à lógica booleana e ao controle clássico, principalmente quando o sistema é complexo de se modelar matematicamente e o projetista possui conhecimento prático no mesmo sistema.

Este trabalho também apresentou o comando por voz como uma alternativa no acionamento de sistemas. Conquanto que proporcione uma boa base dados, uma caracterização fidedigna do sinal de voz e um classificador adequado para a dada aplicação, um sistema RAV é uma excelente opção para automatização, podendo oferecer um desempenho altíssimo de reconhecimento, como visto nesse trabalho.

Algumas dificuldades foram encontradas na elaboração deste trabalho, sendo a principal delas a constante mudança do sinal de referência da carga por conta de flutuações na tensão da fonte, além do fato de que o potenciômetro utilizado não era de precisão. Apesar disso, os resultados encontrados foram satisfatórios, cumprindo com os objetivos propostos.

### 5.1 PERSPECTIVA PARA TRABALHOS FUTUROS

Para trabalhos futuros, propõem-se as seguintes implementações:

- implementação de uma configuração dinâmica de controlador *fuzzy* a fim de permitir a aplicação da configuração ideal para cada sentido de movimento do carro na ponte rolante, conforme mencionado no final da Seção 4.1.

- Alteração da estrutura do protótipo a fim de possibilitar o controle de elevação da carga.
- Emprego de um sistema RAV independente de locutor.
- Aplicação de outros classificadores a um sistema RAV independente de locutor como, por exemplo, o HMM para fins de comparação do nível de reconhecimento.
- Implementação de realimentação por imagem para a medição da posição do carro da ponte no trilho.
- Substituição do potenciômetro por outros sensores como, por exemplo, giroscópio para a realimentação da carga a fim de aumentar a precisão de leitura.
- Desenvolvimento de aplicativo móvel para acionamento e supervisão do sistema.

Essas propostas são importantes pois permitirão a utilização de outras metodologias e ferramentas, permitindo uma análise comparativa com os resultados obtidos por este trabalho. Isso visa enriquecer a aplicação e ampliar as possibilidades de uso do protótipo.

## REFERÊNCIAS

ADOKO, A. C.; GOKCEOGLU, C.; WU, L.; ZUO, Q. J. Knowledge-based and data-driven fuzzy modeling for rockburst prediction. **International Journal of Rock Mechanics and Mining Sciences**, v. 61, p. 86–95, 2013.

ALVES, A. J. **eFLL (embedded Fuzzy Logic Library)**. Biblioteca para sistemas embarcados a fim de implementar sistemas *Fuzzy*. Disponível em: <<https://github.com/zerokol/eFLL>>. Acesso em: 14 mai. 2019.

ASSIS, A. A. **Desenvolvimento de um sistema de controle difuso anti-balanço para um modelo didático de ponte rolante**. 2015. 65 f. Monografia (Graduação em Engenharia Elétrica) - Instituto Federal do Espírito Santo, Vitória, 2015. Disponível em: <<https://biblioteca2.ifes.edu.br/vinculos/000009/0000099D.pdf>>. Acesso em: 30 out. 2017.

AVILA, J. R.; ALCÁNTARA-RAMÍREZ, R.; JAIMES-PONCE, J.; SILLER-ALCALÁ, I. Design of a self-balancing tower crane. In: 2nd WSEAS Int. Conf. on Circuits, Systems, Signal and Telecommunications (CISST'08), 2., 2008, Acapulco, México. **Anais...** Acapulco, México: WSEAS, 2008. Disponível em: <<https://pdfs.semanticscholar.org/b259/6c7a457e0359589199fb587196b57b44db6c.pdf>>. Acesso em: 12 jun. 2019.

BAI, E.; CHO, H. Semi-global convergence results on control of systems containing a dead zone by an adaptive dead zone inverse. In: American Control Conference - ACC'95. **Anais...** Seattle, WA, USA: IEEE, 1995. p. 2044-2048.

BAIÔCO, R. L. **Sistema anti-balanço para transporte de cargas em pontes rolantes**. 2012. 110 f. Monografia (Graduação em Engenharia Elétrica) - Instituto Federal do Espírito Santo, Vitória, 2012. Disponível em: <<https://biblioteca2.ifes.edu.br/vinculos/000009/0000099D.pdf>>. Acesso em: 21 nov. 2017.

BENBA, A.; JILBAB, A.; HAMMOUCH, A. Detecting Patients with Parkinson ' s disease using Mel Frequency Cepstral Coefficients and Support Vector Machines. **International Journal on Electrical Engineering and Informatics**, v. 7, n. 2, p. 297–307, 2015.

CHOQUEHUANCA, C. Controle. In:\_\_\_\_\_. **Projeto e controle robusto de um transportador pessoal robótico auto equilibrante**. 2010. p. 39. Dissertação (Mestrado) – Pontifícia Universidade Católica, Rio de Janeiro, 2010. Disponível em: <[https://www.maxwell.vrac.puc-rio.br/17228/17228\\_4.PDF](https://www.maxwell.vrac.puc-rio.br/17228/17228_4.PDF)>. Acesso em: 20 nov. 2017.

CIPRIANO, J. L. G. **Desenvolvimento de Arquitetura Para Sistemas de Reconhecimento Automático de Voz Baseados em Modelos Ocultos de Markov**. 2001. 125 f. Tese (Doutorado em Ciência da Computação) - Programa de Pós-Graduação em Computação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2001. Disponível em:

<[http://www.lume.ufrgs.br/handle/10183/2633?locale-attribute=pt\\_BR](http://www.lume.ufrgs.br/handle/10183/2633?locale-attribute=pt_BR)>. Acesso em: 1 de abr. 2019.

DESAI, S.; BLACK, A. W.; YEGNANARAYANA, B.; PRAHALLAD, K. Spectral Mapping Using Artificial Neural Networks for Voice Conversion. **IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING**, v. 18, n. 5, p. 954–964, 2010.

ENCYCLOPÆDIA BRITANNICA, Britannica Academic. **Graphical user interface (GUI)**. 2010. Disponível em: <[academic-eb-britannica.ez120.periodicos.capes.gov.br/levels/collegiate/article/graphical-user-interface/109589#](http://academic-eb-britannica.ez120.periodicos.capes.gov.br/levels/collegiate/article/graphical-user-interface/109589#)>. Acesso em: 21 nov. 2017.

FARIAS, C.; SANTOS, C.; LOURENÇO, A.; CARNEIRO, T. Kohonen neural networks for rainfall-runoff modelling: case study of Piancó river basin. **Journal of Urban and Environmental Engineering**, v. 7, n. 1, p. 176–182, 2013.

GEVAERT, W.; TSENOV, G.; MLADENOV, V. Neural networks used for speech recognition. **Journal of Automatic Control**, n. 20, p. 1–7, 2010.

GIL, A. C. **Como elaborar projetos de pesquisa**. 4 ed. São Paulo: Atlas, 2002.

GUARAV, D. S. D.; GOPAL, M. B. Development of Application Specific Continuous Speech Recognition System in Hindi. **Journal of Signal and Information Processing**, v. 3, n. 3, p. 394–401, 2012.

GUINCHO 24 HORAS. **Ponte rolante**. il. color. Disponível em <<http://www.transportadora.sorocaba.emp.br/public/img/default/guincho-24-horas/home/ponte-rolante.png>>. Acesso em: 11 jun. 2019.

HALLIDAY, D.; RESNICK, R.; WALKER, J. **Fundamentals of Physics**. 9 ed. NJ, USA: John Wiley & Sons, 2010.

HAMMING, R. W. **Digital Filters**. 3. ed. Hertfordshire, UK: Prentice Hall International, 1989.

HAYKIN, S. **Neural Networks. A Comprehensive Foundation**. 2. ed. Hamilton, Ontario, Canadá: Prentice Hall, 1998.

HECHT-NIELSEN, R. H. **Kolmogorov's Mapping Neural Network Existence Theorem**. Proceedings of the IEEE First International Conference on Neural Networks. **Anais...** San Diego, CA, USA: Piscataway, NJ: IEEE, 1987

HICKERSBERGER, H.; ZAGLER, W. L. A voice command system for AUTONOMY using a novel speech alignment algorithm. **International Journal of Speech Technology**, v. 16, n. 4, p. 461–469, 2013.

HUANG, X.; ACERO, A.; HON, H. **Spoken Language Processing: A guide to theory, algorithm, and system development**. 1. ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

LYONS, James. Mel Frequency Cepstral Coefficient (MFCC) tutorial. 2013. **Practical Cryptography**. Disponível em:

<<http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/>>. Acesso em: 16 mai. 2019.

MANDAL, S. et al. Human error identification and risk prioritization in overhead crane operations using HTA, SHERPA and fuzzy VIKOR method. **Expert Systems with Applications**, v. 42, n. 20, p. 7195–7206, 2015.

MARQUEZ, A. A.; VENTURINO, P.; OTEGUI, J. L. Common root causes in recent failures of cranes. **Engineering Failure Analysis**, v. 39, p. 55–64, 2014.

MARTINS, R. M. **Análise comparativa entre os métodos HMM e GMM-UBM na busca pelo  $\alpha$ -ótimo dos locutores crianças para utilização da técnica VTLN**. 2014. 78 f. Dissertação (Mestrado em Telecomunicações) – Instituto Nacional de Telecomunicações, Santa Rita do Sapucaí, 2014. Disponível em:

<<http://tede.inatel.br:8080/jspui/bitstream/tede/23/5/Dissertação%20V.Final%20Ramon%20Mayor%20Martins.pdf>>. Acesso em: 11 de abr. 2019.

MATHWORKS. **MATLAB**, R2012b. *Software* interativo de alto desempenho voltado para o cálculo numérico e operações com matrizes. 2012. Disponível em:

<<https://www.mathworks.com/products/matlab.html>>. Acesso em: 3 jul. 2019.

MEDINA, C.; VELAZCO, S.; SALINAS, J. Experimental control of simple pendulum model. **The Pendulum: Scientific, Historical, Philosophical and Educational Perspectives**, v. I, n. 1992, p. 67–76, 2005.

MICROCHIP. **Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V**. San Jose, CA, USA: [s.n.], 2014. 435 p. (Folha de especificações de microcontrolador). Disponível em: <[http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561\\_datasheet.pdf](http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf)>. Acesso em: 10 jun. 2019.

NIMJE, K.; SHANDILYA, M. Automatic isolated digit recognition system : an approach using HMM. **Journal of Scientific & Industrial Research**, v. 70, n. 4, p. 270–272, 2011.

NUNES, I.; HERNANE, D.; FLAUZINO, R.; LIBONI, L. H.; ALVES, S. **Artificial Neural Networks**. 1. ed. São Carlos, SP, Brasil: Springer International Publishing, 2017.

NYQUIST, H. Certain Topics in Telegraph Transmission Theory. **Transactions of the American Institute of Electrical Engineers**, v. 47, n. 2, p. 617–644, 1928.

OGATA, K. **Modern Control Engineering**. 4 ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

RABINER, L. R.; SCHAFER, R. W. **Digital Processing of Speech Signals**. 1. ed. NJ, USA: Prentice Hall, 1978.

RABINER, L. R.; SCHAFER, R. W. Introduction to Digital Speech Processing. **Foundations and Trends in Signal Processing**, v. 2, n. 1–2, p. 1–194, 2007.

RANDALL, R. B. A history of cepstrum analysis and its application to mechanical problems. **Mechanical Systems and Signal Processing**, v. 97, p. 3–19, 2017.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386–408, 1958.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **Nature**, v. 323, n. 6088, p. 533–536, 1986.

SAPOZHNIKOVA, M. U. et al. Anti-fraud system on the basis of data mining technologies. In: 2017 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 17., 2017, Bilbao, Espanha. **Anais...Bilbao, Espanha: IEEE**, 2017. Disponível em: <<https://ieeexplore.ieee.org/document/8388649>>. Acesso em: 31 mai. 2019.

SHAW, I.; SIMOES, M. G. **Controle E Modelagem Fuzzy**. 2 ed. São Paulo: Blucher, 2007.

SIDDHANT, C. J.; CHEERAN, A. N. MATLAB Based Feature Extraction Using Mel Frequency Cepstrum Coefficients for Automatic Speech Recognition. **International Journal of Science, Engineering and Technology Research (IJSETR)**, v. 3, n. 6, p. 1820–1823, 2014.

SILVA, D. F.; SOUZA, V. M. A. DE; BATISTA, G. E. A. P. A. A comparative study between MFCC and LSF coefficients in automatic recognition of isolated digits pronounced in Portuguese and English. **Acta Scientiarum. Technology**, v. 35, n. 4, p. 621–628, 2013.

STUPAK, N.; DIFONZO, N.; YOUNGE, A. J.; HOMAN, C. SOCIALSENSE: Graphical user interface design considerations for social network experiment software. **Computers in Human Behavior**, v. 26, n. 3, p. 365–370, 2010.

TAYLOR, P. **Text-to-Speech Synthesis**. 1. ed. NY, USA: Cambridge University Press, 2009.

THIAGO, E. R. DE S. **Reconhecimento de Voz utilizando extração de Coeficientes Mel-Cepstrais e Redes Neurais Artificiais**. 2017. 74 f. Monografia (Graduação em Engenharia de Telecomunicações) - Instituto Federal de Santa Catarina, São José, 2017. Disponível em: <[https://wiki.sj.ifsc.edu.br/wiki/index.php/Reconhecimento\\_de\\_Voz\\_utilizando\\_extra%C3%A7%C3%A3o\\_de\\_Coeficientes\\_Mel-Cepstrais\\_e\\_Red\\_Ne\\_Ne\\_Artificiais](https://wiki.sj.ifsc.edu.br/wiki/index.php/Reconhecimento_de_Voz_utilizando_extra%C3%A7%C3%A3o_de_Coeficientes_Mel-Cepstrais_e_Red_Ne_Ne_Artificiais)>. Acesso em: 17 mai. 2019.

VIEIRA, J. P. A.; BEZERRA, U. H.; NUNES, M.; DO NASCIMENTO, A. C.; BARRA JR., W. Controladores fuzzy aplicados ao conversor de geradores de indução duplamente excitados em sistemas eólicos integrados a sistemas de potência. **Controle y Automacao**, v. 18, n. 1, p. 115–126, 2007.



YEROGLU, C.; TAN, N. Classical controller design techniques for fractional order case. **ISA Transactions**, v. 50, n. 3, p. 461–472, 2011.

YU, W.; MORENO-ARMENDARIZ, M. A.; RODRIGUEZ, F. O. Stable adaptive compensation with fuzzy CMAC for an overhead crane. **Information Sciences**, v. 181, n. 21, p. 4895–4907, 2011.

YUAN, M. **Speech Recognition on DSP : Algorithm Optimization and Performance Analysis**. 2004. 105 f. Tese (Mestrado em Engenharia Eletrônica) – Chinese University of Honk Kong, Honk Kong, 2004. Disponível em: <<https://core.ac.uk/download/pdf/48536575.pdf>>. Acesso em: 12 de abr. 2019.

ZADEH, L. A. Fuzzy Sets. **Information and Control**, v. 8, n. 3, p. 338–353, 1965.

ZADEH, L. A. The birth and evolution of fuzzy logic. **International Journal of General Systems**, v. 17, n. 2–3, p. 95–105, 1990.

ZHAI, Y.; LI, M. C.; LUO, J.; ZHOU, Y.; LIU, L. Research of the motion balance of spherical mobile robot based on fuzzy control. **Journal Of Vibroengineering**, v. 17, n. 1, p. 13–23, 2015.

ZIEGLER, J.G.; NICHOLS, N.B. Optimum Settings for Automatic Controllers. **Transactions of the ASME**, v. 64, p. 759-768, 1942.

