

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

JEFFERSON WILLIAN FRANÇA GÓES

**APERFEIÇOAMENTO DE UM SISTEMA DE
RECONHECIMENTO DE PADRÕES DO ALFABETO DE
LIBRAS UTILIZANDO LUVA SENSORA**

TRABALHO DE CONCLUSÃO DE CURSO

PATO BRANCO

2019

JEFFERSON WILLIAN FRANÇA GÓES

**APERFEIÇOAMENTO DE UM SISTEMA DE
RECONHECIMENTO DE PADRÕES DO ALFABETO DE
LIBRAS UTILIZANDO LUVA SENSORA**

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2 , do Curso Superior de Engenharia de Computação do Departamento Acadêmico de Informática - DAINF - da Universidade Tecnológica Federal do Paraná - UTFPR, Câmpus Pato Branco, como requisito parcial para obtenção do título de "Engenheiro em Computação".

Orientador: Prof. Dr. Fábio Luiz Bertotti

Coorientador: Prof. Dr. Dalcimar Casanova

PATO BRANCO

2019



TERMO DE APROVAÇÃO

Às 15 horas e 50 minutos do dia 11 de julho de 2019, na sala V007, da Universidade Tecnológica Federal do Paraná, Campus Pato Branco, reuniu-se a banca examinadora composta pelos professores Fábio Luiz Bertotti (orientador), Dalcimar Casanova (coorientador). César Rafael Torrico e Kathya Silvia Collazos Linares para avaliar o trabalho de conclusão de curso com o título **Aperfeiçoamento de um sistema de reconhecimento de padrões do alfabeto de LIBRAS utilizando luva sensora**, do aluno **Jefferson Willian França Góes**, matrícula 1376888, do curso de Engenharia de Computação. Após a apresentação do aluno foi arguido pela banca examinadora. Em seguida foi realizada a deliberação pela banca examinadora que considerou o trabalho aprovado.

Prof. Fábio Luiz Bertotti
Orientador (UTFPR)

Prof. Dalcimar Casanova
Coorientador (UTFPR)

Prof. César Rafael Torrico
(UTFPR)

Profa. Kathya Silvia Collazos Linares
(UTFPR)

Profa. Beatriz Terezinha Borsoi
Coordenador de TCC

Prof. Pablo Gauterio Cavalcanti
Coordenador do Curso de
Engenharia de Computação

A Folha de Aprovação assinada encontra-se na Coordenação do Curso.

Um computador pode ser chamado de inteligente se ele pode enganar uma pessoa a pensar que é um ser humano.

Alan Turing

AGRADECIMENTOS

Primeiramente agradeço ao meu orientador pela sugestão de um trabalho que já foi feito, dando continuidade aos trabalhos feitos por outros acadêmicos. Também agradeço ao meu coorientador pelo suporte dado em relação as redes neurais e como fazer tudo isso funcionar. Agradecendo também a minha namorada pelas correções fornecidas no trabalho e estar ao meu lado nessa jornada que também se estende à minha família e amigos que me aguentaram e me ajudaram nesta fase difícil da faculdade. E por último mas não menos importante um agradecimento aos professores em geral que me fizeram evoluir e descobrir mais sobre eu mesmo para superar todos os desafios.

RESUMO

Língua Brasileira de Sinais (LIBRAS) é uma língua gestual-visual amplamente utilizada por surdos no Brasil. No ensino de LIBRAS, a interação com o objeto de ensino pode potencializar o aprendizado. A partir dessa premissa, acredita-se que um sistema de detecção do alfabeto de LIBRAS pode contribuir no processo de aprendizagem, assim como na tradução da linguagem em elementos textuais. Desta forma, este trabalho apresenta o aperfeiçoamento de um sistema para reconhecimento de padrões do alfabeto de LIBRAS utilizando uma luva sensora previamente concebida. Como resultado, obteve-se um sistema que utiliza redes neurais capaz de reconhecer todas as letras do alfabeto, com um percentual de acerto de 84

Palavras-chave: Redes Neurais. Microcontroladores. Processamento de sinais. LIBRAS.

ABSTRACT

The Brazilian Sign Language (LIBRAS) it's a gestural-visual language used by deaf people in Brazil. In the teaching of LIBRAS, an interaction with the object of teaching can potentiate learning. From this premise, it is believed that a system of detection of the alphabet of LIBRAS can contribute in the learning process, as well as in the translation of the language into textual elements. Thus, this work presents the improvement of a system for recognizing patterns of the alphabet of LIBRAS using a previously conceived sensor glove. As a result, we obtained a system that uses neural networks capable of recognizing all the letters of the alphabet, with a correct percentage of 84

Keywords: Neural Network. Microcontroller. Signal Processing. LIBRAS.

LISTA DE FIGURAS

Figura 1:	Alfabeto manual de LIBRAS	15
Figura 2:	Comparação entre datilologia e sinal pronto da palavra certo . .	16
Figura 3:	Luva com sensores de fibra ótica da Five Dimension Tecnologies	17
Figura 4:	Luva de dados CyberGlove III da Cyber Glove Systems LLC . .	18
Figura 5:	Luva de dados IGS Cobra da Synertial	18
Figura 6:	Mapa de sensores e geradores de sinal na luva	20
Figura 7:	Bobina construída para luva	20
Figura 8:	Vista frontal e lateral da luva sensora desenvolvida	21
Figura 9:	Diagrama de blocos do sistema desenvolvido por Lazzarotto (2016, p.65)	22
Figura 10:	Modelo de um acelerômetro que consiste em uma massa sísmica, um elemento mola e um elemento amortecimento	24
Figura 11:	Aeronave com o sistema de coordenadas baseado nas rotações	25
Figura 12:	Sensor inercial com o eixo de <i>Roll</i> definido	26
Figura 13:	Exemplo de efeito de indução mútua	29
Figura 14:	Topologia dos circuitos de sensoriamento	30
Figura 15:	Modelo de sistemas de classificações de padrões	31
Figura 16:	Modelo não linear de um neurônio	32
Figura 17:	Função de ativação limiar	34
Figura 18:	Função de ativação sigmoide	34
Figura 19:	Arquitetura de camada única	35
Figura 20:	Arquitetura de múltiplas camadas totalmente conectada	35
Figura 21:	Diagrama de aprendizado supervisionado	36
Figura 22:	Microcontrolador STM32F103C8T6	39
Figura 23:	Display LCD 16x2 de cristal líquido HD44780	40

Figura 24:	Sensor inercial MPU6050	40
Figura 25:	Luva sensora desenvolvida por Lazzarotto (2016)	41
Figura 26:	Placa para recepção e processamento dos sinais vindos do sen- sor indutivo	41
Figura 27:	Pinos do microcontrolador configurados no <i>software STM32CubeMX</i>	42
Figura 28:	Fluxograma que detalha a rotina desenvolvida para calibrar os sensores indutivos	43
Figura 29:	Valor dos sensores indutivos ao fazer o gesto da letra A em LI- BRAS no <i>software STM32Studio</i>	44
Figura 30:	Fluxograma que detalha o funcionamento da rotina de coleta de dados do sensor indutivo.	45
Figura 31:	Vista das ligações da placa de adaptação para o microcontrola- dor <i>stm32f103c8t6</i>	46
Figura 32:	Vista de cima da placa de adaptação para o microcontrolador <i>stm32f103c8t6</i>	47
Figura 33:	Posicionamento da MPU6050 e seus eixos para a calibração . .	48
Figura 34:	Posicionamento do sensor inercial na luva com eixo de coord- enadas.	49
Figura 35:	Letras do alfabeto de libras que contém movimento.	49
Figura 36:	Gráficos dos sinais do acelerômetro, amostrando 500 ms.	50
Figura 37:	Janela de amostras	50
Figura 38:	Fluxograma que descreve o comportamento do <i>firmware</i> a ser desenvolvido.	53
Figura 39:	Conjunto ao final do trabalho.	54
Figura 40:	Demonstração da letra C.	55

LISTA DE TABELAS

1	Relação de sensores e geradores com as variáveis da Figura 29	44
2	Matriz de confusão das letras sem conflito	56
3	Matriz de confusão das letras conflitantes	56

LISTA DE SIGLAS

ADC	Conversor Analógico Digital
fem	Força Eletromotriz
MEMS	Sistemas microeletromecânicos
MLP	Perceptron Multicamada

SUMÁRIO

1 CONSIDERAÇÕES INICIAIS	13
1.1 OBJETIVOS	14
1.1.1 Objetivo Geral	14
1.1.2 Objetivos Específicos	14
2 REFERENCIAL TEÓRICO	15
2.1 LIBRAS	15
2.2 LUVAS SENSORAS	16
2.2.1 Luva com sensores de fibra ótica	17
2.2.2 Luva com sensores de flexão	17
2.3 LUVA COM SENsoRES INERCIAS	18
2.3.1 Luva com sensores indutivos	19
2.4 TRABALHO PREVIAMENTE DESENVOLVIDO	19
2.4.1 Construção das bobinas	19
2.4.2 Circuito de condicionamento de sinais	21
2.5 ACELERÔMETRO	23
2.5.1 Funcionamento	23
2.5.2 Eixos de rotação	24
2.6 SENSORES INDUTIVOS	28
2.7 RECONHECIMENTO DE PADRÕES	30
2.8 REDES NEURAIS	32
2.8.1 Arquitetura de um neurônio	32
2.8.2 Funções de ativação	33
2.8.3 Arquitetura de uma rede neural	34
2.8.4 Aprendizagem supervisionada	36
2.8.5 O algoritmo <i>backpropagation</i>	36
3 MATERIAIS E MÉTODOS	38
3.1 MATERIAIS	38

3.1.1	Placa de desenvolvimento com microcontrolador STM32F103C8T6	38
3.1.2	Módulo <i>display</i> LCD 16x2 HD44780	39
3.1.3	Sensor inercial MPU-6050	40
3.1.4	Conjunto de luva e circuito de condicionamento de sinais	41
3.2	MÉTODO	42
3.2.1	Transferência do trabalho para o novo microcontrolador	42
3.2.2	Placa de adaptação para novo microcontrolador	46
3.2.3	Configuração sensor MPU6050 e Processamento de sinais	47
3.2.3.1	Configuração e calibração do sensor MPU-6050	47
3.2.3.2	Letras conflitantes e processamento dos sinais	48
3.2.4	Redes Neurais Implementadas	50
3.2.4.1	Rede Neural para os sensores indutivos	51
3.2.4.2	Rede Neural sensor inercial	51
4	RESULTADOS	54
4.1	SISTEMA COMPLETO.....	54
5	CONCLUSÕES.....	58
	APÊNDICE A - CÓDIGOS DESENVOLVIDOS	60

1 CONSIDERAÇÕES INICIAIS

A linguagem de sinais (LS), é uma idioma não verbal, altamente desenvolvida e organizada, na qual indivíduo usa expressões faciais e gestos com as mãos para se expressar. Nem todas as pessoas conseguem entender e compreender uma linguagem baseada em sinais, os que compreendem, geralmente, são membros da família ou membros da comunidade surda (DAS *et al.*, 2016).

Para Castro e Carvalho (2009), a Língua Brasileira de Sinais (LIBRAS) é considerado como um idioma que possui uma estrutura gramatical própria, contendo particularidades idiomáticas e variações regionais, as quais assemelham-se ao sotaque ou gírias. Contudo é importante ressaltar a diferença entre LIBRAS e o alfabeto manual, o qual é um recurso para os falantes da língua de sinais que auxilia na representação de siglas, lugares ou algum vocabulário para o qual não existe sinal definido (GESER, 2009).

Considerando as dificuldades de interação entre as pessoas que falam e os surdos, o protótipo desenvolvido como resultado deste trabalho visa prover uma forma de interação mais efetiva entre essas pessoas utilizando o alfabeto de sinais.

O propósito deste trabalho se deriva do projeto já implementado, o qual tinha por objetivo a elaboração de uma luva com sensores indutivos para detecção do alfabeto manual de LIBRAS. O sucesso em detectar letras as quais não possuem movimento serão levados em conta na implementação de um sensor inercial (LAZZAROTTO, 2016, p. 95).

A detecção do alfabeto manual somente com sensores indutivos, dispostos em uma luva sensora construído por Lazzarotto (2016), possui limitações quando há padrões com movimento, sendo necessária a inclusão de um sensor inercial na luva para o reconhecimento de padrões com movimento.

Levando em consideração o que já foi desenvolvido, ao introduzir um sensor do tipo inercial ao sistema, a captura de dados é sensível ao tempo, e os dados devem ser processados. O microcontrolador atual (MSP430) não suportaria tal carga, sendo necessário o *upgrade* para um ARM Cortex-M3. Considerando que serão classificados há uma necessidade de modificar a rede neural já implementada.

1.1 OBJETIVOS

1.1.1 OBJETIVO GERAL

Aperfeiçoar um sistema para detecção de padrões do alfabeto de LIBRAS por meio de luva sensora, adicionando um sensor inercial e promovendo melhorias de hardware necessárias para suportar o processamento dos dados do sensor inercial.

1.1.2 OBJETIVOS ESPECÍFICOS

- Portar um código existente para o microcontrolador STM32F103C8T6;
- Implementar o sensor inercial MPU-6050;
- Coletar dados para treinamento e validação de um algoritmo de classificação;
- Utilizar algoritmos de aprendizado em Python para realizar treinamento e simulações;
- Implementar o algoritmo de classificação no microcontrolador;
- Desenvolver uma interface para visualização dos resultados da classificação;
- Testar e validar o sistema a partir de outros usuários.

2 REFERENCIAL TEÓRICO

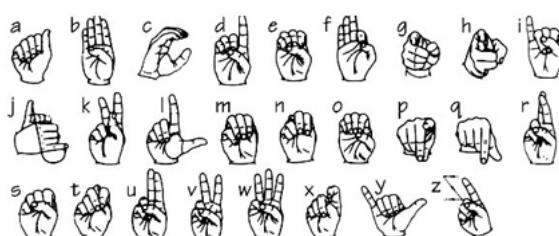
2.1 LIBRAS

A Língua Brasileira de Sinais (LIBRAS) é um idioma que possui uma estrutura gramatical própria, contendo particularidades idiomáticas e regionalismos, que se assemelham ao sotaque ou as gírias da língua portuguesa. Cada país fala uma língua diferente, nos Estados Unidos, por exemplo, é utilizada a *American Sign Language* (ASL), com seu vocabulário derivado da Língua Francesa de Sinais há 180 anos, e LIBRAS, também é derivada da Língua Francesa de Sinais. Embora a Inglaterra e Estados Unidos tenham inglês como idioma principal, as línguas de sinais são completamente diferentes. Isto também serve para o português e LIBRAS, onde LIBRAS não é o português sinalizado.(CASTRO; CARVALHO, 2009).

Para Zieren *et al.* (2006), em linguagens de sinais em geral, a informação é gerada por combinação dos seguintes fatores: configuração de mão; posicionamento da mão ou ponto de articulação; movimento; orientação (direção do sinal); e expressão facial e corporal. Alguns sinais podem ser distinguidos pela configuração de mão somente, e em outros casos é a combinação de todos os fatores que devem ser executados evitando uma interpretação ambígua.

Em alguns casos, não existem sinais que expressem siglas, nomes, ações ou gírias. Então é desenvolvido de um alfabeto manual (vide Figura 1) que é constituído de configuração de mão que representa uma letra e, em alguns casos, um movimento. Através da datilologia ou soletração digital, este alfabeto é utilizado para expressar essas palavras para as quais não existe um sinal equivalente (FERREIRA, 2010).

Figura 1: Alfabeto manual de LIBRAS



Fonte: Castro e Carvalho (2009).

Como a Figura 2 apresenta, a palavra "certo" que em datilologia exige cinco gestos para definir o seu significado, enquanto utilizando o sinal existente, pode ser definida apenas com um gesto.

Figura 2: Comparação entre datilologia e sinal pronto da palavra certo



Fonte: Ferreira (2010, p. 22).

Existem ainda, muitas limitações no processo de comunicação, entre surdos e ouvintes em ambientes como trabalho, escolas, universidades e no próprio ambiente familiar de surdos, provocando um distanciamento (SCHIMIGUEL *et al.*, 2014), que podem ser minimizadas, gradativamente, pelo uso de objetos de aprendizagem (COSTA MÁRCIA MACHADO MARINHO, 2017).

Conforme Martins (2015), as tecnologias podem facilitar a inserção comunicativa dos surdos, assim como para os ouvintes, com o uso frequente, por exemplo, das redes sociais, que apesar de primar pelo lazer, possibilitam um maior contato com o português, uso de dicionários on-line e os objetos de aprendizagem (hipermídias, softwares, jogos etc), permitindo aos surdos, o contato com materiais mais interessantes e atrativos.

2.2 LUVAS SENSORAS

Segundo Fahn e Sun (2005) mapeamento de mão é a tecnologia mais popular para entrada de dados na Realidade Virtual (VR). As entradas baseadas em luva permitem que o usuário aplique a sua coordenação e habilidade para as atividades de VR. Os tipos de luvas abordados são as luvas mais expressivas no mercado, e a luva indutiva que é a luva utilizada neste trabalho.

2.2.1 LUVA COM SENSORES DE FIBRA ÓTICA

As luvas de dados baseadas em sensores de fibra ótica usam um LED emissor de infravermelho como fonte, com o sinal sendo conduzido por um tubo flexível de fibra. Então, uma fotocélula é colocado no outro lado e mede a intensidade do sinal enviado via fibra (FAHN; SUN, 2005).

A luva de dados 5DT Data Glove 16 da empresa Five Dimentions Technologies, demonstrada na Figura 3, com 15 sensores de fibra ótica, 2 em cada dedo e 1 sensor na junta do dedo. E um último, um sensor inercial para determinar os movimentos circulares com a luva. Contém também Conversor A/D (Analógico para digital) de até 12 bits para cada sensor. Da luva inteira (14 sensores) podem ser obtidas até 100 amostras por segundo (5DT, 2018).

Figura 3: Luva com sensores de fibra ótica da Five Dimension Technologies



Fonte: 5DT (2018).

2.2.2 LUVA COM SENSORES DE FLEXÃO

Os sensores de flexão utilizam um material que varia a resistência de acordo com o grau da dobra realizada, utilizando uma escala de 0 a 90 graus. Em luvas que utilizam esse tipo de sensor são colocados cinco sensores, um para cada dedo. Quando os dedos são flexionados, a resistência do sensor varia resultando em uma tensão equivalente ao grau de flexão de cada dedo. (FAHN; SUN, 2005).

A empresa CyberGlove Systems produz uma série de luvas para o desenvolvimento de animações e captura de movimentos, entre elas, a CyberGlove III que pode ser vista na Figura 4, a qual faz uso de 22 sensores de flexão com resolução menor do que um grau. Também dispõe de interface Wi-Fi e conexão para cartão USB (CYBERGLOVE, 2018).

Figura 4: Luva de dados CyberGlove III da Cyber Glove Systems LLC



Fonte: CyberGlove (2018).

2.3 LUVA COM SENsoRES INERcIAIS

Em luvas com sensores inerciais, os sensores são calibrados em um mesmo ponto onde é considerado o eixo zero, com base nessa calibragem são utilizados 2 sensores entre uma junta para medir o ângulo das duas partes do dedo em relação a junta (BAKHSHI *et al.*, 2011).

Um exemplo deste tipo é a IGS Cobra da empresa Synertial(vide 5), com 16 sensores inerciais com 9 eixos de liberdade (acelerômetro, giroscópio e magnetômetro) (SYNERTIAL, 2018).

Figura 5: Luva de dados IGS Cobra da Synertial



Fonte: Synertial (2018)

2.3.1 LUVA COM SENSORES INDUTIVOS

Uma luva indutiva ou luva com sensores indutivos utiliza um sensor de rastreamento magnético, onde uma fonte irradia um campo magnético e um pequeno sensor retorna a posição e orientação em relação a fonte magnética. O princípio de funcionamento será abordado na seção 2.6.

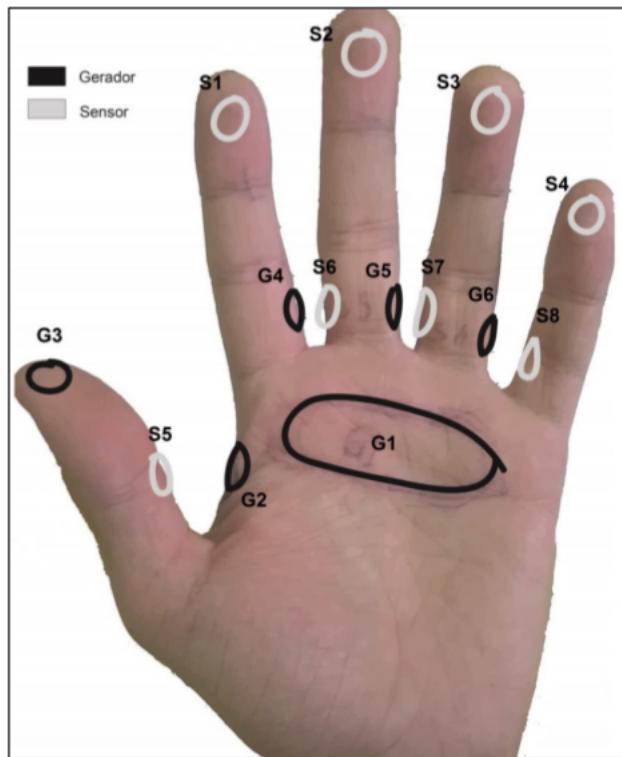
2.4 TRABALHO PREVIAMENTE DESENVOLVIDO

O trabalho elaborado por Lazzarotto (2016) tinha como objetivo geral: Desenvolver um dispositivo do tipo luva sensora capaz de reconhecer configurações de mão, correspondente ao alfabeto manual de LIBRAS, e realizar a tradução destas para o seu significado na Língua Portuguesa, apresentando o resultado na forma de texto em um *display*. A luva foi desenvolvida a partir de sensores indutivos, incluindo circuito de excitação para as bobinas, assim como circuito de leitura e sistema embarcado usando o microcontrolador MSP430F5529.

2.4.1 CONSTRUÇÃO DAS BOBINAS

Com base no trabalho de Fahn e Sun (2005), foi possível obter um método para construir uma luva com sensores indutivos que tem o funcionamento explicado na seção 2.6. Para obter a posição dos sensores na luva foi observado diferentes sinais de LIBRAS e como eles se diferem, por exemplo U e V, somente o ângulo entre o dedo indicador e dedo médio pode diferenciar essas letras. Considerando todas as letras e suas posições de mão, foi obtido o seguinte esquema de sensores e geradores, como demonstrado na Figura 6.

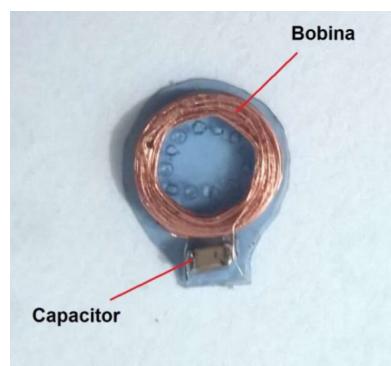
Figura 6: Mapa de sensores e geradores de sinal na luva



Fonte: Lazzarotto (2016, p. 68).

A espessura das bobinas foi definida para que não atrapalhasse no movimento normal da mão ao vestir a luva, tendo 14 mm de diâmetro. Todas as bobinas foram construídas a mão e com o mesmo número de espiras. Naturalmente que devido ao processo manual de fabricação há uma variação nas indutâncias. A frequência ressonante do circuito LC foi definida em torno de 100 kHz, projetando o resto dos componentes a partir disso. Uma das bobinas construídas é exibida na Figura 7.

Figura 7: Bobina construída para luva



Fonte: Lazzarotto (2016, p. 71).

O resultado é apresentado na Figura 8, onde é mostrada a luva construída com os sensores indutivos.

Figura 8: Vista frontal e lateral da luva sensora desenvolvida

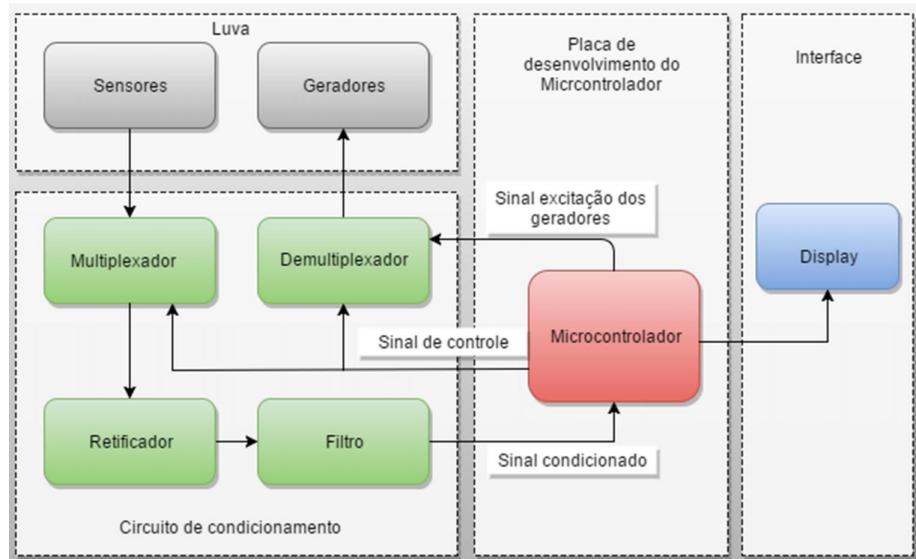


Fonte: Lazzarotto (2016, p. 87).

2.4.2 CIRCUITO DE CONDICIONAMENTO DE SINAIS

Para o funcionamento do sistema como um todo, todos os sinais provenientes dos sensores indutivos devem ser tratados para a leitura correta no conversor analógico digital (ADC) do microcontrolador. Na figura 9 é possível ver todo o tratamento de *hardware* feito para o processamento dos sinais dos sensores indutivos.

Figura 9: Diagrama de blocos do sistema desenvolvido por Lazzarotto (2016, p.65)



Fonte: Lazzarotto (2016, p. 65).

A modelagem do circuito de aquisição e processamento dos sinais foi proposta por Fahn e Sun (2005), o qual foi adaptado para simplificar o circuito. Uma das modificações foi substituir o gerador senoidal e deixar o microcontrolador gerar uma onda quadrada, considerando que a mudança de forma de onda não interfere significativamente na resposta dos sensores, sendo que é a frequência que determina a ressonância dos geradores (LAZZAROTTO, 2016).

Um circuito retificador a envoltória do sinal CC é necessário para converter um sinal CA em um sinal CC. Para que a saída represente o pico do sinal CC, é necessário empregar um detector de pico seguido de um filtro passa-baixas (BOYLESTAD; NASHELSKY, 2013).

De acordo com (MALVINO; BATES, 2007), um filtro permite a passagem de uma faixa de frequências enquanto rejeita outra. Filtros podem separar os sinais desejados dos indesejados, bloquear sinais de interferência e modificar sinais. Existem os filtros passivos que são constituídos de resistores, capacitores e indutores. Também existem os ativos, que são compostos de resistores, capacitores e amplificadores operacionais ou amp-ops.

O uso de amp-ops nos filtros ativos pode eliminar a necessidade dos volumosos indutores, que não são satisfatórios quando integrados aos circuitos. Em contrapartida, filtros ativos não atenuam necessariamente os sinais sobre a banda passante, como acontece com seus similares feitos com elementos passivos (CATHEY, 1994).

Como são utilizadas várias bobinas, utiliza-se de uma técnica para não ter leituras afetadas por interferência, utilizando o método de divisão de tempo, onde cada bobina geradora é acionada sozinha por um período de tempo. O sinal de saída do multiplexador deve passar por um processo de condicionamento, primeiramente por um atenuador, para que o valor de amplitude se fique adequado ao valor de alimentação dos amplificadores operacionais (LAZZAROTTO, 2016).

Em seguida, o sinal passa por um circuito retificador, uma vez que a informação que possibilita estimar a posição das bobinas é o modulo da fem induzida nas bobinas dos sensores. O sinal retificado é filtrado para eliminar *ripple* do sinal, e em seguida passa por ajustes de amplitude para ser conectado ao ADC do microcontrolador.

2.5 ACELERÔMETRO

Um acelerômetro é um componente eletrônico que mede as forças exercidas num determinado objeto. Essas forças podem ser de dois tipos: estáticas ou dinâmicas, sendo estática a força da aceleração gravitacional (constante), e dinâmicas causadas pelo movimento ou vibração provocados no acelerômetro (DUARTE, 2013, p. 20).

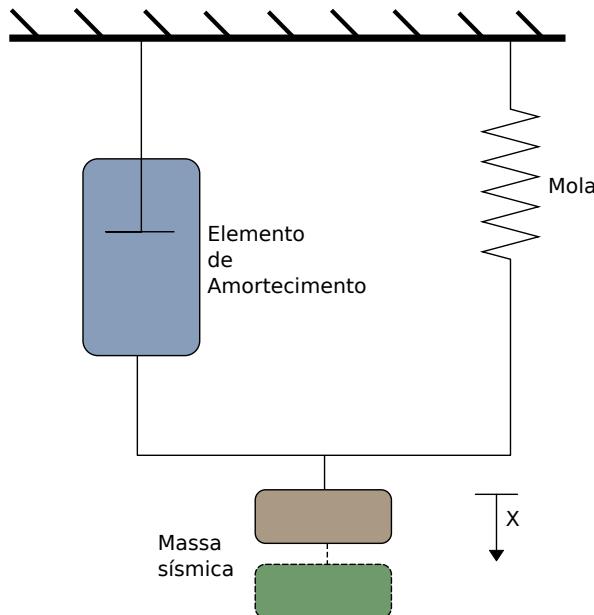
2.5.1 FUNCIONAMENTO

O princípio básico de um acelerômetro é a ação de aceleração de uma massa para produzir força, conforme descrito na segunda lei de Newton (vide Equação 1):

$$F = m \times a, \quad (1)$$

onde F é a força em Newtons [N], m a massa em quilos [kg] e a aceleração em m/s^2 . Portanto, é necessário a utilização de uma massa nos acelerômetros, chamada massa inercial para medir a aceleração (BALBINOT; VALNER, 2007, p. 271). O acelerômetro pode ser modelado matematicamente como mostra na Figura 10.

Figura 10: Modelo de um acelerômetro que consiste em uma massa sísmica, um elemento mola e um elemento amortecimento



Fonte: Balbinot e Valner (2007, p. 271).

Conforme a Figura 10, uma força pode ser aplicada a massa de acordo com a segunda lei de Newton, e o esquema pode ser modelado conforme (BALBINOT; VALNER, 2007):

$$\frac{x(s)}{a(s)} = \frac{1}{s^2 + \frac{b}{m} + s\frac{K}{m}}, \quad (2)$$

em que s representa o operador de Laplace, x o deslocamento da massa de sua posição de repouso, a a aceleração a ser mensurada, b o coeficiente de amortecimento, m a massa destinada ao movimento e K a constante da mola.

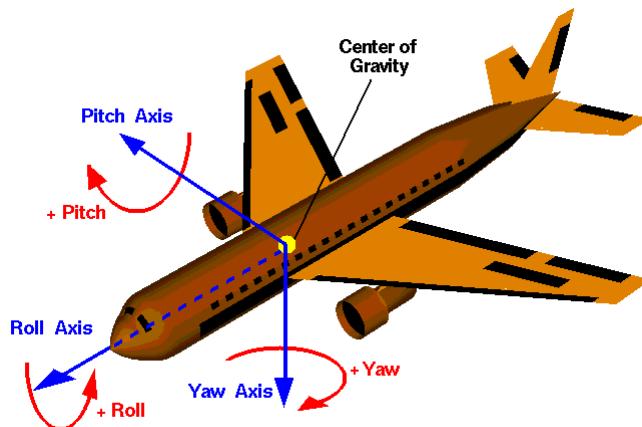
Os acelerômetros são encontrados em diversos tamanhos e tecnologias, entre eles os piezoelétricos, piezorresistivos e capacitivos. Para reduzir o tamanho do dispositivo, são aplicadas técnicas da área de microeletrônica, resultando em um dispositivo classificado como sistema microeletromecânico ou MEMS (Micro-Electro-Mechanical System). (BALBINOT; VALNER, 2007).

2.5.2 EIXOS DE ROTAÇÃO

Conforme Glenn Research Center (2015) uma aeronave em pleno voo irá rotacionar em torno do seu centro de gravidade, que é o ponto de onde se localiza a massa média da aeronave. É possível definir um sistema de coordenadas tridimensional baseado no centro de gravidade onde, cada eixo é perpendicular a outros dois

eixos. Logo é possível definir a orientação pela quantidade de rotação das partes da aeronaves em relação aos eixos principais (GLENN RESEARCH CENTER, 2015). Na Figura 11 observa-se uma aeronave com o sistema de coordenadas baseados nas rotações.

Figura 11: Aeronave com o sistema de coordenadas baseado nas rotações

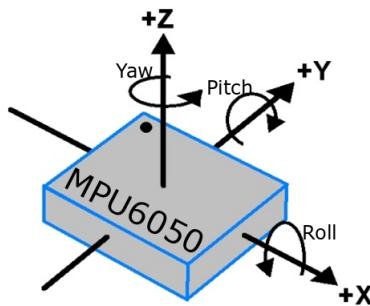


Fonte: Glenn Research Center (2015).

O eixo nomeado de *Yaw* (ψ) é definido como perpendicular as asas do avião, tem origem no centro de gravidade e é direcionado para baixo (via Figura 11). O eixo *Pitch* (θ) é perpendicular ao eixo *Yaw* e paralelo as asas da aeronave, com origem no centro de gravidade e tem a direção das asas. O eixo *Roll* (ϕ) é perpendicular aos outros eixos, com origem no centro de gravidade e é direcionado ao nariz da aeronave.

Observando estes eixos e definindo no sensor inercial, para distinguir as letras que possuem movimento será necessário somente o eixo de rotação *Roll*, como pode ser observado na Figura 12.

Figura 12: Sensor inercial com o eixo de *Roll* definido



Fonte: Adaptado de ElectronicWings (2019).

De acordo com Freescale Semiconductor (2013), o acelerômetro dos celulares é utilizado para definir a orientação da tela utilizando o sistema de coordenadas tridimensionais. Logo é possível obter os eixos *Pitch* e *Roll* a partir dos valores dos 3 eixos do acelerômetro.

Assumindo que o acelerômetro é orientado no campo gravitacional da terra g e passa por aceleração linear a_r medido no *frame* de referência da terra r , terá uma saída dada por (FREESCALE SEMICONDUCTOR, 2013):

$$G_p = \begin{pmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{pmatrix} = R(g - a_r), \quad (3)$$

onde R é a matriz de rotação que descreve a orientação do acelerômetro com relação as coordenadas da terra.

Supõe-se que o acelerômetro não tem aceleração linear (necessário assumir isso para resolver a Equação 3 para a matriz de rotação R), quando houver presença de aceleração linear será adicionado um erro na estimativa da orientação. Também assume-se que a orientação inicial do acelerômetro é plana com o campo gravitacional da terra e alinhado com o eixo z . Considerando isso, a saída G_p do acelerômetro é demonstrada por (FREESCALE SEMICONDUCTOR, 2013) (medido em g):

$$G_p = \begin{pmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{pmatrix} = Rg = R \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad (4)$$

Para determinar os ângulos *Yaw*, *Pitch* e *Roll*, é necessário descrever as componentes da matriz de rotação R . As matrizes de rotação $\text{Yaw}(R_z(\psi))$, $\text{Pitch}(R_y(\theta))$ e $\text{Roll} (R_x(\phi))$ transformam o vetor G_p sobre uma rotação do sistema de coordenadas da Figura 12 para ângulos de *Roll* (ϕ), *Pitch* (θ), e *Yaw* (ψ). Essas matrizes são expressas como (FREESCALE SEMICONDUCTOR, 2013):

$$R_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix}, \quad (5)$$

$$R_y(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix}, \quad (6)$$

$$R_z(\psi) = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (7)$$

Existem seis possibilidades para ordenar essas três matrizes de rotação. Em princípio, todas são igualmente válidas (FREESCALE SEMICONDUCTOR, 2013). Considerando isso e inicialmente o valor de $1g$ no eixo z , resulta em (FREESCALE SEMICONDUCTOR, 2013):

$$R_{xyz} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = R_x(\phi)R_y(\theta)R_z(\psi) \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}. \quad (8)$$

Fazendo a substituição e a multiplicação das matrizes das Equações 5, 6, e 7 na Equação 8 demonstra-se que (FREESCALE SEMICONDUCTOR, 2013):

$$\begin{pmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \cos\phi\sin\psi & \cos\phi\cos\psi + \sin\theta\sin\phi\sin\psi & \cos\theta\sin\phi \\ \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \cos\psi\sin\phi & \cos\theta\cos\phi \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \quad (9)$$

efetuando a multiplicação resulta na matriz:

$$= \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix}. \quad (10)$$

Reescrevendo 10 com base no vetor G_p resulta em:

$$\frac{G_p}{\|G_p\|} = \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix} \Rightarrow \frac{1}{\sqrt{G_{px}^2 + G_{py}^2 + G_{pz}^2}} \begin{pmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{pmatrix} = \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix}, \quad (11)$$

Resolvendo 11 para o eixo necessário para este trabalho, que é o *Roll*, tem-se:

$$\tan_{xyz}\phi = \frac{G_{py}}{G_{pz}}. \quad (12)$$

Utilizando os valores de $G_{py} = 0,082198$ e $G_{pz} = -0,887432$, a resolução de 12 pode ser tanto -5.29 ou 174.71 , uma vez que os valores de *Roll* variam de -180° a 180 . Para solucionar este problema se utiliza da função atan2, que automaticamente retorna o ângulo em radianos no quadrante correto baseado no sinal dos dois argumentos (FREESCALE SEMICONDUCTOR, 2013). Finalmente, chega-se a expressão para calcular o *Roll* do acelerômetro:

$$\phi_{xyz} = \text{atan2}(G_{py}, G_{pz}). \quad (13)$$

2.6 SENSORES INDUTIVOS

Como apresentado por Balbinot e Valner (2007), sensores indutivos são dispositivos sem contato e utilizados geralmente para medição de posição. Utilizando o efeito de indução eletromagnética em um corpo condutor, é possível produzir corrente elétrica quando o campo magnético é variável ou quando é efetuado movimento em um campo estático. Considerando uma espira C, a corrente induzida pode ser representada por (NUSSENZVEIG, 1996):

$$i = \frac{-1}{R} \frac{d\Phi_c}{dt}, \quad (14)$$

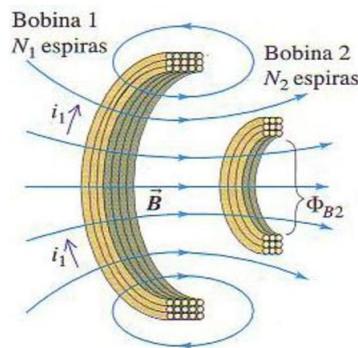
onde i é a corrente induzida, R é a resistência de uma espira C, e $\frac{d\Phi_c}{dt}$ é a variação de campo que atravessa a espira C. A existência dessa corrente na espira está associada uma força eletromotriz (fem) induzida descrita por (NUSSENZVEIG, 1996):

$$\varepsilon = Ri = -\frac{d\Phi_c}{dt}, \quad (15)$$

em que ε é a fem induzida.

Como neste trabalho são utilizadas pares de bobinas geradoras e sensoras é utilizado o efeito de indução mútua que é apresentado por Young e Freedman (2009). Considerando duas bobinas vizinhas (vide Figura 13). Uma corrente i_1 circulando na bobina 1 produz campo magnético \vec{B} e um fluxo magnético através da bobina 2. Quando i_1 varia, o fluxo magnético através da bobina 2 também varia, produzindo uma fem.

Figura 13: Exemplo de efeito de indução mútua



Fonte: Young e Freedman (2009, p. 317).

A fem induzida na bobina 2 pode ser representada em termos da variação de corrente na bobina 1, conforme (YOUNG; FREEDMAN, 2009):

$$\varepsilon_2 = -M \frac{di_1}{dt}, \quad (16)$$

onde M é conhecida como indução mútua e depende das indutâncias das bobinas 1 e 2, assim como do acoplamento entre elas. Em termos de fluxos magnéticos, a indutância mútua pode ser obtida a partir de (YOUNG; FREEDMAN, 2009):

$$M = \frac{N_2 \Phi_{B2}}{i_1} = \frac{N_1 \Phi_{B1}}{i_2}, \quad (17)$$

em que N_1 e N_2 são os número de espiras das respectivas bobinas, Φ_{B1} e Φ_{B2} correspondem aos fluxos magnéticos nas bobinas 1 e 2 respectivamente, i_1 e i_2 são as correntes de cada uma das bobinas.

Este princípio da indução é aplicado em transformadores, com circuitos CA para aumentar ou diminuir uma tensão de entrada. Um dos problemas envolvidos em

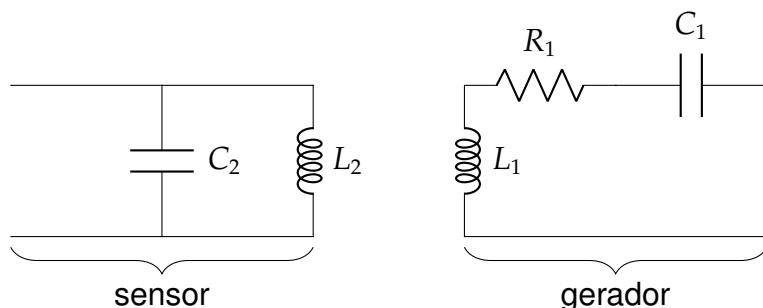
transformadores é que a potência fornecida pela bobina fonte depende da resistência da fonte de carga do circuito conectado a outra bobina (YOUNG; FREEDMAN, 2009).

Para solucionar este problema, considera-se a condição de ressonância, onde a impedância atinge o seu valor mínimo e a corrente atinge o valor máximo. Para isso, é necessário adicionar um capacitor ao circuito da bobina. Desta forma, a frequência de ressonância do circuito LC é dada por:

$$f_0 = \frac{1}{2\pi \sqrt{LC}}, \quad (18)$$

L é o indutor e C é o capacitor. A Figura 14 demonstra o esquemático do sensor e gerador projetados para operar em ressonância.

Figura 14: Topologia dos circuitos de sensoriamento



Fonte: Lazzarotto (2016, p. 67).

Para o projeto do valor de C , foi assumida a frequência de ressonância pré-definida de 100 kHz e a indutância das bobinas desenvolvidas por Lazzarotto (2016). Isolando-se C em 18 obtem-se:

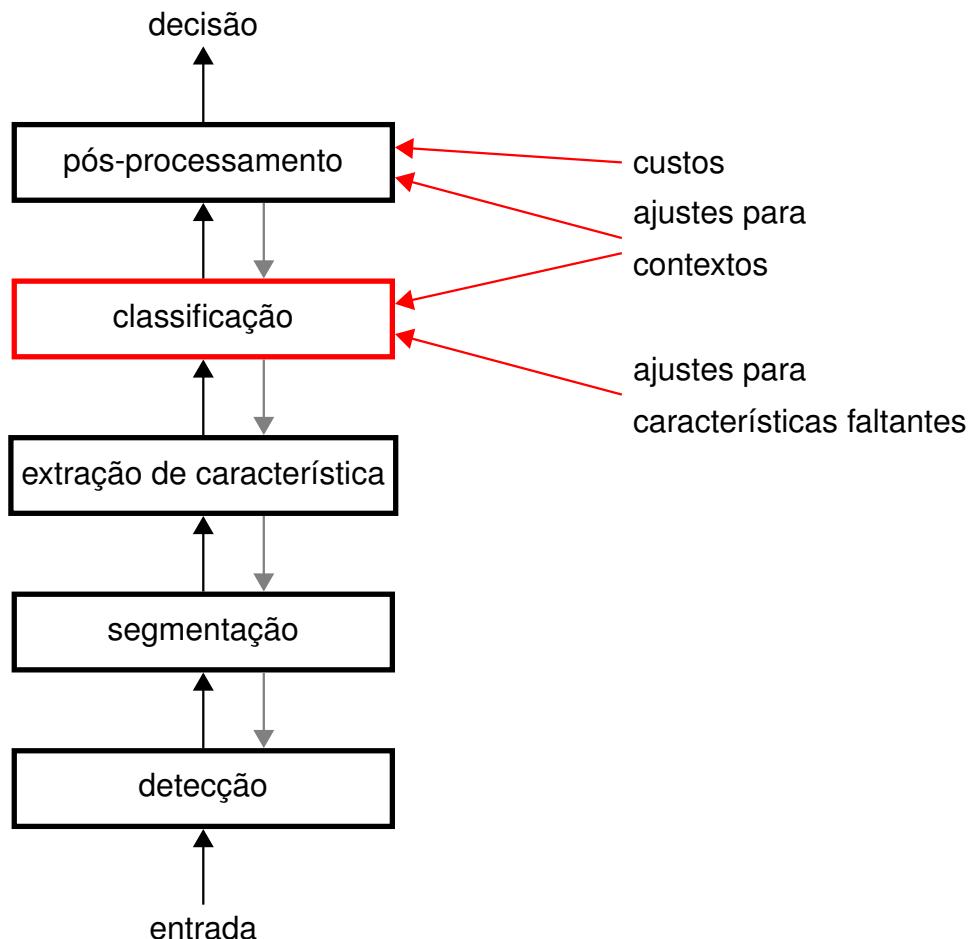
$$C = \frac{1}{4\pi^2 f_0^2 L}. \quad (19)$$

2.7 RECONHECIMENTO DE PADRÕES

Para Haykin (2008), o reconhecimento de padrões é formalmente definido como o processo que um determinado sinal/padrão recebido é classificado para uma das classes predefinidas. A Figura 15 exibe um diagrama dos componentes de um sistema típico de reconhecimento de padrões. Para entender o problema de projeto de um sistema, é necessário o entendimento dos problemas que cada um desses

componentes resolve (DUDA *et al.*, 2001).

Figura 15: Modelo de sistemas de classificações de padrões



Fonte: Duda *et al.* (2001).

A entrada corresponde a um fenômeno que se deseja avaliar. A detecção converte esse fenômeno em um sinal elétrico, por exemplo, sendo representado por um sensor que no caso do presente trabalho corresponde aos sensores indutivos e acelerômetro. As dificuldades variam de acordo com as limitações sensor, tais como resolução, sensibilidade, ruído, latência (DUDA *et al.*, 2001). A segmentação separa os objetos detectados pelo sensor de outros objetos ou do cenário. Tomando como base o sinal do acelerômetro, a segmentação é utilizada para separar onde começa o sinal de uma letra e onde termina (DUDA *et al.*, 2001).

O objetivo tradicional da extração de característica é caracterizar um objeto para ser identificado por valores que podem ser mensurados, em que, esses valores são muito similares para objetos de mesma categoria e muito distantes para objetos de categorias diferentes. Isso leva a ideia de procurar características marcantes que

são invariantes para transformações de entrada (DUDA *et al.*, 2001).

O papel do componente classificador em um sistema completo é usar as características escolhidas pela extração e atribuir este objeto a uma classe. Como a classificação perfeita é muitas vezes impossível, uma tarefa mais geral do classificador é definir a porcentagem que um objeto pertence a uma classe (DUDA *et al.*, 2001).

No pós-processamento são levados em conta as considerações, como os efeitos do contexto e o custo dos erros para determinar a ação apropriada. Uma simples medida de desempenho é a porcentagem de erros, a qual se refere a quantidade de classificações para uma classificação incorreta. Nesta etapa também pode ser analisada a eficiência de se utilizar múltiplos classificadores, adicionando um para a detecção dos gestos por uma câmera, por exemplo (DUDA *et al.*, 2001).

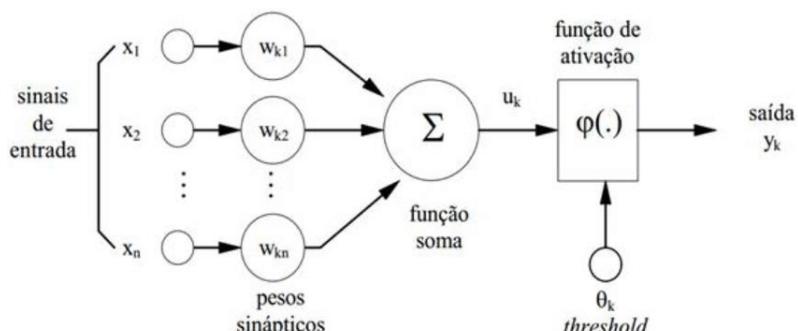
2.8 REDES NEURAIS

Conforme Silva *et al.* (2010), redes neurais são estruturas baseadas no sistema nervoso dos seres vivos que possuem a capacidade de armazenar e manter conhecimento.

2.8.1 ARQUITETURA DE UM NEURÔNIO

O neurônio é uma unidade de processamento de informação fundamental para a operação de uma rede neural, sendo a base para redes neurais artificiais (HAYKIN, 2001). Na Figura 16, pode-se notar a presença de três elementos: o conjunto de pesos sinápticos; função de ativação; e a junção aditiva.

Figura 16: Modelo não linear de um neurônio



Fonte: Haykin (2001, p. 38).

De acordo com Silva *et al.* (2010), os pesos sinápticos são os valores que ser-

virão para ponderar cada uma das variáveis de entrada da rede, permitindo quantificar as suas relevâncias em relação a funcionalidade do respectivo neurônio. As entradas, após serem ponderadas, são somadas pela junção aditiva ou combinador linear, que é um somador dos valores de entrada ponderados pelos respectivos pesos sinápticos. Esse processo é representado por (HAYKIN, 2001):

$$u_k = \sum_{j=1}^n w_{kj}x_j, \quad (20)$$

onde x_j são os sinais de entrada, w_{kj} são os pesos sinápticos, u_k é a saída do combinador linear.

A função de ativação ou função restritiva é aplicada para restringir a amplitude de saída de um neurônio. Tipicamente, a amplitude de saída de um neurônio é escrita como o intervalo unitário fechado $[0, 1]$ ou alternativamente $[-1, 1]$. A função de ativação recebe a saída do combinador linear, conforme (HAYKIN, 2001):

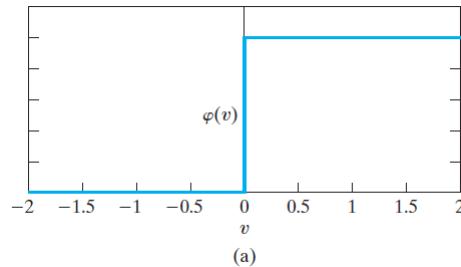
$$y_l = \varphi(u_k + b_k), \quad (21)$$

onde y_k é o sinal de saída do neurônio e b_k é o *bias*. O *bias* tem o efeito de aumentar ou diminuir a entrada líquida da função de ativação (HAYKIN, 2001). O valor do bias é ajustado da mesma forma que os pesos sinápticos. O bias possibilita que um neurônio apresente uma saída não nula mesmo que todas as suas entradas sejam nulas (LUDWIG; COSTA, 2007).

2.8.2 FUNÇÕES DE ATIVAÇÃO

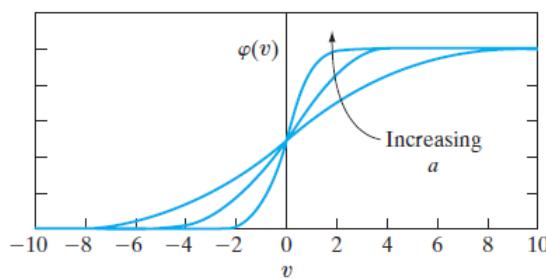
A função de ativação, representada por $\varphi(v)$, define a saída em termos do potencial de ativação v . Os principais tipos de funções de ativação consistem da função limiar, representada na Figura 17, e da função sigmoide, mostrada na Figura 18 (HAYKIN, 2001).

Figura 17: Função de ativação limiar



Fonte: Haykin (2001, p. 39).

Figura 18: Função de ativação sigmoide



Fonte: Haykin (2001, p. 39).

A função sigmoide é a forma mais comum de função de ativação para a construção de redes neurais artificiais, sendo definida como uma função estritamente crescente que exibe um balanceamento adequado entre comportamento linear e não linear. Dois exemplos de funções de ativação sigmoides são a função logística (HAYKIN, 2001):

$$\varphi(v) = \frac{1}{1 + \exp(-av)}, \quad (22)$$

e a função tangente hiperbólica:

$$\varphi(v) = \tanh(v). \quad (23)$$

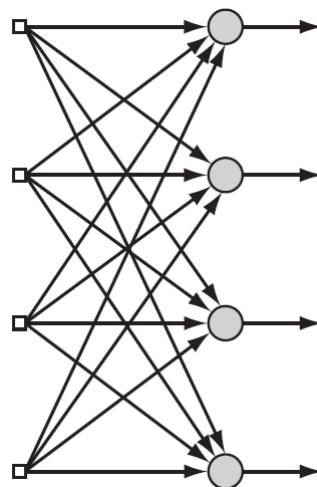
2.8.3 ARQUITETURA DE UMA REDE NEURAL

A arquitetura de uma rede neural artificial define a forma como seus diversos neurônios estão arranjados, uns em relação aos outros (SILVA *et al.*, 2010).

Nas redes *feedforward* camada única possuí uma camada de entrada e que se

projeta sobre a camada de saída dos neurônios, mas não vice-versa (HAYKIN, 2001). Essa rede é ilustrada na Figura 19.

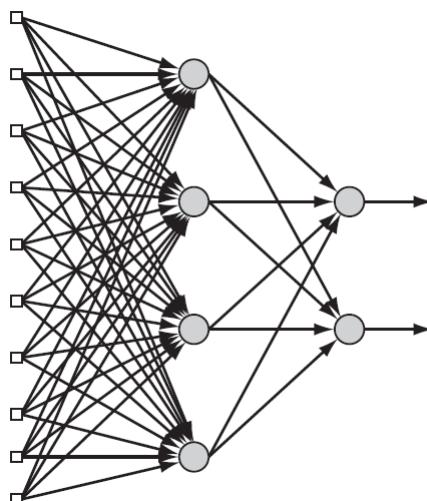
Figura 19: Arquitetura de camada única



Fonte: Haykin (2001, p. 47).

De acordo com Silva *et al.* (2010), as redes *feedforward* multicamadas são constituídas de uma ou mais camadas escondidas de neurônios, conforme ilustrado na Figura 20. A função dos neurônios ocultos é intervir entre a entrada externa e a saída da rede de maneira útil. Adicionando-se uma ou mais camadas ocultas, a rede torna-se capaz de extrair características de ordem elevada (HAYKIN, 2001).

Figura 20: Arquitetura de múltiplas camadas totalmente conectada



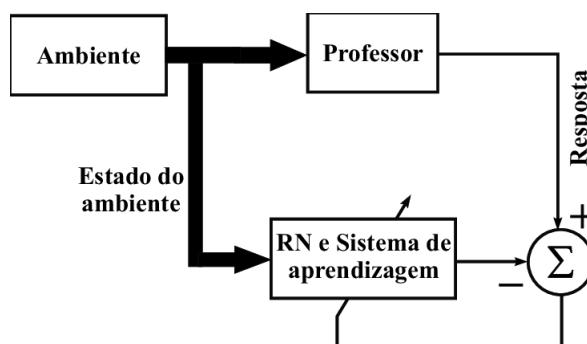
Fonte: Haykin (2001, p. 48).

2.8.4 APRENDIZAGEM SUPERVISIONADA

Conforme Silva *et al.* (2010), a estratégia de treinamento supervisionado ou aprendizagem supervisionada consiste em se ter disponível, considerando cada amostra dos sinais de entrada, as respectivas saídas desejadas. Suponha um professor que conhece o ambiente no qual a rede será exposta. Retirando ambos do ambiente e expondo-os ao um vetor de treinamento, o professor é capaz de fornecer a saída correta para a rede neural. Então os parâmetros da rede são ajustados sob a influência combinada do vetor de treinamento e do sinal de erro. O sinal de erro é a diferença entre a resposta desejada e a resposta da rede (HAYKIN, 2001).

Esse ajuste é feito passo a passo, transferindo o conhecimento do professor para a rede, da forma mais completa possível, e quando essa condição é alcançada, pode-se dispensar o professor e deixar a rede neural atuar sozinha (HAYKIN, 2001).

Figura 21: Diagrama de aprendizado supervisionado



Fonte: Haykin (2001, p. 88).

2.8.5 O ALGORITMO BACKPROPAGATION

Em resumo, o algoritmo de retropropagação pode ser descrito em cinco etapas:

- (i) Inicialização: arbitram-se valores aleatórios aos pesos sinápticos e níveis de *bias*, em uma distribuição uniforme, cuja média deve ser zero (LUDWIG; COSTA, 2007).
- (ii) Apresentação dos exemplos: apresenta-se uma época de exemplos a rede. Para cada exemplo, realiza-se uma propagação dos sinais e a retropropagação dos erros com a correção dos pesos e níveis de *bias* (LUDWIG; COSTA, 2007).

- (iii) Propagação: aplica-se à camada de entrada da rede o vetor de sinais de entrada e calcula-se a saída para todos os neurônios até a camada de saída, em seguida se calcula o erro para cada neurônio de saída (LUDWIG; COSTA, 2007).
- (iv) Retropropagação: calcula-se os ajustes dos pesos daquela camada bem como o *bias*, os quais devem ser somados com os valores atuais. O processo segue até se ajustar os valores de pesos e *bias* da camada de entrada (LUDWIG; COSTA, 2007).
- (v) Iteração: apresentam-se novas épocas de exemplos de treinamento para a rede de forma aleatória até que seja satisfeito o critério de parada (LUDWIG; COSTA, 2007).

3 MATERIAIS E MÉTODOS

Neste capítulo são apresentados os materiais necessários para o desenvolvimento deste projeto bem como todos os procedimentos seguidos para levar os resultados apresentados.

3.1 MATERIAIS

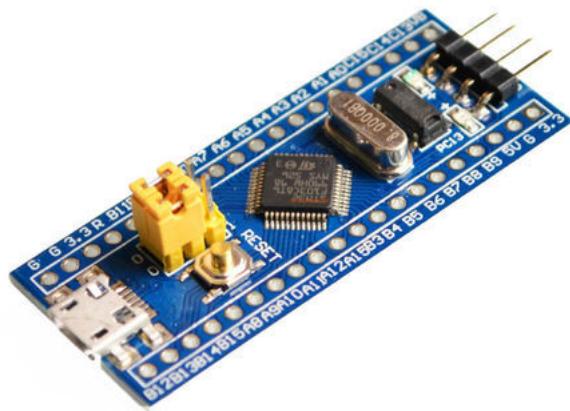
No presente trabalho foram utilizados diversos materiais e ferramentas computacionais, sendo que os principais compreendem:

- Placa de desenvolvimento com microcontrolador STM32F103C8T6;
- Módulo *display* LCD 16x2 HD44780;
- Luva de sensores construída manualmente;
- Sensor inercial MPU-6050;
- Circuito condicionador de sinal;
- IDE de programação para família de microcontroladores STM32(Atollic);
- Depurador/programador ST-Link/V2;
- Interpretador da linguagem Python.

3.1.1 PLACA DE DESENVOLVIMENTO COM MICROCONTROLADOR STM32F103C8T6

A referida placa de desenvolvimento, ilustrada na Figura 22, possui um microcontrolador ARM-Cortex M3 de 32 bits com 20kB de memória SRAM e 64 kB de memória flash. Ainda contém 2 conversores A/D (Analógico para digital) de 12 bits, 3 temporizadores de propósito geral e até 37 portas de entrada e saída tolerantes a 5V (STMICROELECTRONICS, 2015).

Figura 22: Microcontrolador STM32F103C8T6

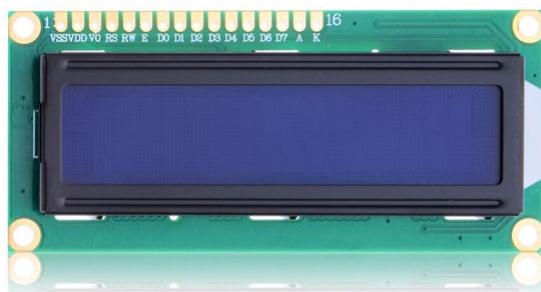


Fonte: Indiamart (2018).

3.1.2 MÓDULO DISPLAY LCD 16X2 HD44780

O módulo HD44780 *display* de matrizes de pontos feito é com cristal líquido que pode reproduzir alfanuméricos, caracteres japoneses e símbolos customizados. Pode ser configurado para operar com transferência de 8 bits ou 4 bits. Todas as operações efetuadas para manipulação do *display* são feitas na memória do controlador interno, sendo fornecido um driver transforma as informações escritas na memória para a tela de cristal líquido. O referido *display* tem compatibilidade de pinagem com outros lcds de 16x2 para ser facilmente trocado caso haja necessidade. A baixa tensão de operação (2.7V até 5.5V) permite que o *display* seja utilizado em sistemas ligados a bateria (HITACHI, 2015). Na Figura 23 está o *display* HD44780.

Figura 23: Display LCD 16x2 de cristal líquido HD44780



Fonte: FilipeFlop (2018).

3.1.3 SENSOR INERCIAL MPU-6050

O dispositivo MPU6050 é um sensor inercial de 6 eixos integrados que combina 3 eixos de giroscópio e 3 eixos de acelerômetro, e conta com um processador digital de movimentos, isso tudo em um encapsulamento de 4x4x0,9 mm. Utilizado com comunicação I2C, o sensor aceita entradas de um compasso externo de 3 eixos para retornar uma fusão de coordenadas de movimento de 9 eixos (INVENSENSE, 2013).

O dispositivo MPU6050 também possui internamente 3 conversores A/D (Analógico para digital) de 16 bits para giroscópio e 3 conversores A/D para acelerômetro afim de converter as entradas analógicas dos sensores. A escala de operação do acelerômetro é programável e pode ser escolhida entre $\pm 2g$, $\pm 4g$, $\pm 8g$, e $\pm 16g$. O mesmo ocorre para o giroscópio, que tem as escalas de operação de ± 250 , ± 500 , ± 1000 , e $\pm 2000 \text{ }^{\circ}/\text{sec}$ (INVENSENSE, 2013). A Figura 24 ilustra a placa com o sensor MPU-6050 utilizada no presente projeto.

Figura 24: Sensor inercial MPU6050

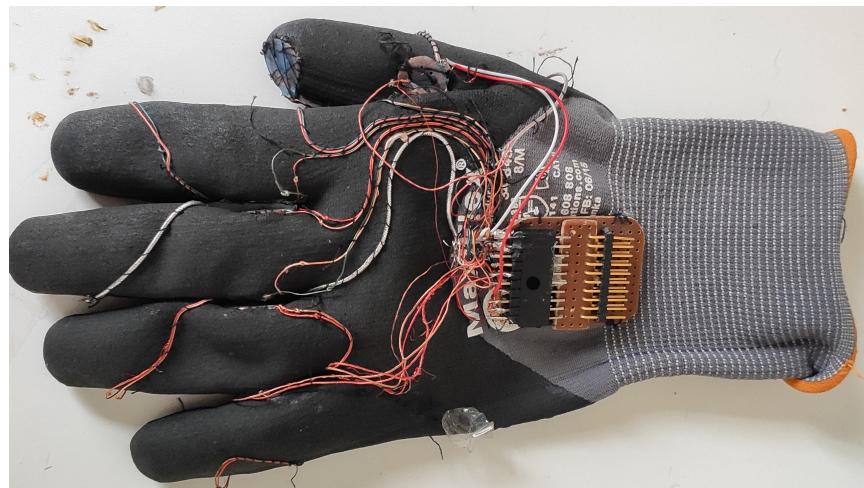


Fonte: FilipeFlop (2018).

3.1.4 CONJUNTO DE LUVA E CIRCUITO DE CONDICIONAMENTO DE SINAIS

A luva sensora, demonstrada na Figura 25, contém as bobinas sensoras e geradoras, todas ligadas ao um conector.

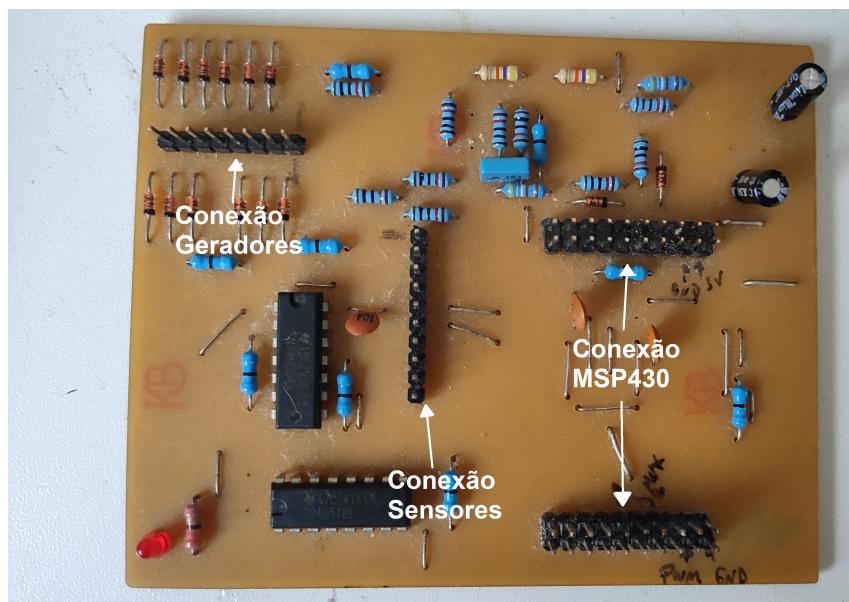
Figura 25: Luva sensora desenvolvida por Lazzarotto (2016)



Fonte: Autoria própria.

A luva é conectada a placa de processamento de sinais das bobinas da Figura 32. O funcionamento desta placa foi demonstrado na sessão 2.4.2.

Figura 26: Placa para recepção e processamento dos sinais vindos do sensor indutivo



Fonte: Autoria própria.

A conexão é para o microcontrolador MSP430F5529, onde será encaixada

uma placa de adaptação para o microcontrolador STM32F103C8T6.

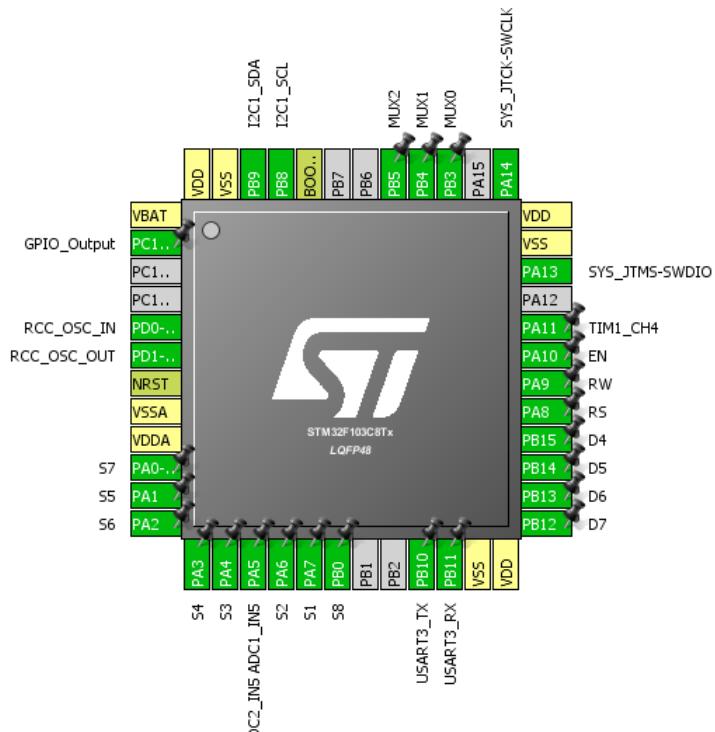
3.2 MÉTODO

A metodologia deste trabalho foi separada em compreensão do funcionamento e transferência do código para o novo microcontrolador, implementação do sensor inercial, processamento dos sinais, treinamento das redes neurais, e implementação das redes já treinadas no microcontrolador.

3.2.1 TRANSFERÊNCIA DO TRABALHO PARA O NOVO MICROCONTROLADOR

Primeiramente, foi necessário a análise do sistema já implementado em um trabalho anterior para conhecer o funcionamento como um todo. Sendo feito a análise da resposta dos sensores indutivos, e fluxo do programa implementado. Após analisado o código, a estrutura física da luva, e funcionamento do sistema de condicionamento de sinais, foi feita a implementação no novo microcontrolador ARM. Foi utilizado o software STM32CubeMX, para realizar as configurações dos periféricos de forma visual. A Figura 27 mostra os pinos configurados no software.

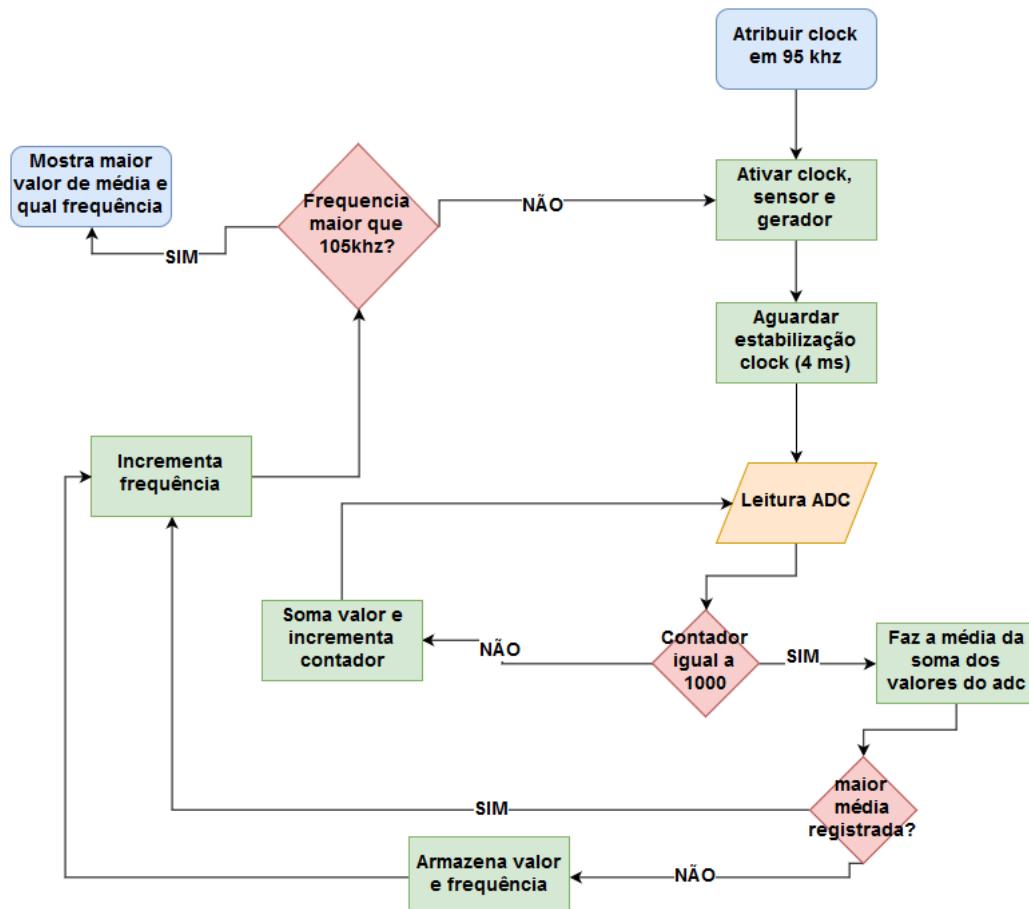
Figura 27: Pinos do microcontrolador configurados no software STM32CubeMX



Fonte: Autoria Própria

Como o microcontrolador proposto para este trabalho possui uma frequência de *clock* maior que aquele utilizado por Lazzarotto (2016), foi necessário descobrir a frequência de *clock* ideal para alimentar as bobinas geradoras. Utilizando um *software* chamado STM32Studio, pelo qual é possível monitorar variáveis em tempo real pelo computador, foi monitorada uma bobina sensora para alcançar o seu valor máximo quando o mais próximo possível da bobina geradora. Para isso, foi feito um pequeno teste com cada par sensor/gerador variando a frequência na bobina geradora de 95 kHz até 105 kHz. Ao obter o valor máximo, este valor de frequência é deixado fixo para esse par.

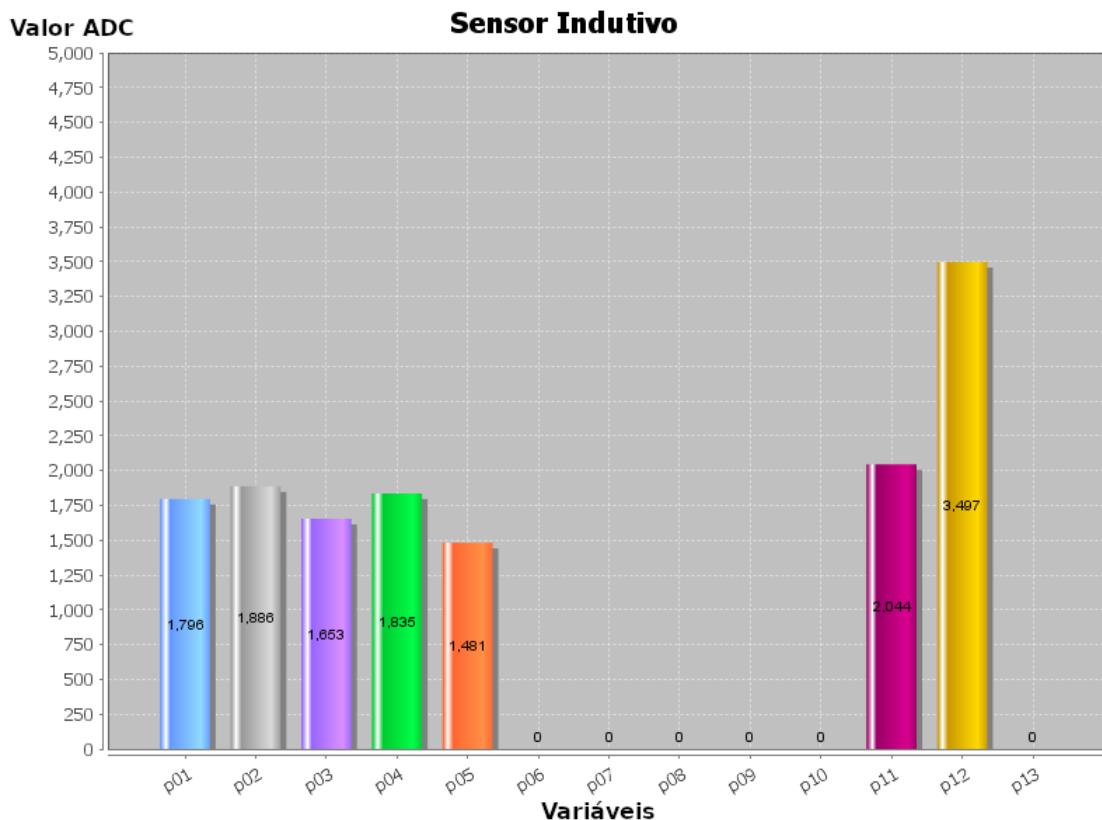
Figura 28: Fluxograma que detalha a rotina desenvolvida para calibrar os sensores indutivos



Fonte: Autoria Própria.

Após configurados os valores para cada par sensor/gerador, foi feito o gesto que representa a letra A em LIBRAS, e o resultado obtido nos sensores indutivos pode ser observado no gráfico da Figura 29.

Figura 29: Valor dos sensores indutivos ao fazer o gesto da letra A em LIBRAS no software STM32Studio



Fonte: Autoria Própria.

A relação de sensores e geradores com as variáveis exibidas na Figura 29 é exibida na Tabela 1.

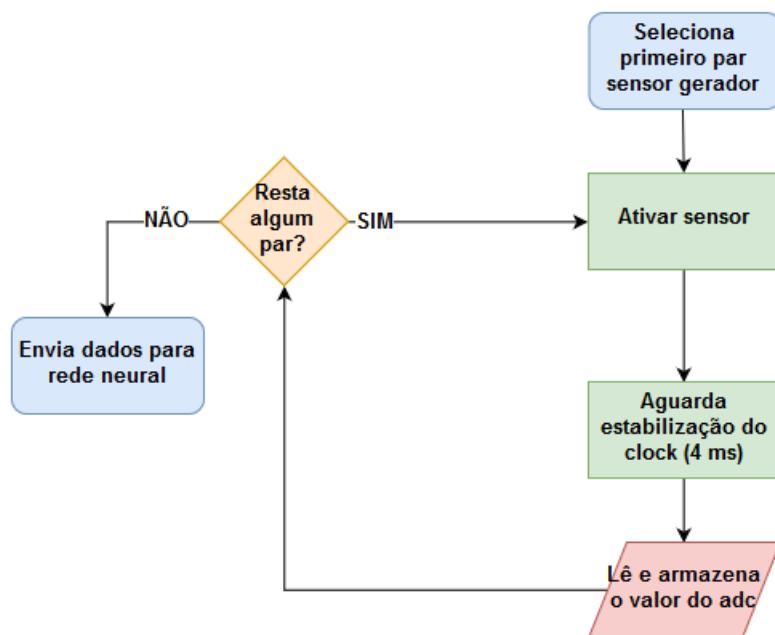
Tabela 1: Relação de sensores e geradores com as variáveis da Figura 29

Variável	p01	p02	p03	p04	p05	p06	p07	p08	p09	p10	p11	p12	p13
Sensor	S1	S2	S3	S4	S5	S5	S1	S2	S3	S6	G4	G5	G6
Gerador	G1	G1	G1	G1	G1	G2	G3	G3	G3	G3	S6	S7	S8

Fonte: Autoria Própria.

Na Figura 1 percebe-se que a medida p13 não está marcando um valor válido. Isso se deve ao fato do sensor/gerador terem dado defeito constantemente. Então foi analisada a necessidade deste sensor e foi decidido não utilizá-lo. A rotina para a coleta dos dados de cada par sensor/gerador tem o funcionamento descrito pelo fluxograma da Figura 30.

Figura 30: Fluxograma que detalha o funcionamento da rotina de coleta de dados do sensor indutivo.



Fonte: Autoria Própria.

A implementação da rotina de coleta de dados é exibida no Código 2.

Após toda a codificação, foi testado o funcionamento com os mesmos valores da rede neural que foi implementada no trabalho anterior desenvolvido por Lazzarotto (2016). Ao retirar um dos sensores induktivos, a entrada da rede neural implementada pelo Lazzarotto (2016) teria uma resposta errada, sendo. Considerando isto e também a diferença de periférico ADC do microcontrolador MSP430, utilizado no trabalho anterior, para o ADC do microcontrolador deste trabalho (STM32F103), foi decidido modificar a rede neural.

Porém, a resposta não estava satisfatória, devido a diferença dos periféricos A/D de um microcontrolador para outro. Então foi decidido refazer a rede neural.

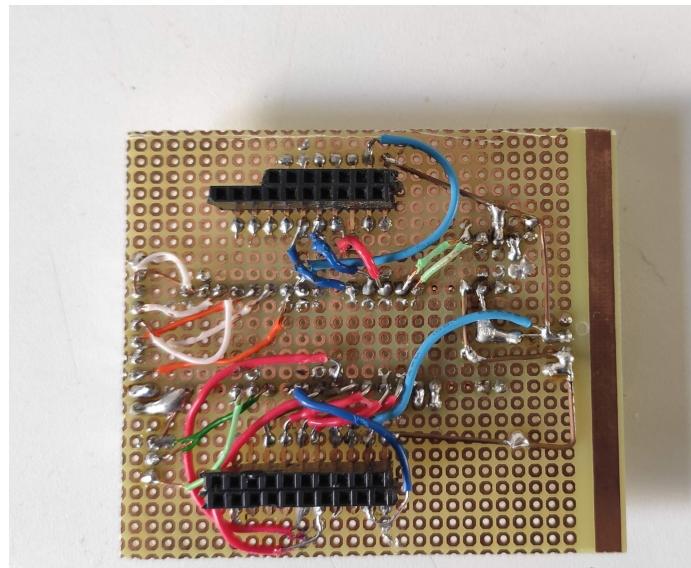
O laço principal do *firmware* é demonstrado no Código 3, onde é feita a coleta dos sensores induktivos, enviado para serial caso solicitado, e classificada a letra.

Para a implementação do *display*, foi feito um código baseado na implementação de (SETIAWAN, 2016).

3.2.2 PLACA DE ADAPTAÇÃO PARA NOVO MICROCONTROLADOR

Para não refazer a placa toda, foi feita uma placa de adaptação para o novo microcontrolador, que possui como única função encaixar na placa de processamento de sinais e direcionar os pinos para os pinos corretos do stm32.

Figura 31: Vista das ligações da placa de adaptação para o microcontrolador stm32f103c8t6

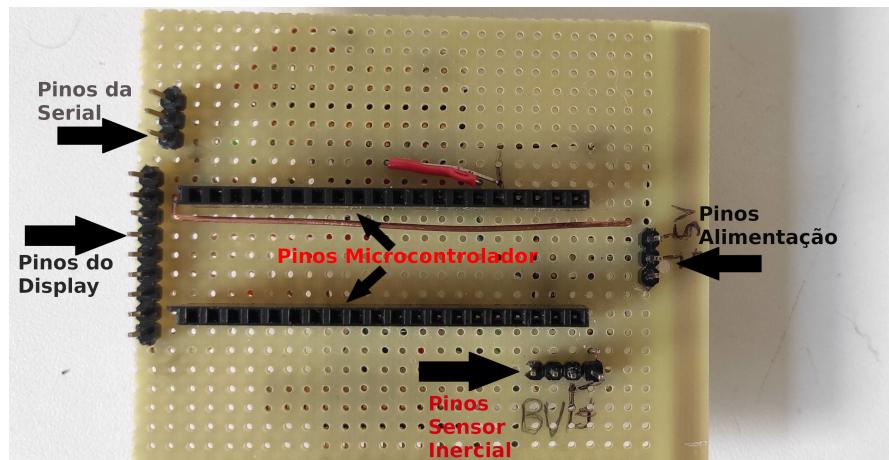


Fonte: Autoria Própria.

Como demonstrado nas Figura 31, a placa foi feita a partir de uma placa ilhada de contatos pela facilidade de fazer as ligações sem precisar de muito planejamento e software para produzi-la.

O posicionamento dos componentes da placa pode ser visto na Figura 32. A placa conta com barras de pinos para ligar o *display*, barra de pinos para conectar a interface serial e comunicar com o computador, barra de pinos para conectar o sensor inercial MPU6050, e também a barra de pinos para a alimentação das placas.

Figura 32: Vista de cima da placa de adaptação para o microcontrolador stm32f103c8t6



Fonte: Autoria Própria.

3.2.3 CONFIGURAÇÃO SENSOR MPU6050 E PROCESSAMENTO DE SINAIS

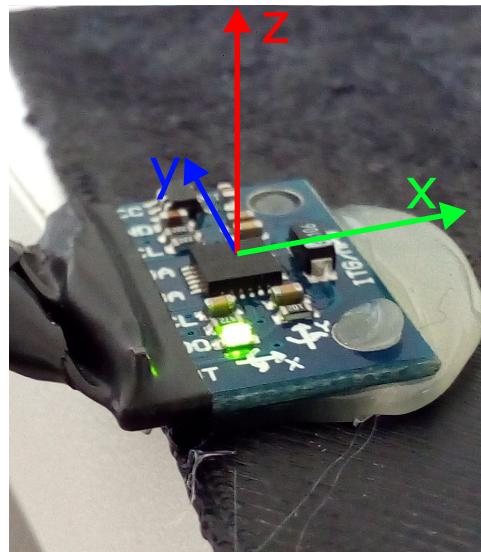
Esta sessão detalha a configuração do sensor MPU6050 e explica como foi feita o processamento dos sinais vindos do sensor.

3.2.3.1 CONFIGURAÇÃO E CALIBRAÇÃO DO SENSOR MPU-6050

O sensor inercial, utilizando comunicação de circuitos inter-integrados ou I²C, sendo necessário o uso e configuração de uma interface serial neste padrão no microcontrolador.

A biblioteca de abstração de hardware (HAL) disponibilizada pela fabricante do microcontrolador facilita a configuração de periféricos, necessitando apenas de poucas funções para configurar. Utilizando o *datasheet* do sensor MPU6050, foi identificada a sequência de passos para iniciar, calibrar e ler os valores. Para calibrar o acelerômetro, são estimados os valores de *offsets* necessários para o funcionamento do sensor. Esses valores variam para cada placa fabricada. Então foi desenvolvida uma rotina de calibração para o sensor. O sensor deve ser posicionado em uma superfície plana com o eixo Z em 90 graus com a superfície, como é exibido na Figura 33.

Figura 33: Posicionamento da MPU6050 e seus eixos para a calibração



Fonte: Autoria Própria.

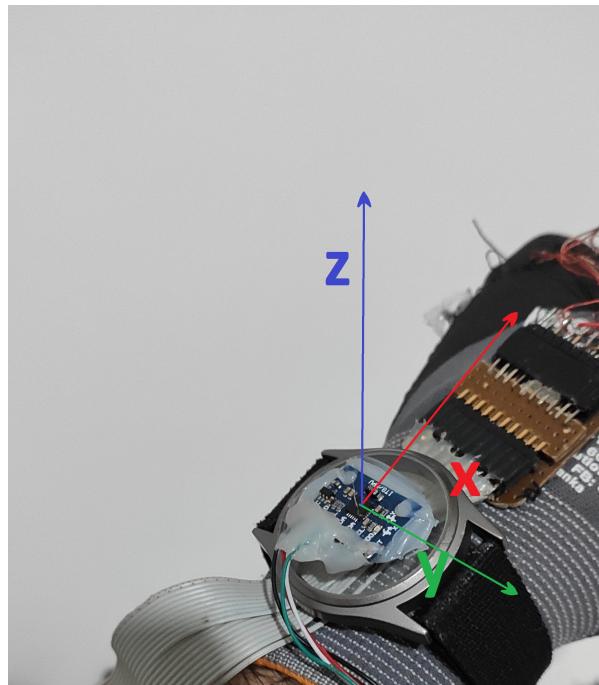
Para este trabalho, foi utilizado somente o acelerômetro para a detecção dos movimentos.

3.2.3.2 LETRAS CONFLITANTES E PROCESSAMENTO DOS SINAIS

Observando todas as letras do alfabeto de LIBRAS (Figura 1), nota-se que algumas letras tem o mesmo posicionamento de mãos, sendo apenas diferenciadas pelo movimento. Essas são: J e I; P, K e H; e X e Z; No trabalho anterior desenvolvido por Lazzarotto (2016), foram deixadas de fora as letras com movimento (X, K, H, e J). Para reconhecer todas as letras, foi utilizado o acelerômetro para detecção de movimento. Ao incluir detecção de movimento, as letras que contém a mesma posição de mão, como as letras J e I, tornam-se conflitantes, mesmo que a letra I não contenha movimento, porque a posição de mão dela é igual ao da letra J. O mesmo ocorre com as letras P, K, e H, onde a letra P não contém movimento e as letras H e K usam movimento.

O trabalho do acelerômetro do sensor inercial é detectar esse movimento ocorrido. Na Figura 35, são exibidas apenas as letras com movimentação. Posicionando o eixos de coordenadas do sensor inercial na luva como demonstrado na Figura 34.

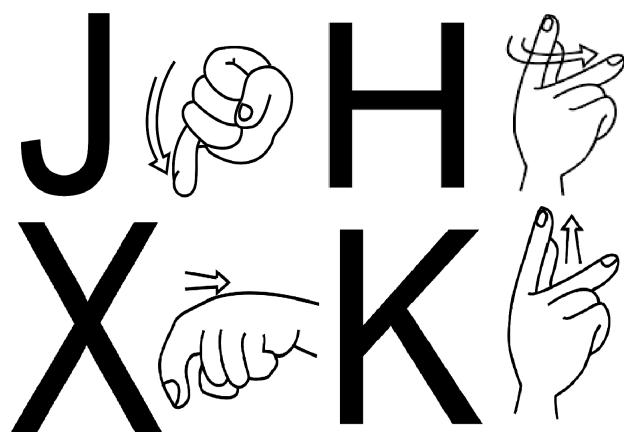
Figura 34: Posicionamento do sensor inercial na luva com eixo de coordenadas.



Fonte: Autoria Própria.

Para detectar os movimentos das letras exibidas na Figura 35, foram utilizadas as 3 coordenadas do acelerômetro (x , y , e z), e a coordenada de rotação Roll.

Figura 35: Letras do alfabeto de libras que contém movimento.

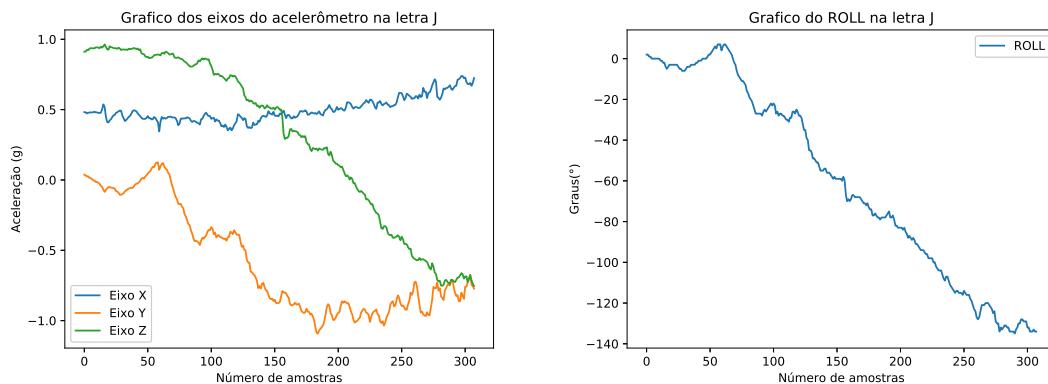


Fonte: Autoria Própria.

Os sinais provenientes da luva são os doze pares sensor/gerador. O sinal lido das bobinas pelo ADC já está processado, somente necessitando normalizar. E também os sinais vindos do acelerômetro.

Para captar o movimento, foi definido uma janela de tempo de 500 ms, que é o tempo necessário para fazer o movimento de uma letra. Na Figura 36 pode ser visto os sinais do acelerômetro no período de 500 ms.

Figura 36: Gráficos dos sinais do acelerômetro, amostrando 500 ms.



Fonte: Autoria Própria.

Essa janela de tempo é dividida em 10 pedaços, e a cada 50 ms, são coletados os dados dos eixos do sensor inercial, e sempre a amostra mais velha dos dados é descartada, fazendo com que a janela de 500 ms deslize no tempo. Um exemplo de como será a janela pode ser visto na Figura 37.

Figura 37: Janela de amostras



Fonte: Autoria Própria.

3.2.4 REDES NEURAIS IMPLEMENTADAS

Para manter o foco na solução sugerida por este trabalho, foram separadas as letras em dois grupos: letras sem conflito, e as letras conflitantes. As letras sem conflito são aquelas que somente o sensor indutivo consegue classificar, sem a necessidade do sensor inercial. E as letras conflitantes são as quais tem a necessidade de utilizar o sensor inercial como diferencial entre elas. As letras consideradas conflitantes são P, H, K, I, J, Z, X. Considerando esses dois grupos, foram feitas duas redes neurais. A posição de mão das letras conflitantes entram na classificação das não conflitantes, mas só é exibido um resultado quando é executada a segunda classificação com os dados do sensor inercial.

3.2.4.1 REDE NEURAL PARA OS SENSORES INDUTIVOS

Como já mencionado na subseção 3.2.1, a rede neural que trata dos sinais provenientes dos sensores indutivos (pares de bobinas sensora/geradora) foi refeita, ao retirar um dos sensores indutivos, uma das entradas sempre será zero afetando o desempenho do sistema.

As características são bem definidas como sendo cada par de bobinas sensora/geradora da luva. Foi criada um Perceptron Multi Camada (MLP) com 21 neurônios de camada oculta, utilizando a função de ativação tangente hiperbólica (Sessão 2.8.2).

O treinamento foi feito com um conjunto de 50 amostras de cada letra, onde 75% para treinamento e 25% para teste da rede. Esse processo de divisão dos dados é feito com uma mistura na ordem dos dados. Para permanecer parecida com a rede implementada no trabalho anterior, foi utilizada a mesma quantia de neurônios de camada oculta (21 neurônios).

O código 4 foi elaborado para o treinamento, obtendo uma taxa de acerto de 96,36% ao final do treinamento e teste.

Os dados dos sensores de cada letra feita foi armazenada em arquivos, e a função `arruma_dados` é responsável por juntar esses dados em uma matriz de características. Após as características são redimensionadas para ficar no intervalo da função de ativação de $[-1, 1]$. Após isso é feita a divisão dos dados em vetores de teste e treino, (`x_train` e `x_test`). Então é feito o treino chamando a função `fit` do *MLP-Classifier*. O resultado da classificação é visualizado com a função `predict`, na qual usa o vetores de teste para comparar a classificação feita pela rede e a resposta real.

Para efetivamente fazer a classificação, é implementado o método *feedforward*, utilizando os pesos obtidos no treinamento. Resultando no Código 6.

Após a classificação, para transformar o número em uma letra é utilizada uma função de pós-processamento descrita no Código 7.

Nas letras de retorno pode-se notar os números 1, 2, e 3. Essas são as classes de pose de mão que são das letras conflitantes, esse é a ligação com a outra rede neural implementada, ou seja, não são exibidos esses números no *display*.

3.2.4.2 REDE NEURAL SENSOR INERCIAL

Considerando os dados já processados, foram colhidos pela serial e inseridos no código em Python para treinamento da rede. As características são elas que são

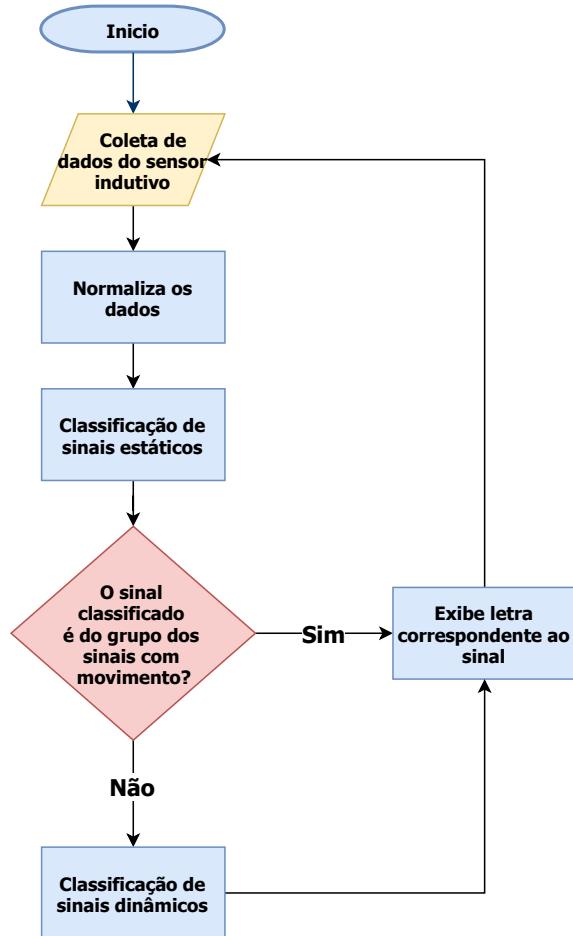
10 pedaços de tempo em que cada pedaço tem 4 valores. Foram colhidos 25 amostras de cada movimento, incluindo uma posição sem movimento. No Código 8, está demonstrado o treinamento da rede responsável pelo acelerômetro. Foi utilizado a função de ativação tangente hiperbólica. Utilizando 80 % dos dados para treino e 20 % para teste separados aleatoriamente.

A função *LoadFromCsv* carrega os amostras dos arquivos e reúne em um vetor de características. Então esses dados são redimensionados para se adequar a função de ativação que é o intervalo de $[-1, 1]$. Então são separados os conjuntos de treino e teste para executar a função *fit* que treina a rede neural. A função *predict* compara os valores reais de classificação com os valores classificados e retorna o valor de acerto da rede treinada.

A função de *feedforward* implementada é a mesma do Código 6, mudando apenas o pesos. O pós-processamento também só alterando o vetor de classes de saída que são as do Código 10.

A função que faz a ligação das duas redes neurais é descrita no Código 11, onde são os dados do sensor indutivo são tratados e classificados. Caso a classificação seja uma das classes das letras conflitantes (1, 2, ou 3), então é acionado o acelerômetro para detectar o movimento e depois ele é classificado. No diagrama da Figura 38 está o detalhado o *firmware* desenvolvido.

Figura 38: Fluxograma que descreve o comportamento do *firmware* a ser desenvolvido.



Fonte: Autoria Própria.

4 RESULTADOS

Neste capítulo são demonstrados os resultados obtidos após o desenvolvimento do projeto.

4.1 SISTEMA COMPLETO

O sistema completado é demonstrado na Figura 39.



Figura 39: Conjunto ao final do trabalho.

A Figura 40 apresenta uma demonstração da luva formando a letra C e também o *display* mostrando a classificação correta da letra.

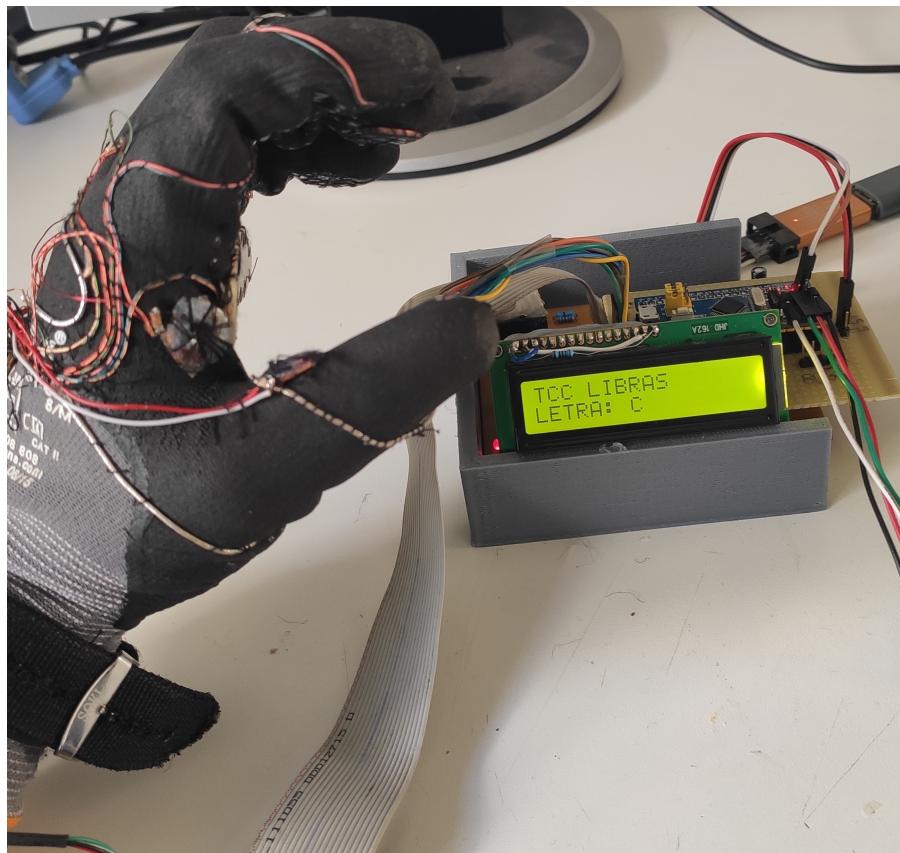


Figura 40: Demonstração da letra C.

Esse grupo de letras já era classificado no trabalho anterior, mas como a rede foi refeita, ela deve ser validada. Na Tabela 2, está a matriz de confusão dessa rede refeita, onde foram testada cada letra dez vezes em sequência aleatória.

		Classe Predita																			
		A	B	C	D	E	F	G	L	M	N	O	Q	R	S	T	U	V	W	Y	
C l a s s e r a l	A	15																			
	B		13	2																	
	C			15																	
	D			2	13																
	E					12												3			
	F						15														
	G							14	1												
	L								13								2				
	M		2							13											
	N										15										
	O											15									
	Q												13			2					
	R													15							
	S														15						
	T															15					
	U													3			12				
	V																	1	14		
	W																	2	13		
	Y	1																			14

Tabela 2: Matriz de confusão das letras sem conflito

Contanto todos os erros obtidos da Tabela 2, temos um total de 21 vezes que uma classe foi classificadas erroneamente. Obtendo um total de 92,63% de acerto, muito próximo dos 96,36% de resultado do treinamento.

Agora para as letras conflitantes, como já explicado, a rede neural feita para classificar os movimentos obteve os resultados demonstrados pela Tabela 3.

		Classe Predita						
		I	J	K	H	P	Z	X
R e a l	I	15						
	J	3	10	2				
	K			12		3		
	H				13	2		
	P					15		
	Z						15	
	X						6	9

Tabela 3: Matriz de confusão das letras conflitantes

Considerando a matriz das letras conflitantes da Tabela 3, temos 88,7% de acerto, um percentual bom comparado aos 100% aos do resultado do treinamento no código em Python.

Reunindo o resultado das duas redes, tem-se um percentual de 88,7% de acerto. Esses 11,3 % de erro são justificados por vários fatores. Um deles são os pontos cegos dos sensores indutivos como Lazzarotto (2016) descreve nos em seus resultados que podem ser resolvidos modificando a posição dos sensores e adicionando mais sensores indutivos. Também se deve ao fato de se estar utilizando somente o acelerômetro na classificação, onde poderia obter maior precisão utilizando giroscópio. A abordagem de janela deslizante funcionou bem, mas para uma classificação mais efetiva seria necessário uma abordagem de um algoritmo de aprendizagem de máquina que considera o tempo para aumentar essa taxa de acerto.

5 CONCLUSÕES

O presente trabalho resultou em aprimoramento em um sistema de reconhecimento de letras do alfabeto de LIBRAS, permitindo detectar letras que contém movimento e completando as letras identificáveis pelo sistema. O código foi portado, obviamente com alterações para funcionar corretamente e a coleta dos dados do sensor inercial e as redes neurais funcionaram. A única coisa que não foi possível cumprir foi o teste com outros usuários para validar o funcionamento do sistema.

A luva, a qual já havia sido construída em um trabalho anterior, possui fios e sensores expostos e frágeis. Então o manuseio da luva teve que ser bem delicado pois os fios das bobinas sempre soltavam, necessitando reparos. Uma solução pra isso em um trabalho futuro seria trocar a luva e esconder a fiação. Também escolher uma luva mais flexível, pois em alguns sinais é necessário fazer um pouco de força para a sinalização correta da letra, o que também pode vir a danificar os sensores. A comunicação I2C utilizada com o sensor inercial se mostrou pouco robusta por causa do tamanho do cabo (inicialmente 60 cm), que teve que ser reduzido para não causar tantos problemas de comunicação.

Com relação ao microcontrolador, este foi suficiente para a demanda de processamento. Apesar disso, devido a abordagem de *software* com *background/foreground*, foi necessário temporizar e diminuir a frequência que se capturava os sinais. Como a aplicação é sensível ao tempo, seria interessante aplicar um *protothread* ou um sistema operacional de tempo real para eliminar algumas esperas ocupadas.

Para trabalhos futuros, como identificar sinais de LIBRAS simbolizam palavras e utilizam apenas uma mão e sem expressões faciais, seria necessário uma análise sobre o sensor inercial, trocando algumas posições de sensores para as características da rede neural ficarem mais marcantes. Para o processamento dos dados do sensor inercial poderia ser estudado e implementado o código que o microcontrolador do sensor já realiza os cálculos dos ângulos de rotação(*Pitch*, *Yaw*) e se aplicar um sensor inercial de 9 ou 10 graus de liberdade para melhorar a precisão dos sinais lidos do sensor inercial.

Caso seja a intenção de implementar sinais com duas mãos, seria necessário outra luva, mas a demanda de tempo para a leitura das duas poderia ser elevada. Por

isso deve-se utilizar a abordagem da luva com mais tipos de sensores, por exemplo alguns sensores resistivos de flexão, que são lidos mais rapidamente e simples de interpretar, juntamente com sensores inerciais de 9 ou 10 graus de liberdade.

E por fim, para implementar um sistema completo para identificar sinais e expressões faciais, seria necessário implementar processamento de imagem.

(12)asdadas

APÊNDICE A - CÓDIGOS DESENVOLVIDOS

Os códigos desenvolvidos neste trabalho estão disponíveis em Góes (2019).

O Código 1 inicia os periféricos utilizados neste trabalho, também inicia o sensor inercial e o *display*. Ao iniciar, são necessários 10 segundos de espera para o funcionamento correto, essa espera é exibido no *display*. Após a espera, é iniciado o sensor inercial e o código entra no laço principal.

Código 1: Rotina de inicialização do projeto

```

1 void StartProjeto(void) {
2     ind_set_tim(&htim1);
3     ind_set_adc(&hadc1);
4     i2c_init(&hi2c1);
5     tim4_init(&htim4);
6     uart_init(&huart3);
7     uart_receive_it(&uart_d, 1);
8     lcd16x2_init(LCD16X2_DISPLAY_ON_CURSOR_OFF_BLINK_OFF);
9     lcd16x2_puts("Iniciando Sensor MPU");
10    for(uint8_t i = 0; i < 10; i++) {
11        HAL_Delay(1000);
12        lcd16x2_clrscr();
13        lcd16x2_puts("Iniciando em ");
14        lcd16x2_gotoxy(0, 1);
15        lcd16x2_putc(i + 49);
16    }
17    lcd16x2_putc(result);
18    if (!i2c_device_ready(MPU_ADDR)) {
19        asm("NOP");
20        while(1);
21    }
22    mpu_init(&mpu);
23    mpu.acc_res = ACC_SENS_2;
24    mpu_set_offset_acc_x(ACC_OFF_X);
25    mpu_set_offset_acc_y(ACC_OFF_Y);
26    mpu_set_offset_acc_z(ACC_OFF_Z);
27    lcd16x2_clrscr();
28    lcd16x2_gotoxy(0, 0);
29    lcd16x2_puts("TCC LIBRAS\nLETRA: -");
30}
31

```

Fonte: Autoria própria.

O Código 2 faz a coleta dos sensores indutivos, selecionando um dos 12 pares de sensor/gerador e ativando o *clock*. Após 4 ms é feita a leitura do conversor A/D (Analógico para Digital). O valor desta leitura é armazenado em um vetor.

Código 2: Rotina de coleta de dados do sensor indutivo

```

1 void feedTheInductiveSensor( void ) {
2     for( uint8_t i = 0; i < 12; i++ ) {
3         ind_set_ger( ger[ i ] );
4         ind_set_sen( sen[ i ] );
5         ind_set_pwm( pwm[ i ] );
6         HAL_Delay( 5 );
7         ind_start_conversion();
8         while( !ind_get_conv_finish() );
9         valores[ i ] = ind_get_conv_value();
10    }
11 }
```

Fonte: Autoria própria.

No Código 3, é demonstrado o laço principal do programa. Antes de entrar no laço, são iniciados os periféricos e variáveis na função *StartProjeto*, e repete-se a captura dos dados do sensor indutivo e realiza-se a classificação. Uma interface serial foi configurada para a transferência dos dados do microcontrolador para o computador para treinamento de rede neural. E quando um dado chega na interface serial, a função *sendToSerial* é chamada para mandar os dados dos sensores.

Código 3: Laço principal do *firmware*

```

1 int main( void )
2 {
3     StartProjeto();
4     while( 1 )
5     {
6         feedTheInductiveSensor();
7         if( xSemSerial ) {
8             sendToSerial();
9         }
10        classifyLetter();
11    }
12 }
```

Fonte: Autoria própria.

O Código 4 é o treinamento da rede neural para os sensores indutivos em Python, onde é utilizado 75% dos dados para treinamento e 25% dos dados para teste.

Código 4: Treinamento da rede neural dos sensores indutivos em Python

```

1 def treinamento(activation , hidden_layers , verbose=False):
2     caracteristicas = arruma_dados()
3     scaler = pre.MinMaxScaler(feature_range=(-1,1))
4     scaler.fit(caracteristicas)
5     caracteristicas = scaler.transform(caracteristicas)
6
7     x_train , x_test , y_train , y_test = train_test_split(caracteristicas ,
8         classes , test_size=0.25)
9     clf = MLPClassifier(hidden_layer_sizes=(hidden_layers ,) , activation=
10        activation , max_iter=1500, shuffle=True , n_iter_no_change=100)
11
12     clf.fit(x_train , y_train)
13     y_pred = clf.predict(x_test)
14
15
16     return clf

```

Fonte: Autoria própria.

Para a implementação do código da rede treinando no microcontrolador, o Código 5 exibe a quantia de entradas, o número de camadas ocultas e o número de saídas da rede. Também são exibidos todos os pesos e os *bias* da rede treinada.

Código 5: Definições da rede neural dos sensores indutivos

```

1 #define NUMBER_OF_INPUTS           12
2 #define NUMBER_OF_HIDDEN          21
3 #define NUMBER_OF_OUTPUTS         22
4 float weightInputHidden[NUMBER_OF_INPUTS][NUMBER_OF_HIDDEN] { ... };
5 float weightHiddenOutput[NUMBER_OF_HIDDEN][NUMBER_OF_OUTPUTS] = { ... };
6 float biasHidden[NUMBER_OF_HIDDEN] = { ... };
7 float biasOutput[NUMBER_OF_OUTPUTS] = { ... };
8

```

Fonte: Autoria própria.

No Código 6, é apresentada a rotina de *feedforward* dos sensores indutivos, os valores dos sensores são a entrada, e baseado nos pesos treinados da rede neural, é dado um resultado da classificação.

Código 6: Feedforward implementado no microcontrolador

```
1 void classSystemPredict( float * entrada, float * saida) {
2     float oculta[NUMBER_OF_HIDDEN];
3     float sum = 0;
4
5     for( int i = 0; i < NUMBER_OF_HIDDEN; i++) {
6         sum = 0;
7         for( int ii = 0; ii < NUMBER_OF_INPUTS; ii++) {
8             sum += entrada[ ii ] * weightInputHidden[ ii ][ i ];
9         }
10        oculta[ i ] = tanhf( sum + biasHidden[ i ] );
11    }
12
13    for( int i = 0; i < NUMBER_OF_OUTPUTS; i++) {
14        float sum = 0;
15        for( int ii = 0; ii < NUMBER_OF_HIDDEN; ii++) {
16            sum += oculta[ ii ] * weightHiddenOutput[ ii ][ i ];
17        }
18        saida[ i ] = tanhf( sum + biasOutput[ i ] );
19    }
20 }
```

Fonte: Autoria própria.

O pós-processamento da rede neural dos sensores indutivos é feito verificando qual foi a saída com o maior valor, variando de 0 a 0,99. A classe com o maior valor é a saída da rede.

Código 7: Pós processamento da rede neural dos sensores indutivos

```

1 char classSystemPostProcess( float * saidas) {
2     float max = 0;
3     int ind = 0;
4     for (int i = 0; i < NUMBER_OF_OUTPUTS; i++) {
5         if(max < saidas[i]) {
6             max = saidas[i];
7             ind = i;
8         }
9     }
10    char ret[] = { 'A', 'B', 'C', 'D', 'E', 'F', 'G', '1', '2', 'L', 'M', 'N',
11        'O', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'Y', '3' };
12    return ret[ind];
13 }
```

Fonte: Autoria própria.

O Código 8 é o treinamento da rede neural para o sensor inercial em Python, onde é utilizado 80% dos dados para treinamento e 20% dos dados para teste.

Código 8: Treinamento da rede neural do sensor inercial Python

```

1 def TrainNeuralNet(data , classes , numberOfHidden , activationFunction ,
2     verbose=False):
3
4     xData , yData = LoadDataFromCsv()
5     scalledData , dataMin , dataMax = ScaleData(xData)
6     xTrain , xTest , yTrain , yTest = train_test_split(data , classes , test_size
7         =0.2)
8     MLP = MLPClassifier(hidden_layer_sizes=(numberOfHidden,) , activation=
9         activationFunction , max_iter=10000, shuffle=True , n_iter_no_change=1000)
10    MLP.fit(xTrain , yTrain)
11
12    yPredict = MLP.predict(xTest)
```

Fonte: Autoria própria.

Para a implementação da rede neural para o sensor inercial também são feitas as definições de quantidade de entrada, saída e número de camadas ocultas. O

Código 9 demonstra essas definições e também os vetores com os pesos da rede já treinada.

Código 9: Definições da rede neural dos sensores indutivos

```

1 #define NUMBER_OF_INPUTS_A           40
2 #define NUMBER_OF_HIDDEN_A          40
3 #define NUMBER_OF_OUTPUTS_A         5
4
5 float weightInputHiddenA [NUMBER_OF_INPUTS_A][NUMBER_OF_HIDDEN_A] { ... };
6 float weightHiddenOutputA [NUMBER_OF_HIDDEN_A][NUMBER_OF_OUTPUTS_A] = { ... };
7 float biasHiddenA [NUMBER_OF_HIDDEN_A] = { ... };
8 float biasOutputA [NUMBER_OF_OUTPUTS_A] = { ... };
9

```

Fonte: Autoria própria.

Código 10: Vetor de respostas possíveis do pós processamento da rede neural do sensor inercial

```

1 char ret [] = { '-' , 'K' , 'J' , 'H' , 'X' };

```

Fonte: Autoria própria.

O Código 11, exibe a rotina de classificação. Após fazer a leitura dos sensores inerciais, é feita a classificação da letra, caso ela se encaixe nas classes 1, 2, ou 3, é acionada a captura dos dados do sensor inercial e feita a classificação do movimento. Finalmente o resultado é exibido no *display*.

Código 11: Função que executa a classificação das letras

```
1 void classifyLetter(void) {
2     lcd16x2_gotoxy(7, 1);
3     float input[NUMBER_OF_INPUTS];
4     float output[NUMBER_OF_OUTPUTS];
5     for(uint8_t i = 0; i < NUMBER_OF_INPUTS; i++)
6         input[i] = (float)valores[i];
7
8     classSystemScaleDataInput(input);
9     classSystemPredict(input, output);
10    result = classSystemPostProcess(output);
11    HAL_Delay(50);
12    if(result == '1' || result == '2' || result == '3') {
13        getMpuValues();
14        classifyMoviment(result);
15    } else {
16        lcd16x2_gotoxy(7, 1);
17        lcd16x2_putc(result);
18    }
19 }
```

Fonte: Autoria própria.

REFERÊNCIAS

- 5DT. *Data Gloves for the fifth dimension User's Manual*. 2018. Disponível em: <<http://www.5dt.com/downloads/dataglove/old/5DTDataGlove16Manual.pdf>>.
- BAKHSI, S.; MAHOR, M. H.; DAVIDSON, B. S. Development of a body joint angle measurement system using imu sensors. In: **2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society**. [S.I.: s.n.], 2011. p. 6923–6926. ISSN 1094-687X.
- BALBINOT, Alexandre; VALNER, João Brusamarello. **Instrumentação e Fundamentos de Medidas**. 2. ed. Rio de Janeiro, RJ: Editora LTC, 2007. 13-272 p. ISBN 9788521615637.
- BOYLESTAD, Robert L.; NASHESKY, Louis. **Dispositivos eletrônicos e teoria**. 11. ed. São Paulo, SP: Pearson Education do Brasil Ltda, 2013. 13-272 p. ISBN 9788564574212.
- CASTRO, A.; CARVALHO, I. **Comunicação por língua brasileira de sinais: livro básico**. 3. ed. Brasília, DF: Senac/DF, 2009. 23 p. ISBN 9788598694115.
- CATHEY, Jimmie J. **Dispositivos e circuitos eletrônicos**. 1. ed. São Paulo, SP: Makron Books, 1994.
- COSTA MÁRCIA MACHADO MARINHO, Gabrielle Silva Marinho Conceição de Maria Machado. M-learning no ensino de libras: Avaliação de objetos de aprendizagem. **Revista Expressão Católica**, v. 6, n. 1, p. 28–35, jul 2017. ISSN: 2357-8483.
- CYBERGLOVE. **CyberGlove+ III DataSheet**. 2018. Disponível em: <<https://bit.ly/2OeR9rm>>.
- DAS, A.; YADAV, L.; SINGHAL, M.; SACHAN, R.; GOYAL, H.; TAPARIA, K.; GULATI, R.; SINGH, A.; TRIVEDI, G. Smart glove for sign language communications. In: **2016 International Conference on Accessibility to Digital World (ICADW)**. [S.I.: s.n.], 2016. p. 27–31.
- DUARTE, Francisco João Aires. **Classificação de atividades físicas através do uso do acelerômetro do Smartphone**. 19-20 p. Monografia (Dissertação de Mestrado) — Instituto Superior de Engenharia de Lisboa, 2013.
- DUDA, Richard O.; HART, Peter E.; STORK, David G. **Pattern Classification**. 2. ed. New York, NY: Willey, 2001. ISBN 9780471056690.
- ELECTRONICWINGS. **Sensors Modules Mpu6050 Gyroscope Accelerometer Temperature Senso...** 2019. Disponível em: <https://www.electronicwings.com/public/images/user_images/images/Sensor%20%26%20Modules/MPU6050-2_Orientation_Polarity_of_Rotation_MPUSensor.png>.

FAHN, Chin-Shyurng; SUN, Herman. Development of a data glove with reducing sensors based on magnetic induction. **IEEE Transactions on Industrial Electronics**, v. 52, n. 2, p. 585–594, April 2005. ISSN 0278-0046.

FERREIRA, Lucinda. **Por uma gramática de línguas de sinais**. 2. ed. Rio de Janeiro, RJ: Tempo Brasileiro, 2010. 269 p. ISBN 8528200698.

FILIPEFLOP. **FILIPEFLOP Componentes Eletronicos**. 2018. Disponível em: <<https://www.filipeflop.com>>.

FREESCALE SEMICONDUCTOR. **Tilt Sensing Using a Three-Axis Accelerometer**. [S.I.], 3 2013. Rev. 6.

GÓES, Jefferson Willian França. **jeffersonwgoes/TCC-Luva-Sensora: TCC**. 2019. Disponível em: <<https://github.com/jeffersonwgoes/TCC-Luva-Sensora>>.

GESSER, Audrei. **Libras? que língua é essa? : crenças e preconceitos em torno da língua de sinais e da realidade surda**. 1. ed. São Paulo, SP: Parábola, 2009. 33 p. (Série estratégias de ensino ; 14). ISBN 9788579340017.

GLENN RESEARCH CENTER. **Aircraft Rotations**. [S.I.], 5 2015.

HAYKIN, Simon. **Redes Neurais: princípios e prática**. 2. ed. Porto Alegre, RS: Bookman, 2001. 899 p. ISBN 9788573077186.

HAYKIN, Simon. **Neural networks and learning machines**. 3. ed. São Paulo, SP: Pearson Education, 2008. ISBN 9780131471399.

HITACHI. **HD44780U (LCD-II), (Dot Matrix Liquid Crystal Display Controller/Driver)**. [S.I.], 7 2015. Rev. 0.0.

INDIAMART. **ARM STM32 Development Board**. 2018. Disponível em: <<https://www.indiamart.com/proddetail/arm-stm32-development-board-14510513355.html>>.

INVENSENSE. **MPU-6000 and MPU-6050 Product Specification Revision 3.4**. [S.I.], 8 2013. Rev. 3.4.

LAZZAROTTO, Ruani. **Sistema de Reconhecimento de Padrões do Alfabeto da Língua Brasileira de Sinais utilizando Microcontrolador**. 103 p. Monografia (Trabalho Conclusão de Curso 2) — Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná, 2016.

LUDWIG, O.; COSTA, Edward Montgomery. **Redes Neurais: Fundamentos e Aplicações com Programas em C**. 1. ed. Rio de Janeiro, RJ: Editora ciéncia moderna, 2007. ISBN 9788573936193.

MALVINO, Albert; BATES, David J. **Dispositivos e circuitos eletrônicos**. 7. ed. São Paulo, SP: McGraw-Hill, 2007. ISBN 9788577260232.

MARTINS, Heloísa Andreia de Matos Lins Lívia Maria Ninci. Tecnologia e educação de surdos: Possibilidades de intervenção. **Revista Eletrônica do Programa de Pós-Graduação em Educação e do Departamento de Educação da Faculdade de Ciências e Tecnologia/Unesp - Presidente Prudente**., v. 26, n. 2, p. 188–206, may 2015.

NUSSENZVEIG, Moysés H. **Curso de Física Básica: Eletromagnetismo**. 3. ed. São Paulo, SP: Butcher, 1996. ISBN 8521200463.

SCHIMIGUEL, J.; FERNANDES, R. F.; FRANÇA, L DOS SANTOS. Desenvolvimento de objetos de aprendizagem na forma de jogos para ensino de libras. **Sintec-IV Simpósio Nacional de Ensino de Ciência e Tecnologia**, nov 2014.

SETIAWAN, Yohanes Erwin. **stm32f103-keil/lcd16x2 at master . yohanes-erwin/stm32f103-keil**. 2016. Disponível em: <<https://github.com/yohanes-erwin/stm32f103-keil/tree/master/lcd16x2>>.

SILVA, Ivan Nunes da; SPATTI, Danilo Hernane; FLAUZINO, Andrade. **Redes Neurais artificiais: para engenharias e ciências aplicadas**. 1. ed. São Paulo, SP: Artliber editora, 2010. ISBN 9788588098534.

STMICROELECTRONICS. **Medium-density performance line ARM®-based 32-bit MCU with 64 or 128 KB Flash, USB, CAN, 7 timers, 2 ADCs, 9 com. interfaces**. [S.I.], 7 2015. Rev. 17.

SYNERTIAL. **Cobra Glove Technical Specification**. 2018. Disponível em: <<https://www.synertial.com/technical-specifications-1>>.

YOUNG, Hugh D.; FREEDMAN, Roger A. **Física III: Eletromagnetismo**. 12. ed. São Paulo, SP: Pearson Education do Brasil, 2009. ISBN 9788588639348.

ZIEREN, J.; CANZLER, U.; BAUER, B.; KRAISS, K.-F. **Advanced Man-Machine Interaction. Fundamentals and Implementation**. [S.I.]: Springer Verlag, 2006. 95 p. (Signals and Communication Technology).