

CARLOS ANDRÉ MATTEI GYORI

**Monitoramento automático de um processo de envase utilizando controlador *fuzzy*
embarcado em *hardware* de baixo custo**

Carlos André Mattei Gyori

**Monitoramento automático de um processo de envase utilizando controlador *fuzzy*
embarcado em *hardware* de baixo custo**

Dissertação apresentada à Faculdade de Engenharia do Campus de Guaratinguetá, Universidade Estadual Paulista, para obtenção do título de Mestre em Engenharia da Produção na área de Modelagem Organizacional.

Orientadora: Profa. Dra. Paloma Maria Silva Rocha Rizol

Coorientadora: Profa. Dra. Marcela Aparecida Guerreiro Machado de Freitas

Guaratinguetá - SP
2021

G997m Gyori, Carlos André Mattei
Monitoramento automático de um processo de envase utilizando controlador fuzzy embarcado em hardware de baixo custo / Carlos André Mattei Gyori. – Guaratinguetá, 2021.
106 f : il.
Bibliografia: f. 74-77

Dissertação (Mestrado). Universidade Estadual Paulista, Faculdade de Engenharia de Guaratinguetá, 2022.
Orientadora: Profa. Dra. Paloma Maria Silva Rocha Rizol
Coorientadora: : Profa. Dra. Marcela Aparecida Guerreiro Machado de Freitas

1. Controle de processo - Métodos estatísticos. 2. Arduino (Controlador programável) 3. Lógica difusa I. Título.

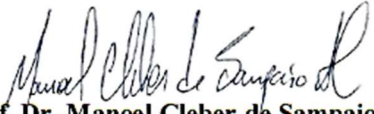
CDU 658.511.3(043)

CARLOS ANDRE MATTEI GYORI

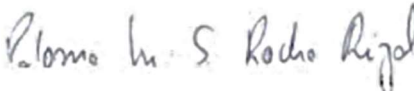
ESTA DISSERTAÇÃO FOI JULGADA ADEQUADA PARA A OBTENÇÃO DO TÍTULO DE
“MESTRE EM ENGENHARIA DE PRODUÇÃO”

PROGRAMA: ENGENHARIA DE PRODUÇÃO
CURSO: MESTRADO

APROVADA EM SUA FORMA FINAL PELO PROGRAMA DE PÓS-GRADUAÇÃO


Prof. Dr. Manoel Cleber de Sampaio Alves
Coordenador


BANCA EXAMINADORA:



Prof. Dr. PALOMA MARIA SILVA ROCHA RIZOL

Orientador - UNESP

participou por videoconferência



Prof. Dr. RUBENS ALVES DIAS

UNESP

participou por videoconferência



Prof. Dr. FABRÍCIO MACIEL GOMES

EEL/USP

participou por videoconferência

DADOS CURRICULARES

CARLOS ANDRÉ MATTEI GYORI

NASCIMENTO	13.10.1978 – Guaratinguetá / SP
FILIAÇÃO	Roberto Terner Gyori Suely Aparecida Monteiro Mattei Gyori
1999/2003	Curso de Graduação Engenharia Civil – Faculdade de Engenharia de Guaratinguetá – UNESP
2010/2012	Curso de Pós-Graduação <i>Lato Sensu</i> Especialização em Engenharia de Segurança do Trabalho – Instituto de Tecnologia do Sudoeste Paulista – INTESP
2016/2017	Curso de Pós-Graduação <i>Lato Sensu</i> Especialização em Gestão Escolar – Faculdade da Aldeia de Carapicuíba – FALC
2009/2019	Curso de Pós-Graduação <i>Lato Sensu</i> MBA em Gestão da Produção – Faculdade de Engenharia de Guaratinguetá – Unesp

Dedico este trabalho de modo especial, à minha família que sempre me apoiou e me incentivou para que essa conquista fosse alcançada.

AGRADECIMENTOS

Em primeiro lugar agradeço a Deus, fonte da vida e da graça. Agradeço pela minha vida, de minha família e de meus amigos;

a minha orientadora, *Profa. Dra. Paloma Maria Silva Rocha Rizol* e minha coorientadora *Profa. Dra. Marcela Aparecida Guerreiro Machado de Freitas* pela orientação, confiança, dedicação e incentivo, que foram imprescindíveis para a realização desse trabalho;

aos amigos *André* e *Cristóvão* pela partilha do conhecimento;

aos colegas *Raul*, *Diego* e *Regimar* pela parceria no trabalho junto à máquina;

a minha esposa *Dinara* e a minha filha *Analú*, que estiveram ao meu lado nos momentos mais difíceis, sempre me dando o apoio necessário para que eu pudesse me dedicar aos estudos.

“A tarefa não é tanto ver aquilo que ninguém viu, mas pensar o que ninguém ainda pensou sobre aquilo que todo mundo vê.”

Arthur Schopenhauer

RESUMO

O controle eficiente do processo é um elemento-chave na manutenção e melhoria da qualidade e produtividade, o que acaba se tornando um importante mecanismo para aumentar a competitividade das empresas inseridas em uma era de grandes avanços tecnológicos. O presente trabalho pretende contribuir com esta área, tendo como objeto de estudo a máquina de envase Prepac-Dermec 1000. A partir de ferramentas de controle estatístico de processo (CEP), verificou-se a necessidade de melhorar a estabilidade do volume de líquido envasado e, para atingir esse objetivo, é proposto um sistema em malha fechada que utiliza um controlador PD *fuzzy* embarcado no Arduino Mega 2560 para atuar no tempo de abertura da válvula, de modo a compensar as variabilidades do processo. O sistema de inferência *fuzzy* é construído a partir do Matlab e embarcado no Arduino utilizando a biblioteca *eFLL* desenvolvida por pesquisadores da Universidade Estadual do Piauí. Um Arduino Leonardo recebe o valor da massa medida de uma balança centesimal e os envia para o Microsoft Excel, o qual gera um gráfico desses dados em tempo real. Os resultados obtidos em simulação indicam que o modelo proposto neste trabalho é capaz de melhorar o controle da máquina, reduzindo desperdícios e agregando mais qualidade ao processo produtivo.

PALAVRAS-CHAVE: Controle de processo. Gráfico de controle. Lógica *fuzzy*. Embarcado. Arduino.

ABSTRACT

Efficient process control is a key element in maintaining and improving quality and productivity, which ends up becoming an important mechanism to increase the competitiveness of companies inserted in an era of great technological advances. The present dissertation intends to contribute to this area, having as object of study the Prepac-Dermec 1000 filling machine. Using statistical process control (SPC) tools, it was verified the need to improve the stability of the volume of the filled liquid and, to achieve this goal, a closed-loop system is proposed that uses a fuzzy-PD controller embedded in Arduino Mega 2560 to act on the valve opening time, to compensate for process variability. The fuzzy inference system (FIS) is built from Matlab and embedded in Arduino using the eFLL library developed by researchers at the State University of Piauí. A Arduino Leonardo receives the measured mass value from a centesimal scale and sends it to Microsoft Excel, which generates a graph of this data in real time. The results obtained in simulation indicate that the model proposed is capable of improving machine control, reducing waste and adding more quality to the production process.

KEYWORDS: Process control. Control charts. Fuzzy logic. Embedded. Arduino.

LISTA DE FIGURAS

Figura 1 – Rede bibliométrica por termos (2011-2020)	19
Figura 2 – Classificação da pesquisa	20
Figura 3 – Fluxograma da pesquisa	21
Figura 4 – Sistema de controle: (a) malha aberta; (b) malha fechada	24
Figura 5 – Processo isento de causas especiais	28
Figura 6 – Causa especial altera a média do processo	28
Figura 7 – Gráfico de controle das médias das massas	30
Figura 8 – Funções de pertinência para obesidade (a) Conjunto <i>crisp</i> e (b) Conjunto <i>fuzzy</i> ...	33
Figura 9 – Funções de pertinência para a variável IMC	34
Figura 10 – Sistema de inferência <i>fuzzy</i>	34
Figura 11 – Telas do <i>Fuzzy Logic Toolbox</i> do Matlab	37
Figura 12 – Placas dos Arduinos: (a) Mega 2560 e (b) Leonardo	38
Figura 13 – Máquina de envase Prepac-Dermec 1000 do DPD/UNESP	44
Figura 14 – Etapas do processo de envase	45
Figura 15 – Gráfico de controle de R : primeiro conjunto de amostras	46
Figura 16 – Gráfico de controle de \bar{X} : primeiro conjunto de amostras	46
Figura 17 – Diagrama de Ishikawa do processo de envase	47
Figura 18 – Mangueira de escape do reservatório	48
Figura 19 – Sistema de resfriamento dos selos	48
Figura 20 – Gráfico de controle de R : segundo conjunto de amostras	50
Figura 21 – Gráfico de controle de \bar{X} : segundo conjunto de amostras	50
Figura 22 – Diagrama esquemático do monitoramento e controle do processo de envase	51
Figura 23 – Função de Pertinência de Entrada: ERRO	52
Figura 24 – Função de Pertinência de Entrada: VAR_ERRO	53
Figura 25 – Função de Pertinência de Saída: VTAV	53
Figura 26 – Conjunto de regras	54
Figura 27 – Diagrama do controlador <i>fuzzy</i> no Matlab	54
Figura 28 – Visualização da saída VTAV a partir do conjunto de regras	56
Figura 29 – Superfície do resultado do modelo	57
Figura 30 – Fluxograma do código do controlador <i>fuzzy</i>	62
Figura 31 – Protótipo do sistema de monitoramento	63
Figura 32 – Segunda iteração da simulação 2 (Apêndice C): Arduino Mega 2560	68

Figura 33 – Segunda iteração da simulação 2: Matlab	68
Figura 34 – Terceira iteração da simulação 4 (Apêndice E): Arduino Mega 2560	69
Figura 35 – Terceira iteração da simulação 4: Matlab	69
Figura 36 – Gráfico de resposta do controlador <i>fuzzy</i>	70
Figura 37: Gráfico de monitoramento do processo de envase em tempo real.....	71

LISTA DE QUADROS

Quadro 1 – Produções acadêmicas relacionadas a palavras-chave da pesquisa	18
Quadro 2 – Tipos de Arduino	39
Quadro 3 – Sistema de inferência <i>fuzzy</i> do controlador	55

LISTA DE TABELAS

Tabela 1 – Massas de líquido envasado: segundo conjunto de amostras	49
Tabela 2 – Parâmetros das simulações do controlador <i>fuzzy</i> embarcado no Arduino	67

LISTA DE ABRVIATURAS E SIGLAS

CC	Corrente Contínua
CEP	Controle Estatístico do Processo
CEQ	Controle Estatístico de Qualidade
CLP	Controlador Lógico Programável
DAFLC	<i>Direct adaptive fuzzy logic controller</i> (controlador <i>fuzzy</i> adaptativo direto)
eFLL	<i>Embedded Fuzzy Logic Library</i> (Biblioteca de Lógica <i>Fuzzy</i> embarcada)
FIS	<i>Fuzzy Inference System</i> (Sistema de Inferência <i>Fuzzy</i>)
HIL	<i>Hardware in the Loop</i>
I	<i>Integral</i> (integral)
IDE	<i>Integrated Development Enviroment</i> (ambiente de desenvolvimento integrado)
IEC	<i>International Electrotechnical Commission</i>
IMC	Índice de massa corpórea
IoT	<i>Internet of Things</i> (internet das coisas)
LIC	Limite Inferior de Controle
LM	Limite Médio
LSC	Limite Superior de Controle
MAX	Máximo
MEF	Método de Elementos Finitos
MIN	Mínimo
P	<i>Proportional</i> (proporcional)
PD	<i>Proportional-derivative</i> (proporcional e derivativo)
PI	<i>Proportional-integral</i> (proporcional e integral)
PID	<i>Proportional-integral-derivative</i> (proporcional, integral e derivativo)
PWM	<i>Pulse Width Modulation</i> (modulação por largura de pulso)
TIP	Tolerância Individual Permitida
TIPI	Tolerância Individual Permitida Inferior
TIPS	Tolerância Individual Permitida Superior

SUMÁRIO

1	INTRODUÇÃO	16
1.1	CONTEXTUALIZAÇÃO DA PESQUISA	16
1.2	QUESTÕES E OBJETIVOS DA PESQUISA	17
1.2.1	Questão da Pesquisa	17
1.2.2	Objetivo Geral	17
1.2.3	Objetivos Específicos	17
1.3	DELIMITAÇÃO DA PESQUISA.....	18
1.4	JUSTIFICATIVA DA PESQUISA	18
1.5	MÉTODO DE PESQUISA	20
1.6	ESTRUTURA DO TRABALHO	22
2	REFERENCIAL TEÓRICO	23
2.1	MONITORAMENTO E CONTROLE DE PROCESSO	23
2.2	CONTROLE ESTATÍSTICO DE PROCESSO.....	26
2.2.1	Gráfico de Controle	29
2.3	LÓGICA <i>FUZZY</i>	32
2.3.1	Conjuntos <i>fuzzy</i>	32
2.3.2	Controlador <i>fuzzy</i>	34
2.4	<i>FUZZY</i> NO MATLAB.....	36
2.5	<i>HARDWARE</i> DE BAIXO CUSTO: ARDUINO.....	37
2.6	<i>FUZZY</i> EM ARDUINO.....	41
3	DESENVOLVIMENTO	44
3.1	A MÁQUINA PREPAC-DERMEC 1000 E O ATUAL PROCESSO DE ENVASE..	44
3.2	PARÂMETROS DE CONTROLE ESTATÍSTICO DO PROCESSO DE ENVASE.	45
3.3	CONTROLADOR <i>FUZZY</i> EMBARCADO EM ARDUINO.....	51
3.3.1	Sistema de inferência <i>fuzzy</i> do controlador em Matlab	52

3.3.2	Código do controlador fuzzy embarcado no Arduino Mega 2560	57
3.3.3	Conexão da balança de medição e do Microsoft Excel ao Arduino Leonardo ...	63
4	ANÁLISE DE RESULTADOS	67
4.1	CONTROLADOR PD <i>FUZZY</i>.....	67
4.2	GRÁFICO DE MONITORAMENTO	71
5	CONSIDERAÇÕES FINAIS.....	73
	REFERÊNCIAS	74
	APÊNDICE A – CÓDIGO PARA EMBARCAR O CONTROLADOR <i>FUZZY</i> NO ARDUINO MEGA 2560	78
	APÊNDICE B – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 193 g.....	89
	APÊNDICE C – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 198 g.....	93
	APÊNDICE D – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 203 g.....	97
	APÊNDICE E – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 212 g.....	99
	APÊNDICE F – CÓDIGO PARA COMUNICAR O ARDUINO LEONARDO COM A BALANÇA E O MICROSOFT EXCEL	104

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO DA PESQUISA

A evolução tecnológica contribuiu para que empresas em todo o mundo aprimorassem seus sistemas produtivos a partir da criação de novos métodos baseados na automação, robótica, inteligência artificial, Internet das Coisas (*IoT – Internet of Things*), dentre outras inovações, que são características da Quarta Revolução Industrial (MDIC, 2019). Neste contexto, ações com o propósito de melhorar a qualidade de monitoramento e controle de processos têm assumido posição de destaque nas organizações, por se tratar de um elemento fundamental para a competitividade empresarial.

Segundo Montgomery (2012), o controle estatístico do processo (CEP) é um conjunto de ferramentas de resolução de problemas, útil na obtenção da estabilidade do processo e na melhoria da capacidade por meio da redução da variabilidade. As causas especiais, que nada mais são do que variações imprevisíveis e esporádicas em sistemas produtivos, devem ser detectadas rapidamente pelo controle estatístico do processo, de modo que a investigação e a ação corretiva possam ser realizadas antes que muitas unidades não conformes sejam fabricadas e isso cause um desperdício significativo na produção. Os gráficos de controle são considerados uma ferramenta poderosa de CEP no monitoramento de processos.

O objeto de estudo deste trabalho é a máquina de envase Prepac-Dermec 1000, utilizada principalmente em indústrias lácteas, que foi modificada para encher saquinhos de 200 ml de suco, ao invés dos 1000 ml programado de fábrica. Segundo Lopes (2018), Cruz (2019) e Gyori (2019) a partir da elaboração de gráficos de controle e análise de parâmetros de CEP, verificou-se que a máquina não estava em controle. Sendo assim, foram realizadas algumas intervenções nas causas especiais identificadas no processo, coletadas novas amostras e elaborados novos gráficos, verificando-se que houve boa melhoria no controle da máquina, porém ainda apresentando alguns pontos fora dos limites de controle. A partir da análise dos parâmetros desses estudos e com o objetivo de tornar o processo mais eficiente, propõe-se um novo sistema de controle para a máquina, que utiliza lógica *fuzzy* embarcada em um Arduino Mega 2560, bem como um sistema de monitoramento que utiliza um Arduino Leonardo, que envia as massas medidas pela balança para o Microsoft Excel, para gerar gráfico em tempo real.

Sabe-se que muitas vezes as variáveis de um processo não são precisas, até mesmo por efeito de causas especiais e, nesses casos, a teoria dos conjuntos *fuzzy* proposta por Zadeh em 1965, pode ser mais adequada para promover melhorias no monitoramento e controle de um

processo. Um controlador *fuzzy* pode ser projetado para comportar-se conforme o raciocínio dedutivo, ou seja, similar ao processo que as pessoas utilizam para inferir conclusões baseadas em informações já conhecidas. A tecnologia do controlador *fuzzy* emergiu como uma ferramenta eficaz para aplicações industriais (MARQUES, 2016).

O Arduino é uma plataforma de prototipagem *open-source* que foi desenvolvida para democratizar a eletrônica a um baixo custo, de modo que programadores profissionais e amadores possam ter acesso ao código fonte, alterá-lo e/ou desenvolvê-lo de acordo com a finalidade do projeto (FONSECA, 2016). O ambiente de programação do Arduino é multiplataforma (roda em Windows, Macintosh e Linux) e a linguagem utilizada se baseia em C/C++, que é dominada pela grande maioria de estudantes e profissionais da área da engenharia (NOGUEIRA, SERVEDIO, 2015).

Neste contexto, espera-se que o sistema de controle proposto para a máquina Prepac-Dermec 1000, que utiliza um controlador PD *fuzzy* embarcado em Arduino para atuar de modo automático no tempo de abertura da válvula, possa melhorar a estabilidade do volume de líquido envasado e, aliado ao sistema de monitoramento, possa agregar mais qualidade ao processo.

1.2 QUESTÕES E OBJETIVOS DA PESQUISA

1.2.1 Questão da Pesquisa

Quão eficiente é sistema de controle em malha fechada utilizando um controlador PD *fuzzy* embarcado em *hardware* de baixo custo frente ao sistema de controle atual?

1.2.2 Objetivo Geral

Desenvolver um sistema de monitoramento e controle do processo de envase da máquina Prepac-Dermec 1000 a partir de um controlador PD *fuzzy* embarcado em Arduino.

1.2.3 Objetivos Específicos

- Apresentar referencial teórico sobre: Monitoramento e Controle de Processo, Controle Estatístico de Processo, Lógica *Fuzzy*, Controlador *Fuzzy*, *Fuzzy* em Matlab; Arduino, *Fuzzy* em Arduino;
- Apresentar os resultados do controle estatístico do processo de envase;

- Definir as variáveis de controle e o sistema de inferência do controlador *fuzzy*;
- Embarcar o controlador *fuzzy* em Arduino Mega 2560;
- Conectar o Arduino Leonardo a balança e ao Microsoft Excel.

1.3 DELIMITAÇÃO DA PESQUISA

A pesquisa está delimitada ao processo de envase da máquina Prepac-Dermec 1000 que pertence ao Departamento de Produção do Campus da Unesp de Guaratinguetá. A análise de variabilidade se limita a utilização de Diagrama de Ishikawa e gráficos de controle. O desenvolvimento do controlador *fuzzy* utiliza o Matlab, a biblioteca *eFLL (embedded Fuzzy Logic Library)* e, como *hardware* de baixo custo escolhido para o sistema, o Arduino. O sistema de monitoramento utiliza o *software* Microsoft Excel.

1.4 JUSTIFICATIVA DA PESQUISA

A partir de um levantamento de publicações acadêmicas na base de dados Scopus, foi possível verificar a relevância do tema proposto, visto que nos últimos anos houve um crescimento de pesquisas desenvolvidas nessa área. O Quadro 1 apresenta o número de publicações com as palavras *process control*, *fuzzy control*, *control charts*, e *low cost hardware*.

Quadro 1 – Produções acadêmicas relacionadas a palavras-chave da pesquisa.

termo de busca	resultados [2011-2020]	autor 1	qtde	autor 2	qtde	autor 3	qtde
fuzzy AND process AND control	11.764 	Precup, R. E.	36	Petriu, E. M.	33	Shi, P.	30
process control AND fuzzy control AND control charts*	38 	Erginel, N.	5	Şentürk, S.	5	Kahraman, C.	4
fuzzy control AND filling	52 	Zoller, C.	2	Avilés, O.	1	Deng, H.	1
process control AND fuzzy control AND embedded	23 	Espinoza, J.	6	Buele, J.	5	Silva, F.	4
process control AND fuzzy control AND low cost	11 	Espinoza, J.	6	Buele, J.	5	Silva, F.	4

Fonte: Scopus (2021).

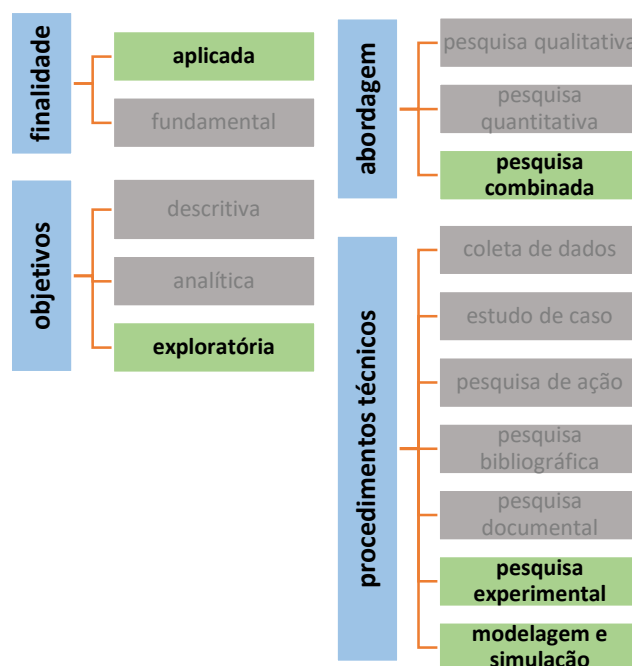
A proximidade entre os termos é maior em função da frequência com que aparecem simultaneamente nas mesmas publicações. Sendo assim, pode-se observar que os termos *fuzzy sets*, *fuzzy control*, *statistical process control*, *control charts* e *process control* apresentam compartilhamento entre si, situando-se na região central do mapa onde se evidenciam a maior concentração das conexões. Isso demonstra a relevância dos termos e que diversas pesquisas vêm sendo desenvolvidas nessas áreas.

Deve-se destacar que este trabalho é um estudo com aplicação prática e dados gerados por procedimentos experimentais, o que contribui para a área acadêmica, além de promover o desenvolvimento de competências e habilidades que podem ser primordiais para atender as exigências do mercado de trabalho contemporâneo. Ainda, cita-se a relevância do trabalho no âmbito do desenvolvimento tecnológico, visto que há potencial futuro para patente.

1.5 MÉTODO DE PESQUISA

A classificação da pesquisa deste trabalho quanto a metodologia adotada é ilustrada na Figura 2 por meio do diagrama de blocos, de acordo com o modelo de Kothari (2013), tomando como base as diretrizes apresentadas por Lakatos e Marconi (2010) e Gil (2010):

Figura 2 – Classificação da pesquisa.



Fonte: Adaptado de Kothari (2013).

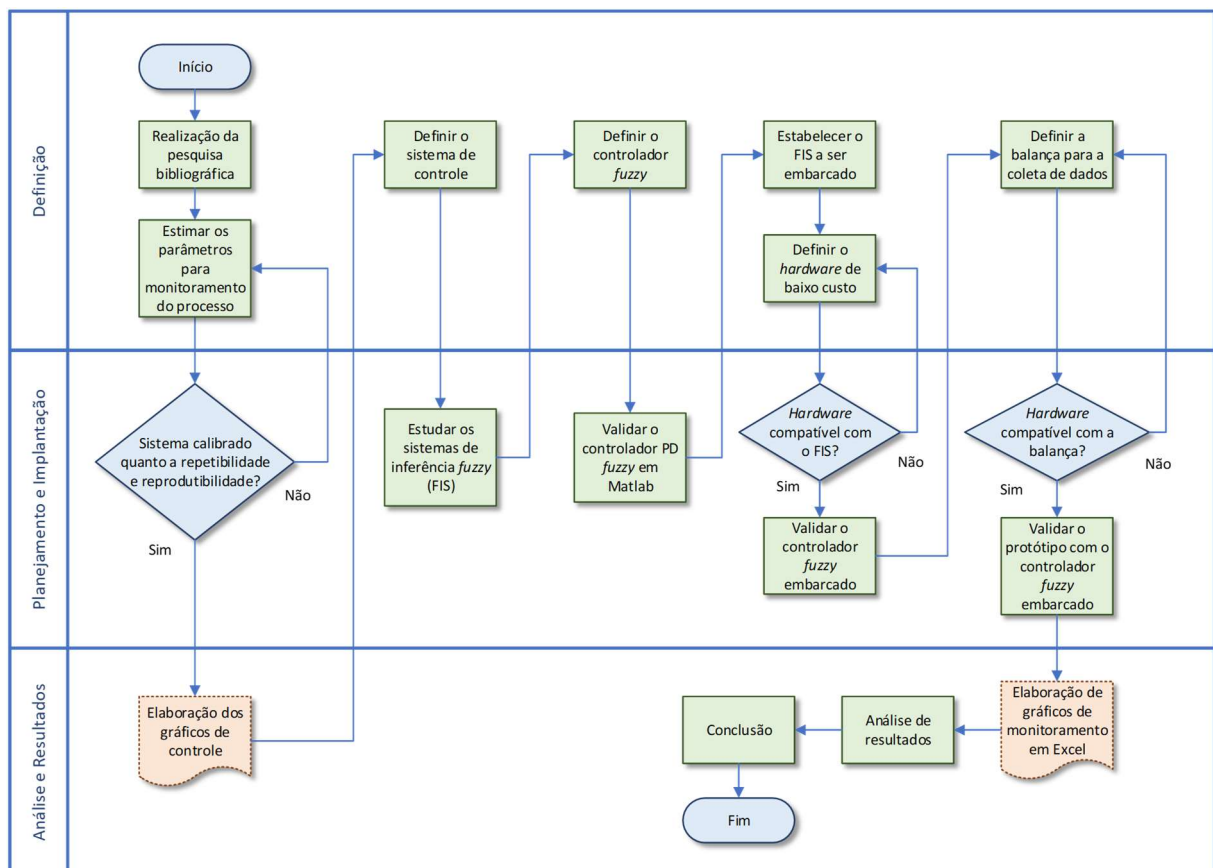
Trata-se de uma pesquisa aplicada com abordagem quantitativa e qualitativa, de caráter exploratório, com base em pesquisa bibliográfica e pesquisa experimental.

Quanto à finalidade é classificada como pesquisa aplicada, pois pretende gerar resultados para aplicação prática dirigida à solução de problemas específicos, neste caso relacionado a melhoria no processo de envase. Quanto à forma de abordagem, caracteriza-se como quantitativa e qualitativas devido às variáveis de decisão envolvidas no método proposto, podendo ser numéricas ou cognitivas (LAKATOS; MARCONI, 2010).

Quanto aos objetivos caracteriza-se como pesquisa exploratória por proporcionar maior familiaridade com o problema a fim de torná-lo explícito. Envolve levantamento bibliográfico das teorias que auxiliam no desenvolvimento do trabalho, bem como a análise de resultados experimentais (GIL, 2010). Quanto aos procedimentos técnicos, utiliza-se a pesquisa experimental por retirar amostras e efetuar intervenções na máquina de envase, além de contemplar a montagem de um protótipo e a modelagem e simulação, por utilizar *softwares* para simular os sistemas propostos, tanto de controle quanto de monitoramento (GIL, 2010).

A Figura 3 apresenta o fluxograma referente às etapas de desenvolvimento do trabalho:

Figura 3 – Fluxograma da pesquisa.



Fonte: Elaborado pelo autor.

1.6 ESTRUTURA DO TRABALHO

Os demais capítulos deste trabalho estão estruturados conforme descrição a seguir:

- Capítulo 2: referencial teórico sobre monitoramento e controle de processo, controle estatístico de processo, gráfico de controle, *fuzzy sets*, conjuntos *fuzzy*, controlador *fuzzy*, *fuzzy* em Matlab, Arduino, *fuzzy* em Arduino.
- Capítulo 3: desenvolvimento da pesquisa, que apresenta as características da máquina de envase Prepac-Dermec 1000, os estudos de CEP para o sistema de envase, a elaboração do Sistema de Inferência *Fuzzy* (FIS - *Fuzzy Inference System*) detalhada em Matlab, os procedimentos realizados para embarcar o FIS no Arduino e conectar o dispositivo à balança e ao Microsoft Excel.
- Capítulo 4: análise dos resultados da simulação do controlador *fuzzy* embarcado no Arduino.
- Capítulo 5: principais comentários e conclusões do trabalho desenvolvido.

2 REFERENCIAL TEÓRICO

2.1 MONITORAMENTO E CONTROLE DE PROCESSO

Os sistemas de controle são uma parte integrante da sociedade moderna, estando presentes em inúmeras aplicações nas mais diversas áreas. Segundo Nise (2011), sistemas de controle são constituídos por processos e subsistemas que leem uma informação, processam e ajustam de forma a obter uma saída nas condições desejadas. São construídos por quatro razões principais:

- Amplificação de potência;
- Controle remoto;
- Conveniência da forma da entrada;
- Compensação de perturbações.

Ogata (2010) apresenta definições sobre alguns componentes de um sistema de controle:

- Variável controlada: é a grandeza ou a condição que é medida e controlada. Normalmente é a saída do sistema.
- Sinal de controle ou variável manipulada: é a grandeza ou a condição variada pelo controlador de modo a afetar o valor da variável controlada.
- Planta: é uma parte de um equipamento, eventualmente um conjunto de componentes de uma máquina que funcionam integrados e cuja finalidade é desempenhar uma determinada operação.
- Processo: toda operação a ser controlada.
- Sistema: é a combinação de componentes que agem em conjunto para atingir determinado objetivo.
- Distúrbio ou perturbação: é caracterizado por um sinal que tende a afetar de modo adverso o valor da variável de saída de um sistema. Se um distúrbio for gerado internamente no sistema, ele é dito um distúrbio interno; ao passo que um distúrbio externo é produzido fora do sistema e se comporta como um sinal de entrada no sistema.
- Controle com realimentação: refere-se a uma operação que na presença de distúrbios tende a diminuir a diferença entre a saída e alguma entrada de referência que atua com base nessa diferença.

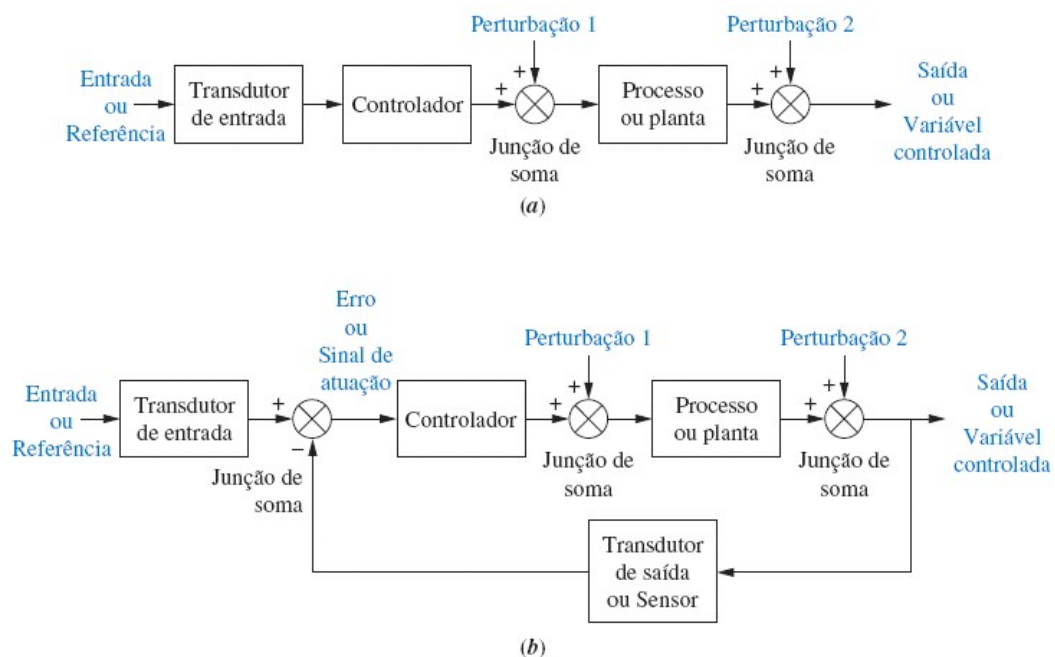
Dentro do conceito de sistemas de controle, pode-se diferenciar os tipos de controle devido sua tratativa com relação à resposta fornecida. Esta tratativa é o que define sistemas de controle de malha aberta e sistemas de controle de malha fechada.

Sistemas de controle de malha aberta são aquelas em que o sinal de saída não exerce nenhuma ação de controle no sistema, ou seja, a precisão do sistema depende de uma calibração e regulagem periódica. Na presença de distúrbios, um sistema em malha aberta não vai exercer corretamente sua função (OGATA, 2010).

Os sistemas de controle em malha fechada emitem um sinal de controle, comparam a saída real com a saída desejada e realimentam a entrada com um sinal de erro, deixando de atuar apenas quando alcançam o resultado desejado (NISE, 2011). Uma vantagem desse tipo de sistema é o fato de que o uso da realimentação faz com que a resposta seja menos sensível a distúrbios externos e a variações internas nos parâmetros do sistema (OGATA, 2010).

A Figura 4 apresenta o diagrama de blocos de ambos os modelos.

Figura 4 – Sistema de controle: (a) malha aberta; (b) malha fechada.



Fonte: Nise (2011).

Uma grande parcela dos projetos de sistemas de controle industriais utiliza controladores que são qualificados de acordo com a sua ação de controle, podendo ser classificados como (OGATA, 2010):

- Controlador liga-desliga (*on-off*): é aquele cuja saída do controlador muda de ligada para desligada, ou vice-versa, à medida que o sinal do erro passa pelo ponto de ajuste. Ele possui uma oscilação contínua da variável controlada em torno do *setpoint* (valor desejado ou valor referência), como por exemplo, o controle de temperatura de uma geladeira ou um condicionador de ar;

- Controlador Proporcional (P): ele atua para que a ação de controle seja proporcional ao erro, isto é, à diferença entre o valor desejado e o valor atual da variável controlada. Ao se aumentar o ganho proporcional, é diminuído o erro em regime permanente, ou também conhecido como regime estacionário ou *offset*, inevitável no controlador proporcional. Entretanto, não se pode aumentar indefinidamente o ganho proporcional, pois ele torna a resposta do sistema mais oscilatória, e pode levar o sistema rapidamente a saturação dos atuadores.
- Controlador Integral (I): a principal função da ação integral é fazer com que o processo busque o erro nulo. A ação de controle integral é também denominada controle de reestabelecimento (*reset*). Entretanto, a ação integral aplicada isoladamente pode piorar a estabilidade relativa do sistema, por isto é geralmente utilizada em conjunto com a ação proporcional.
- Controlador Proporcional e Integral (PI): é o controle que atua proporcionalmente ao erro, sendo este erro posteriormente anulado pela ação integral. Um dos problemas apresentados pelo controlador proporcional é a existência do erro em regime estacionário (*residual*, *offset*) na resposta a uma excitação em degrau. A ação de controle integral em conjunto com a proporcional tem como característica a eliminação de tal erro estacionário, porém apresenta uma demora na correção do valor da variável controlada;
- Controlador Proporcional e Derivativo (PD): tem como característica antecipar o erro atuante, podendo gerar na saída do controlador um valor significativo antes que o sinal do erro se torne um valor grande o suficiente para o controlador proporcional, aumentando assim a sensibilidade do controlador. Tem como característica a sobre-elevação (*overshooting*) do valor desejado, podendo ocasionar danos ao sistema controlado se for muito elevada;
- Controlador Proporcional, Integral e Derivativo (PID): é frequentemente implementado com o objetivo de unir as vantagens das ações de controle proporcional, integral e derivativa. A proporcional elimina as oscilações, a integral elimina o erro de *offset*, enquanto a derivativa fornece ao sistema uma ação antecipativa evitando previamente que o desvio se torne maior quando o processo se caracteriza por ter uma correção lenta comparada com a velocidade do desvio. Havendo possibilidade de ajuste de cada um dos ganhos separadamente, torna-se possível a chamada sintonia do controlador PID. Existem diversos métodos para se determinar a melhor sintonia, a qual faz com que o sistema tenha a melhor resposta possível em termos de erro estacionário, tempo de subida, estabilidade, tempo de acomodação entre outros fatores tanto no regime permanente quanto transitório.

Pode-se citar como principais objetivos operacionais do controle dos processos industriais (BAYER; DE ARAÚJO, 2011):

- Adaptação a perturbações externas;
- Adaptação às restrições dos equipamentos e materiais;
- Aumento da estabilidade operacional;
- Atendimento da especificação do produto;
- Otimização do uso de recursos e matéria-prima;
- Melhora nos resultados econômicos do processo;
- Segurança operacional e pessoal;
- Redução do impacto ambiental.

Como pode ser visto, os sistemas de controle, sejam eles industriais ou de outra natureza, são projetados para realizar tarefas específicas. As especificações de desempenho, como requisitos de resposta transitória, em frequência ou em regime estacionário, devem ser estabelecidas antes do início do processo de projeto. Para problemas rotineiros de projeto, as especificações relacionadas à precisão, estabilidade relativa e velocidade de resposta podem ser dadas em termos de valores numéricos precisos. Em outros casos, podem ser dadas parte em valores numéricos precisos e parte em termos de afirmações qualitativas, sendo que nesse último caso corre o risco de não serem satisfeitas (em função de requisitos conflitantes) e necessitarem de modificações, podendo encarecer o projeto. Outro fato que também pode tornar o projeto mais dispendioso é quando as especificações são mais restritivas que o necessário. Sendo assim, as especificações de desempenho devem ser estabelecidas precisamente, de modo que elas resultem em um sistema de controle ótimo para a desempenhar a função para a qual foi projetado (OGATA, 2010).

2.2 CONTROLE ESTATÍSTICO DE PROCESSO

O controle estatístico do processo (CEP) é uma ferramenta de resolução de problemas, útil na obtenção da estabilidade do processo e na melhoria da capacidade por meio da redução da variabilidade (MONTGOMERY, 2012).

Desde o início da Revolução Industrial, Shewhart preocupou-se em estudar a variabilidade dos processos. Suas explicações sobre a impossibilidade de produzirem itens exatamente iguais são aceitas até hoje. Segundo ele, todo e qualquer processo, por mais bem projetado e por mais bem controlado que seja, possui em sua variabilidade um componente impossível de ser eliminado. Trata-se da variabilidade natural do processo, que é fruto de uma série de pequenas perturbações, ou causas

aleatórias, contra as quais pouco ou nada se pode fazer [...] (COSTA; EPPRECHT; CARPINETTI, 2005).

Montgomery (2012) define essa variabilidade como “efeito cumulativo de pequenas causas inevitáveis”. Um processo que manifeste essa característica, em reduzida escala, caracteriza variações aleatórias, porém estáveis, presentes em qualquer processo produtivo. Entretanto, causas especiais ou atribuíveis são causas de variabilidade que não são aleatórias, sendo que essas influenciam muito mais no processo do que as causas aleatórias, causando um desempenho abaixo do aceitável do processo.

As principais causas especiais encontradas no setor industrial são (SOARES, 2001):

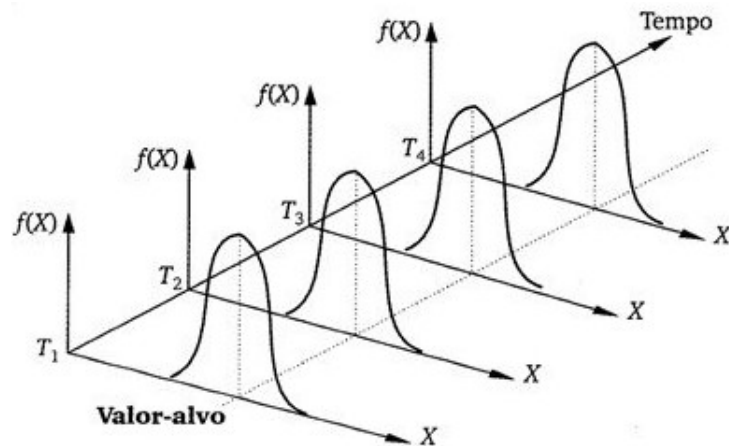
- Desgaste de ferramentas;
- Mancais desajustados;
- Vibrações nas máquinas;
- Fixações malfeitas;
- Matéria prima de má qualidade;
- Falta de cuidado/treino dos operadores;
- Mudanças climáticas;
- Máquinas inadequadas ao processo;
- Gerenciamento inadequado;
- Erros em medições.

De acordo com Ribeiro e Ten Caten (2012), o CEP é uma técnica estatística aplicada à produção que permite a redução sistemática da variabilidade nas características da qualidade de interesse, contribuindo para a melhoria da qualidade intrínseca, da produtividade, da confiabilidade e do custo do que está sendo produzido. Sendo assim, pode-se dizer que o principal objetivo do CEP é possibilitar um controle eficaz da qualidade em processos produtivos.

O controle estatístico teve início nos Estados Unidos na década de XX, como resultado de avanços tecnológicos na área de medição e da aplicação industrial dos gráficos de controle, desenvolvidos pelo Dr. Walter A. Shewhart, da empresa de telefonia Bell Telephone Laboratories. A Segunda Guerra Mundial foi decisiva para a aplicação do controle de qualidade e da estatística moderna em um maior número de indústrias americanas e logo após esse período, foi a vez do Japão começar a adotar o controle estatístico da qualidade (CEQ), seguindo os padrões americanos (MONTGOMERY, 2012).

O CEP representa uma evolução do CEQ, por envolver o monitoramento de processo, cujo objetivo é verificar se ele está em controle, ou seja, apresenta apenas a variabilidade natural devida às causas aleatórias. A Figura 5 mostra um processo isento de causas especiais:

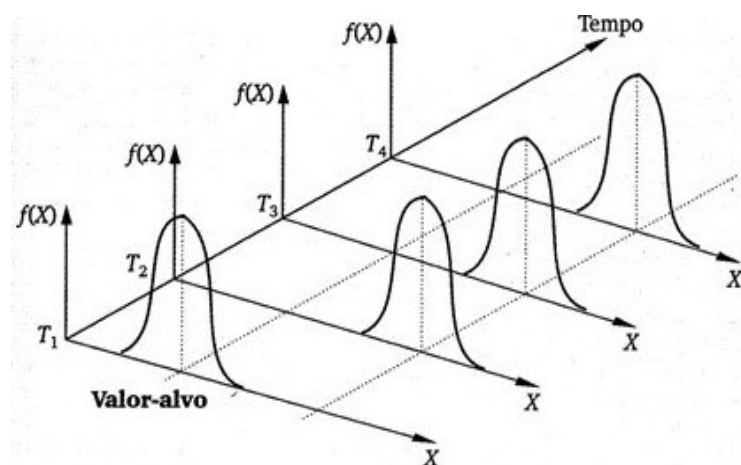
Figura 5 – Processo isento de causas especiais.



Fonte: Costa, Epprecht e Carpinetti (2005).

Caso seja verificado que o processo não está em controle, as causas especiais devem ser detectadas e corrigidas rapidamente, evitando que muitas unidades não conformes sejam fabricadas. A Figura 6 mostra um processo com a média alterada por causas especiais:

Figura 6 – Causa especial altera a média do processo.



Fonte: Costa, Epprecht e Carpinetti (2005).

As “Sete Ferramentas da Qualidade”, que podem auxiliar na gestão de processos, são (MONTGOMERY, 2012):

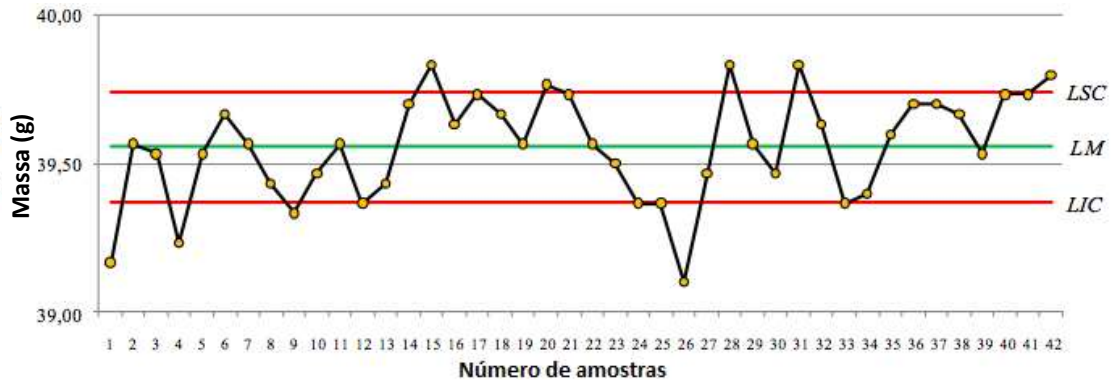
- Histograma: tem como finalidade mostrar a distribuição dos dados através de um gráfico de barras indicando o número de unidades em cada categoria.
- Folha de Verificação: são tabelas ou planilhas simples usadas para facilitar a coleta, apresentação e análise de dados.
- Gráfico de Pareto: é composto por um gráfico de barras que ordena as frequências das ocorrências em ordem decrescente (da maior para a menor), e permite a localização de problemas vitais e a eliminação de futuras perdas.
- Diagrama de Causa-e-Efeito (também conhecido como Diagrama de Ishikawa ou Diagrama Espinha de Peixe): tem como finalidade explorar e indicar todas as causas possíveis de uma condição ou um problema específico.
- Fluxograma: é uma ilustração sequencial de todas as etapas de um processo, que mostra como cada etapa é relacionada. Utiliza símbolos facilmente reconhecidos para denotar os diferentes tipos de operações em um processo.
- Diagrama de Dispersão: mostra o que acontece com uma variável quando a outra sofre uma alteração, para testar possíveis relações de causa e efeito.
- Gráfico de Controle: é usado para monitorar um indicador ou característica de processo ao longo do tempo. Por ser considerada uma das mais importantes ferramentas de CEP para investigar a variação em um processo, será feita uma abordagem mais detalhada desse assunto no próximo tópico.

2.2.1 Gráfico de Controle

O gráfico de controle desenvolvido por Shewhart é uma ferramenta poderosa de CEP, capaz de detectar a presença de causas especiais em um processo, a partir da análise de dados provenientes de amostragem. A frequência da amostragem deve ser compatível com as principais causas de variabilidade presentes no sistema.

A proposta de se construir um gráfico de controle de um processo é substituir a mera detecção e correção de produtos defeituosos pelo estudo e prevenção dos problemas relacionados à qualidade (RIBEIRO; TEM CATEN, 2012). A Figura 7 apresenta o exemplo de um gráfico de controle:

Figura 7 – Gráfico de controle das médias das massas.



Fonte: Adaptado de Saldanha *et al.* (2013).

Como pode ser observado na Figura 7, um gráfico de controle é composto por uma Linha Média (LM), um Limite Superior de Controle (LSC) e um Limite Inferior de Controle (LIC). Se ao plotar um gráfico de controle, os pontos se distribuem em torno de LM, sem ultrapassar os limites LSC e LIC, não se deve intervir no processo, pois as variações são decorrentes de causas aleatórias, intrínsecas ao processo. Entretanto, se forem observados pontos acima de LSC e/ou abaixo de LIC, como no caso do exemplo da Figura 7, isso pode indicar a presença de causas especiais, bem como a necessidade de alguma ação corretiva no processo (COSTA; EPPRECHT; CARPINETTI, 2005).

Quando a variável a ser controlada é uma variável contínua, é usual monitorar o processo a partir de dois gráficos de controle: o da média amostral \bar{X} , para monitorar a centralidade, e o da amplitude amostral R (diferença entre o maior e o menor valores da amostra), para monitorar a dispersão. A Linha Média (LM) para os gráficos é localizada no valor da média (valor esperado) da variável, sendo que os limites de controle (LSC e LIC) são estabelecidos a três desvios padrão dessa. Tais limites são considerados “Limites de Ação”, pois indica a necessidade de investigação e de uma ação corretiva sobre a causa especial que originou o problema média (COSTA; EPPRECHT; CARPINETTI, 2005).

São considerados “Limites de Alerta” linhas que se distanciam de LM de dois desvios padrão e, quando existem um ou mais pontos entre os limites de alerta e controle, deve-se suspeitar que o processo pode não estar operando adequadamente. Uma possível ação a ser tomada quando isso ocorre é aumentar a frequência de amostragem e/ou o tamanho da amostra para se obter rapidamente mais informações sobre o processo e evitar, assim, a ocorrência de alarmes falsos (MONTGOMERY, 2012).

A seguir, apresenta-se algumas regras que são utilizadas na prática para aumentar a sensibilidade dos gráficos de controle (MONTGOMERY, 2012):

- Um ou mais pontos fora dos limites de controle (LSC e LIC);
- Dois, de três pontos consecutivos, fora dos limites de alerta;
- Quatro, de cinco pontos consecutivos, além dos limites de um desvio padrão;
- Uma sequência de oito pontos consecutivos de um mesmo lado de LM;
- Seis pontos de uma sequência em tendência crescente ou decrescente;
- Quinze pontos em sequência na faixa entre mais ou menos um desvio padrão;
- Quatorze pontos em sequência alternadamente para cima e para baixo;
- Oito pontos em sequência de ambos os lados de LM, com nenhum deles acima de três desvios padrão.

Montgomery (2012) apresenta cinco características dos gráficos de controle que justificam a sua grande utilização em processos industriais:

- São considerados uma técnica com potencial para aumentar a produtividade, uma vez que podem auxiliar na redução de perdas de material, custos e retrabalho.
- Ajudam a manter o processo em controle, o que contribui para a prevenção de defeitos e os produtos são produzidos com uma qualidade melhor, seguindo a filosofia: “Faça certo da primeira vez”.
- Evitam ajustes desnecessários do processo, pois distinguem que tipo de causas está atuando, se são causas aleatórias ou especiais. Ajustes desnecessários podem prejudicar o desempenho do processo.
- Fornecem diagnósticos da situação do processo, o que auxilia os operadores do processo a implementarem mudanças que melhoram seu desempenho.
- Fornecem informações sobre a capacidade do processo, sendo estas de grande utilidade para projetistas de produtos e processos.

A metodologia de controle estatístico de processo foi, inicialmente, aplicada ao contexto industrial e, por isso, as aplicações nesse contexto estão bem consolidadas. No entanto, os princípios são gerais, o que permite aplicação da teoria em outros segmentos. Em alguns casos, essa tarefa pode ser mais complicada pela falta de um sistema de medição quantitativo e objetivo (MONTGOMERY, 2012).

Diante do exposto, nota-se que o CEP é capaz de promover melhorias na qualidade de processos, independentemente de sua natureza, sendo uma ferramenta chave para que

organizações alcancem sucesso, crescimento e a uma melhor posição de competitividade em seu mercado de atuação.

2.3 LÓGICA FUZZY

2.3.1 Conjuntos *fuzzy*

A teoria de conjuntos *fuzzy* e os conceitos de lógica *fuzzy* podem ser utilizados para traduzir em termos matemáticos a informação imprecisa expressa por um conjunto de regras com variáveis linguísticas. O resultado é um sistema de inferência com base em regras, no qual a teoria de conjuntos *fuzzy* e lógica *fuzzy* fornecem o ferramental matemático para se lidar com as tais regras. Essa teoria foi concebida por Lofti A. Zadeh, em 1965, com objetivo de realizar um tratamento das informações de caráter impreciso ou vago, sendo uma relevante estratégia para controle ou monitoramento de processos quando necessária a interpretação e análise de dados não precisos (ZADEH, 1965; TANSCHHEIT, 2007).

Os primeiros registros da aplicação bem-sucedida da lógica *fuzzy* em indústrias foi na área de controle. Na Dinamarca, foi realizado o controle de um forno de cimento em 1980. No mesmo período no Japão, a lógica *fuzzy* foi aplicada no controle de um processo de purificação de água e em um sistema de controle automático de trem. Desde então, tem-se verificado uma utilização crescente dos sistemas *fuzzy* em diversos campos (TANAKA, 1996).

Dentre as vantagens da utilização da lógica *fuzzy* tem-se o mecanismo de raciocínio similar ao do ser humano, por meio do uso de termos linguísticos; modelagem de conhecimento de senso comum; conhecimento ambíguo e conhecimento impreciso, mas racional; técnica de aproximação universal; robustez e tolerância à falha; além do baixo custo de desenvolvimento e de manutenção. As limitações estão relacionadas à geração das regras *fuzzy* e à definição das funções de pertinência; ambas, baseadas em uma avaliação subjetiva do conhecimento do especialista. Adiciona-se a isso a inexistência de técnicas de aprendizado (NOGUEIRA, 2013).

Segundo a teoria clássica de conjuntos, um conjunto A em um universo X , os elementos simplesmente pertencem ou não àquele conjunto, sendo este denominado conjunto *crisp* (não-*fuzzy*). É expresso pela função característica μ_A (TANSCHHEIT, 2007):

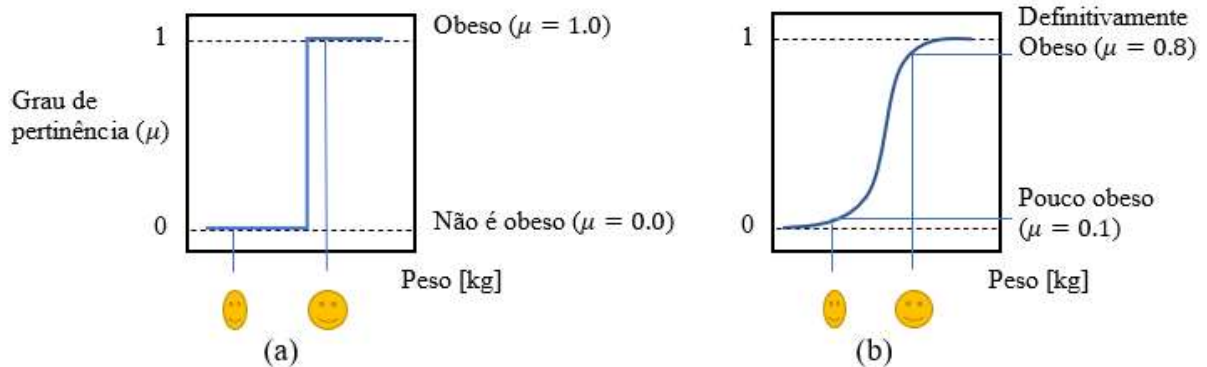
$$\mu_A(x) = \begin{cases} 1 & \text{se e somente se } x \in A \\ 0 & \text{se e somente se } x \notin A \end{cases} \quad (1)$$

Zadeh propôs uma caracterização mais ampla, generalizando a função característica de modo que ela pudesse assumir um número infinito de valores no intervalo $[0,1]$. Um conjunto *fuzzy* A em um universo de discurso X é definido por uma função de pertinência $\mu_A(x): X \rightarrow [0,1]$, é representado por um conjunto de pares ordenados:

$$A = \{(x, \mu_A(x)) \mid x \in X\} \quad (2)$$

Um determinado elemento pode pertencer a mais de um conjunto *fuzzy*, com diferentes graus de pertinência, podendo ser definido em universos contínuos ou discretos (TANSCHEIT, 2007). As funções de pertinência podem ter formas do tipo triangular, trapezoidal, gaussiano, tipo-S ou tipo-Z, cuja curva define como cada ponto de entrada é mapeado em um valor de grau de pertinência entre 0 e 1. A Figura 8 mostra o exemplo da função de pertinência para identificar o grau de obesidade de uma pessoa por um conjunto *crisp* (não-*fuzzy*) (a) e um conjunto *fuzzy* (b).

Figura 8 – Funções de pertinência para obesidade: (a) Conjunto *crisp* e (b) Conjunto *fuzzy*

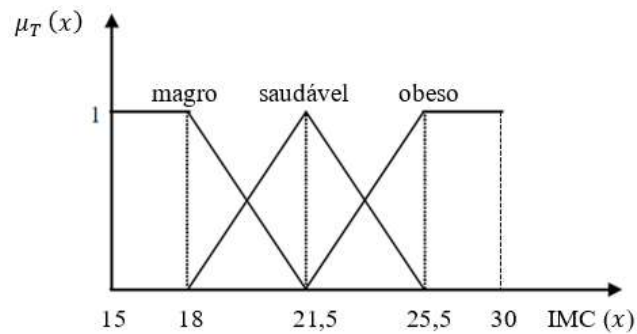


Fonte: Elaborado pelo autor.

As funções de pertinência *fuzzy* podem ser representadas por variáveis linguísticas que podem ser expressas por palavras ou sentenças usadas na língua. Uma variável linguística é definida por três elementos principais: $(x, T(x), X)$, sendo x o nome da variável, $T(x)$ um conjunto de valores linguísticos para os valores de x e X o universo de discurso.

Por exemplo, no caso da variável linguística índice de massa corpórea $T(\text{IMC})$, tem-se: $T(\text{IMC}) = \{\text{magro, saudável, obeso}\}$. Cada termo em $T(\text{IMC})$ é caracterizado por um conjunto *fuzzy* em um universo de discurso $X = [15, 30]$. Estes valores são descritos por funções de pertinência trapezoidais e triangular, conforme exemplificado na Figura 9 (RIZOL; DIAS, 2014).

Figura 9 – Funções de pertinência para a variável IMC.

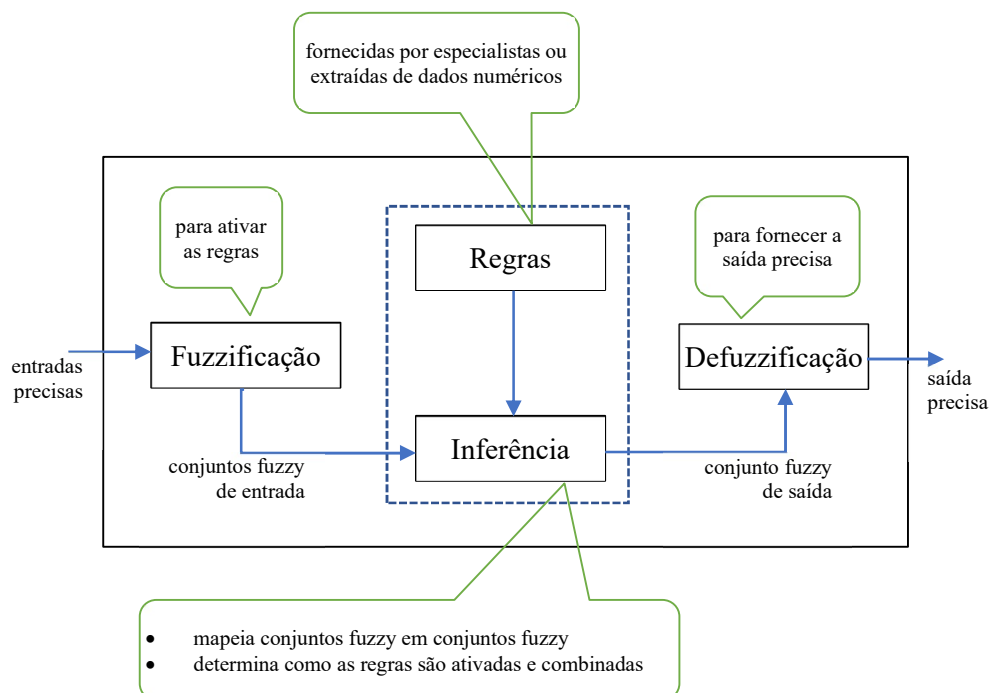


Fonte: Adaptado de Rizol e Dias (2014).

2.3.2 Controlador *fuzzy*

A tecnologia do controlador *fuzzy* emergiu como uma das tecnologias de controle não linear mais eficazes usadas em aplicações industriais. Tudo começou na década de 1970 com o trabalho pioneiro de E.H. Mamdani e Assilam (1975) e a primeira aplicação bem-sucedida da lógica *fuzzy* para controle de motores a vapor, seguida por contribuições de muitos outros cientistas e engenheiros. Desde então, o espectro de aplicações de controladores *fuzzy* está em constante crescimento (KOVACIC, BOGDAN, FAULKNER, 2005). A Figura 10 apresenta a estrutura básica de um controlador *fuzzy*.

Figura 10 – Sistema de inferência *fuzzy*.



Fonte: Adaptado de Tanscheit (2007).

Como pode ser observado na Figura 10, um controlador *fuzzy* é composto por quatro estágios principais: fuzzificação, regras, inferência e defuzzificação (MENDEL, 2001).

A primeira etapa, denominada fuzzificação, tem como função codificar as variáveis de entrada *crisp* em valores *fuzzy*. Essa codificação é baseada no conhecimento do especialista.

As regras ou conjunto de regras devem ser consistentes, podendo ser fornecidas por especialistas, em forma de sentenças linguísticas, caracterizando fundamental importância no controlador *fuzzy*. Extrair regras de especialistas na forma de sentenças do tipo *se...então* não é uma tarefa fácil, por mais experientes que sejam acerca do problema em questão.

O número total de regras em um sistema de controle *fuzzy* depende do número de entradas do controlador e do número de funções de pertinência de cada entrada. Se um controlador tem p entradas com a quantidade n de funções de pertinência, o número total de regras, R é dado por (NOGUEIRA, 2013):

$$R = n^p \quad (3)$$

A inferência ou Sistemas de Inferência *Fuzzy* (FIS - *Fuzzy Inference System*) trata processos bastante complexos, onde existem informações incertas ou imprecisas. Para tanto os sistemas de inferência *fuzzy* utilizam as regras linguísticas definidas pelo especialista (LEAL, 2011).

Os principais métodos de inferência são Mamdani e Sugeno. Os controladores Mamdani são de fácil modelagem por se basearem no conhecimento de especialistas e indicados quando se trata de um controle mais grosseiro. Para controles mais finos, o controlador Sugeno apresenta desempenho superior, porém aumenta a complexidade da modelagem por não ser mais intuitiva e sim matemática (SUGENO, 1974; NOGUEIRA, 2013).

Uma vez obtido o conjunto *fuzzy* de saída pelo processo de inferência, na defuzzificação é efetuada uma interpretação dessa informação, ou seja, converte os valores *fuzzy* em valores reais (*crisp*).

Destaca-se os seguintes métodos de defuzzificação, segundo Ibrahim (2004):

- **Centroide:** também conhecido como o centro de massa ou o centro de gravidade. É o método mais usado para defuzzificação. O valor numérico obtido representa o centro de gravidade da distribuição de possibilidade de saída do sistema *fuzzy*.
- **Centro de maior área:** aplicado quando a saída é constituída por, pelo menos, dois subconjuntos *fuzzy* convexos que não se sobrepõem.

- **Primeiro dos máximos:** considera o primeiro ponto entre os valores que tem o maior grau de pertinência inferido pelas regras
- **Método da média ponderada:** Este método é válido para funções de pertinência simétricas na saída. Produz um valor numérico considerando a média ponderada dos valores dos picos, sendo as massas os graus de pertinência.
- **Média dos máximos:** Encontra o ponto médio entre os valores que tem o maior grau de pertinência inferido pelas regras. No caso de ser único o valor de defuzzificação é a própria abscissa deste pico.

A seleção do método depende do conhecimento do especialista, devendo ser considerada a complexidade computacional, o sistema de aplicação e a viabilidade dos resultados de um ponto de vista da engenharia.

2.4 FUZZY NO MATLAB

O Matlab (*MATrix LABORatory*) é um programa produzido pela Mathworks, Inc. que serve para tratar matrizes e números complexos da mesma forma que uma calculadora trata números reais. Além disso, possui recursos de programação, agindo como uma linguagem interpretada semelhante à linguagem C ou Pascal, porém voltada para o processamento numérico intensivo. Contém também programas para projeto de controle e recursos gráficos (LOTUFO, 2009).

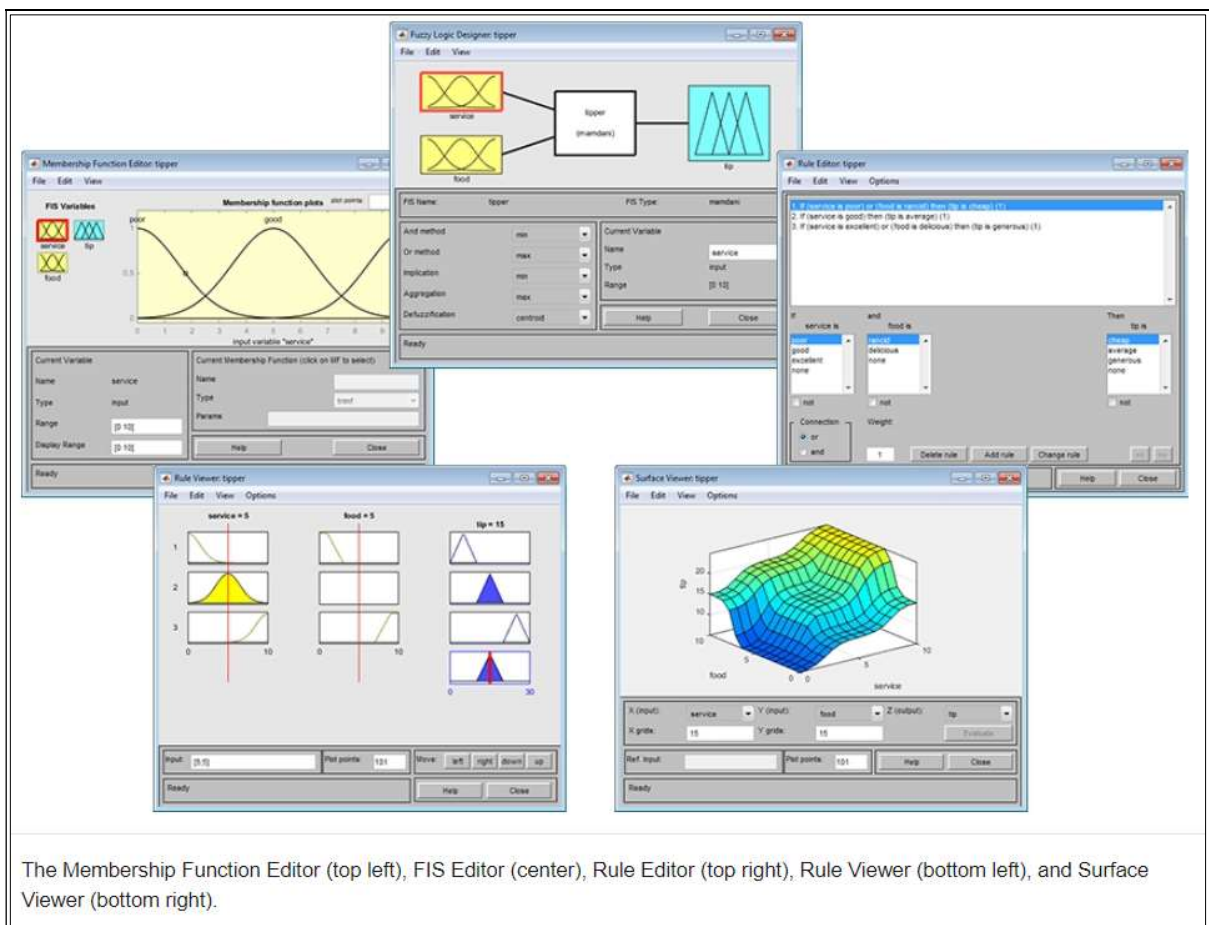
O Simulink, também produzido pela Mathworks, Inc., é um dos pacotes nativos do Matlab. Trata-se de uma ferramenta de diagramação gráfica por blocos e bibliotecas customizáveis, destinado para modelagem, simulação e análise de sistemas dinâmicos. Ele suporta projetos de sistemas, simulação, geração de código automático e teste contínuo e verificação de sistemas embutidos, incluindo a modelagem de controladores *fuzzy*, com possibilidade de incorporar algoritmos e exportar resultados para outras plataformas de análises (LOTUFO, 2009; MATHWORKS, 2021).

O bloco do Simulink também fornece suporte integrado para o desenvolvimento de protótipos, testes e modelos de execução em *hardware in the loop* (HIL) em *hardwares* de baixo custo, como Arduino e Raspberry Pi, por exemplo. Desse modo, é possível desenvolver algoritmos para sistemas de controle e embarcá-los em um *hardware* adequado, permitindo, inclusive, simulação em tempo real, incorporação de algoritmos em modelos e exportação dos resultados da simulação para análise posterior.

Segundo Xue e Chen (2014), o Matlab é uma plataforma de *software* que pode ser utilizada para modelagem, análise e projeto de sistemas de controle, com inúmeras caixas de ferramentas e conjuntos de blocos (*toolbox*), cada um constando de uma coleção de arquivos destinados a tratar diversas classes de problemas científicos, dentre eles o *Fuzzy Logic Toolbox* que vem a ser o de principal interesse para este trabalho.

A Figura 11 apresenta as telas do *Fuzzy Logic Toolbox* do Matlab:

Figura 11 – Telas do *Fuzzy Logic Toolbox* do Matlab.



Fonte: Mathworks (2021).

2.5 HARDWARE DE BAIXO CUSTO: ARDUINO

Esta plataforma de prototipagem eletrônica *open-source* foi desenvolvida com o intuito de democratizar a eletrônica e a robótica, ou seja, permite que programadores profissionais e amadores possam ter acesso ao código fonte, alterá-lo e/ou desenvolvê-lo de acordo com as necessidades específicas de cada projeto (FONSECA, 2016).

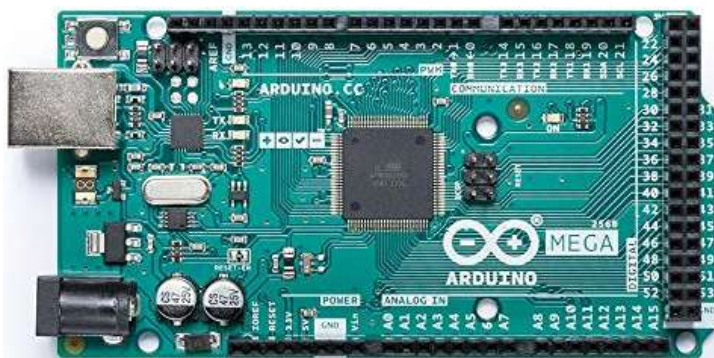
O Arduino foi criado em 2005 por um grupo de 5 pesquisadores: Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis. O objetivo era elaborar um dispositivo que fosse ao mesmo tempo barato, funcional e fácil de programar, sendo dessa forma acessível a estudantes e projetistas amadores. Além disso, foi adotado o conceito de *hardware* livre, o que significa que qualquer um pode montar, modificar, melhorar e personalizar o Arduino, partindo do mesmo *hardware* básico. Assim, foi criada uma placa composta por um microcontrolador Atmel, circuitos de entrada/saída e que pode ser facilmente conectada à um computador e programada via IDE (*Integrated Development Environment*, ou Ambiente de Desenvolvimento Integrado) utilizando uma linguagem baseada em C/C++, sem a necessidade de equipamentos extras além de um cabo USB (THOMSEN, 2018).

O ambiente de programação do Arduino é multiplataforma (roda em Windows, Macintosh e Linux) e está disponível gratuitamente em Arduino (2021). Para programar o Arduino basta conectá-lo via cabo USB a um computador que possua o *software* instalado. A linguagem de programação utilizada se baseia em C/C++. Esta linguagem é dominada pela grande maioria de estudantes e profissionais da área da engenharia. E como se trata de um *hardware* de baixo custo, a substituição é simples e barata (NOGUEIRA, SERVEDIO, 2015).

Uma das características fundamentais do Arduino é a facilidade de utilização, a qual permite uma grande diversidade de aplicações, tanto em nível de *software* como de *hardware*, tornando a automação muito mais acessível e economicamente viável (BANZI; SHILOH, 2014).

A Figura 12 apresenta a placa do Arduino Mega 2560 e do Arduino Leonardo que, por sua vez, foram escolhidos para serem utilizados no projeto proposto:

Figura 12 – Placas dos Arduinos: (a) Mega 2560 e (b) Leonardo.



(a) Arduino Mega 2560



(b) Arduino Leonardo

Fonte: Arduino (2021).


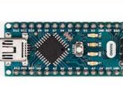





A placa do Arduino é onde se conecta toda a parte física de um projeto. Depois de programado, o microcontrolador pode ser usado de forma independente (*stand alone*), ou seja,

pode ser utilizado para controlar um robô, as luzes de uma residência, a temperatura do aparelho de ar-condicionado, uma máquina de porte industrial ou qualquer outro projeto.

Existem no mercado diversos modelos de placa Arduino. A utilização de uma ou outra placa depende do projeto a ser desenvolvido e o número de portas necessárias. As opções vão das mais comuns, como o Arduino UNO e suas 14 portas digitais e 6 analógicas, passando por placas com maior poder de processamento, como o Arduino MEGA, com microcontrolador ATmega2560 e 54 portas digitais, e o Arduino DUE, baseado em processador ARM de 32 bits e 512 kBytes de memória.

No Quadro 2 são apresentados os modelos mais populares segundo o site do fabricante e suas principais especificações:

Quadro 2 – Tipos de Arduino.

Placa	1 LEONARDO	2 NANO	3 UNO	4 MICRO	5 DUE	6 MEGA2560	7 YÚN
Característica							
Imagem							
Microcontrolador	ATmega32u4	ATmega328	ATmega328	ATmega32u4	AT91SAM3X8E	ATmega2560	ATmega32u4
Portas digitais	20	14	14	20	54	54	20
portas PWM	7	6	6	7	12	15	7
Portas analógicas	12	8	6	12	12	16	12
Memória	32K	32k	32K	32k	512K	256K	32K
Clock	16MHz	16MHz	16MHz	16MHz	84MHz	16MHz	16MHz
Conexão	Micro USB	USB mini-B	USB	Micro USB	Micro USB	USB	Micro USB
Conector para alim. externa	Sim	Não	Sim	Não	Sim	Sim	Sim
Tensão de operação	5V	5V	5V	5V	3.3V	5V	5V
Corrente máxima portas E/S	40mA	40mA	40mA	20mA	130mA	40mA	40mA
Alimentação	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc	7-12Vdc
Preço (USD) nov/2021	18.40	17.59	19.55	18.40	34.25	34.25	56.40

Fonte: Arduino (2021).

Projetos com Arduino podem ser embarcados (*stand-alone*), ou seja, algoritmo executado no próprio Arduino, ou podem se comunicar com *software* rodado em computador (ARDUINO, 2021). Sendo assim, o microcontrolador pode interagir, por exemplo, com o Simulink que é um ambiente de simulação do Matlab, conforme apresentado no capítulo anterior. Um sistema de controle pode ser totalmente executado no Simulink, sendo que o programa executado no Arduino fica responsável apenas pela aquisição de dados entre o *software* e o microcontrolador. Para que isso seja possível, deve-se primeiramente carregar na memória do Arduino um algoritmo exclusivamente desenvolvido para configurar e administrar a comunicação entre o Matlab e o Arduino (NOGUEIRA, SERVEDIO, 2015).

Segundo Pearce (2020), tanto *softwares* livres quanto os de código aberto (*open source*), bem como a fabricação e comercialização de *hardwares* de código aberto e gratuito, mostram-se promissores para a área da ciência e tecnologia, pois um pesquisador pode criar e publicar um projeto e, deste modo, beneficiar as mais diversas áreas. Isso leva a uma oportunidade imediata de alcançar as melhores práticas, de modo que seja possível que todos no campo das pesquisas continuem a impulsionar a ciência. Além disso, esse tipo de tecnologia é acessível em todos os níveis do sistema educacional, o que também contribui para que uma porcentagem maior dos pesquisadores do mundo seja capaz de participar da ciência experimental. Dados os resultados deste estudo, é viável, do ponto de vista financeiro, priorizar as tecnologias de código aberto para a pesquisa científica, visto que, em geral, as economias de custo para projetos que utilizam esses dispositivos são significativas. Verificou-se que a utilização do Arduino, por exemplo, pode resultar em uma redução em torno de 89% nos custos de um projeto, quando comparado a uma tecnologia que desempenha a mesma função, mas que não é *open source* (PEARCE, 2020).

Minchala *et al.* (2020) apresentam uma avaliação de desempenho do desenvolvimento da instrumentação, sistemas de comunicação e controle de um processo de dois tanques, em que o Arduino consiste em um dos *hardwares* que são integrados a controladores lógicos programáveis (CLPs) para realizarem a automatização do processo e que são conectados a um sistema de controle de monitoramento e aquisição de dados. Os resultados encontrados neste trabalho foram satisfatórios, principalmente quanto à melhoria da comunicação virtual entre o sistema de monitoramento e o sistema de controle, feita por meio da plataforma *Thingier.io*, que permite a criação de painéis e gráficos de visualização em tempo real para monitoramento remoto. Trata-se de uma solução simples, porém robusta, que envolve uma tecnologia de baixo custo e código de fonte aberta, principalmente, quando comparado aos sistemas de controle convencionais encontrados nas indústrias que utilizam CLPs, cuja programação se baseia no

padrão IEC 61131 (que não está em conformidade com os requisitos para interface remota) e possuem menor velocidade de processamento, memória e velocidade de leitura das portas de entrada do que quando integrados a dispositivos como Arduino (MINCHALA *et al.*, 2020).

2.6 FUZZY EM ARDUINO

A biblioteca *eFLL* (*embedded fuzzy logic library*) é um projeto desenvolvido pelo *Robotic Research Group* na Universidade Estadual do Piauí (UESPI-Teresina). Escrita em C++/C, é uma opção versátil, leve e eficiente de operar a lógica *fuzzy* em sistemas embarcados, não possuindo limitações explícitas de quantidade de conjuntos, regras, entradas ou saídas, devendo ser considerada apenas a capacidade de processamento e armazenamento de cada microcontrolador. A biblioteca utiliza o método de inferência Mamdani, também conhecido como MAX e MIN, e para a defuzzificação, o Método do Centroide. Ainda, a solução é destinada não somente ao Arduino, mas a qualquer sistema, embarcado ou não, cuja programação seja feita em linguagem C. A biblioteca *eFLL* possui os seguintes objetos (ALVES, 2012):

- **Objeto *fuzzy***: engloba todo o sistema *fuzzy*, por meio dele pode-se manipular os conjuntos *fuzzy*, as regras de inferência, entradas e saídas.
- **Objeto *fuzzyInput***: agrupa todos os conjuntos *fuzzy* de entradas que pertencem ao mesmo domínio.
- **Objeto *fuzzyOutput***: agrupa todos os conjuntos *fuzzy* de saída que pertencem ao mesmo domínio.
- **Objeto *fuzzySet***: um dos principais objetos da biblioteca *fuzzy*, com o qual é possível modelar cada conjunto do sistema em questão. Atualmente a biblioteca suporta as funções de pertinência triangular, trapezoidal e *singleton*, que são montadas com base nos pontos A, B, C e D, passados por parâmetros no construtor *fuzzySet* (*float a, float b, float c, float d*).

Os seguintes objetos auxiliam na montagem das funções de pertinência e nas regras de inferência *fuzzy* (ALVES, 2012):

- **Objeto *fuzzyRule***: utilizado para construir o conjunto de regras do Objeto *fuzzy*;
- **Objeto *fuzzyRuleAntecedent***: usado para criar o Objeto *fuzzyRule*, sendo responsável por fazer a expressão condicional do antecedente de um *fuzzyRule*;
- **Objeto *fuzzyRuleConsequente***: usado para renderizar o objeto *fuzzyRule*, responsável por retornar o valor da expressão condicional.

As referências pesquisadas mostram que existem diversos trabalhos sendo desenvolvidos que utilizam a lógica *fuzzy* embarcada em Arduino. Zaki *et al.* (2018) apresentam uma proposta de um controlador *fuzzy* adaptativo direto (DAFLC) para fazer o controle preciso da velocidade de um motor de corrente contínua (CC), estimado a partir de dois níveis, sendo que o nível inferior usa o controlador *fuzzy* Mamdani e o superior usa o modelo inverso baseado no método Takagi-Sugeno, cuja saída é usada para adaptar os parâmetros do nível inferior. O controlador é implementado usando um kit Arduino DUE e os resultados comprovam melhorias, tanto na resposta de desempenho quanto na perturbação devido à carga no controle de velocidade do motor CC.

Um trabalho desenvolvido por Silva *et al.* (2017), analisa o comportamento da temperatura da interface ferramenta-cavaco durante o processo de torneamento externo de um aço com diâmetro de 50 mm e 200 mm de comprimento. O parâmetro de controle utilizado no estudo para reduzir o desgaste da ferramenta foi a velocidade de corte, a partir da qual a temperatura foi controlada indiretamente. O comportamento da temperatura foi inicialmente modelado pelo método de elementos finitos (MEF) e o resultado da simulação comparado com os resultados experimentais. Utiliza-se no experimento um módulo Arduino UNO para a aquisição de dados e o ajuste de velocidade é feito por meio de um controlador *fuzzy*, a partir de um sistema *loop* fechado, que por consequência, controla a temperatura na interface ferramenta-cavaco e diminui o desgaste da máquina.

Muller (2019) desenvolveu um sistema de recomendação utilizando lógica *fuzzy* embarcada no Arduino MEGA 2560, o qual monitora os batimentos cardíacos de usuários de bicicletas ergométricas e indica a rotação adequada conforme a frequência cardíaca e a carga mensurada. Conectou-se um alternador CA ao volante de inércia da bicicleta ergométrica, gerando assim energia elétrica renovável de acordo com o esforço aplicado ao pedal. O objetivo do sistema desenvolvido é de assegurar que o ciclista gere energia elétrica sustentável, por um determinado período, com segurança a sua saúde.

O trabalho de Szesz *et al.* (2016) desenvolveu um controlador *fuzzy* embarcado em Arduino eficiente para fazer o monitoramento de temperatura e umidade de um processo de aeração, cuja técnica é a mais utilizada para melhorar a secagem e as condições de armazenamento de grãos.

Esses foram apenas alguns exemplos dentre diversos outros trabalhos desenvolvidos nessa área de pesquisa que, provavelmente, ainda irá evoluir e contribuir muito no campo das pesquisas científicas e tecnológicas. Segundo Nogueira e Servedio (2015) uma grande vantagem do Arduino está na sua comunidade de usuários, sendo uma plataforma adequada

como ferramenta didática de ensino tecnológico, pela sua simplicidade e facilidade de obtenção e utilização, mas também por sua versatilidade em agregar projetos de complexidade variada na área industrial.

3 DESENVOLVIMENTO

3.1 A MÁQUINA PREPAC-DERMEC 1000 E O ATUAL PROCESSO DE ENVASE

A busca por melhorias na qualidade dos processos produtivos deve se concentrar em iniciativas que, promovidas continuamente, permitam identificar problemas, priorizar ações corretivas, implantá-las e dar sequência a intervenções que gerem resultados positivos e confiáveis. Nesse contexto, foi identificada a oportunidade de se melhorar o processo de envase da máquina Prepac-Dermec 1000 apresentada na Figura 13, que é propriedade do Departamento de Produção (DPD) do Campus da Unesp de Guaratinguetá.

Figura 13 – Máquina de envase Prepac-Dermec 1000 do DPD/UNESP.



Fonte: Elaborado pelo autor.

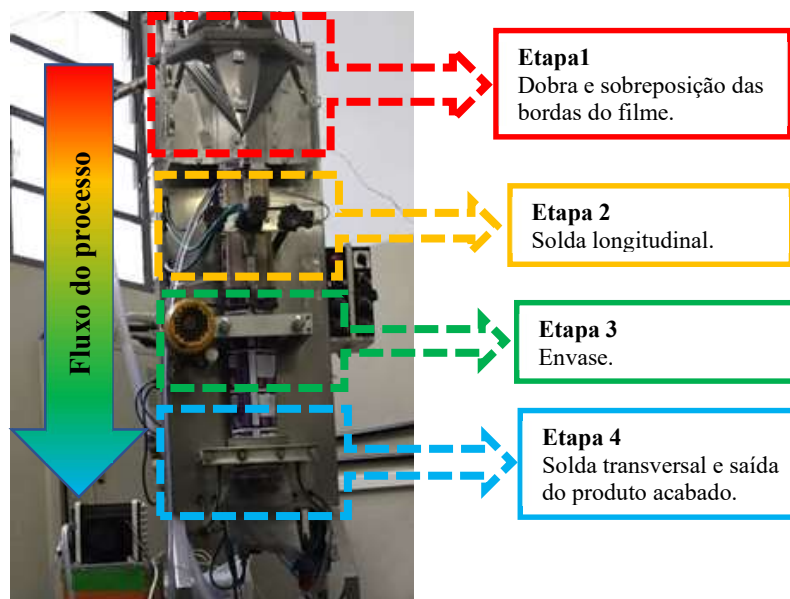
Utilizada em indústrias para realizar envase de líquidos, a máquina Prepac-Dermec 1000, atualmente, tem como função encher embalagens com 200 ml de suco. A máquina é equipada com microcontroladores que coletam dados do sistema de envase, como temperatura das soldas (superior e inferior) da embalagem, tempo de enchimento da embalagem e quantidade de embalagens envasadas. O sistema conta com o auxílio de uma balança de precisão centesimal para a medição dos parâmetros de massa, tanto de embalagem quanto de conteúdo líquido. A massa de referência do líquido envasado é 202,71 g, sendo que 2,71 g é a massa da embalagem. O tempo de referência de abertura da válvula para encher os saquinhos com os 200 ml de suco é de 358 ms. A máquina Prepac-Dermec, objeto de estudo deste trabalho, tem capacidade de operar a taxas de 2.000 sacos/hora.

O processo de envase da máquina foi dividido em quatro etapas. Na Etapa 1, a embalagem

a ser envasada é composta por um filme polimérico na forma de bobina, que se desenrola pela passagem por rolos até formar uma dobra no formato de gravata, conforme ilustrado na Figura 14. Nessa etapa, é imprescindível o correto posicionamento do filme para garantir a qualidade da dobra e respectiva sobreposição das bordas, caso contrário, há possibilidade de interromper o processo produtivo e gerar desperdícios.

Na Etapa 2, solda superior, há a primeira soldagem do filme no sentido longitudinal formando um tipo de “tubo” contínuo. A Etapa 3, é o processo de envase do líquido propriamente dito e objeto de estudo desse trabalho. Já na Etapa 4, solda inferior, há a soldagem transversal do filme e o respectivo corte de separação, sendo obtido dessa maneira o produto (suco de fruta) embalado em unidades de 200 ml. As etapas e o fluxo do processo de envase são ilustradas na Figura 14:

Figura 14 – Etapas do processo de envase.



Fonte: Elaborado pelo autor.

No tópico a seguir, é apresentado o estudo de controle estatístico de processo (CEP) que foi realizado no processo de envase da máquina Prepac-Dermec 1000.

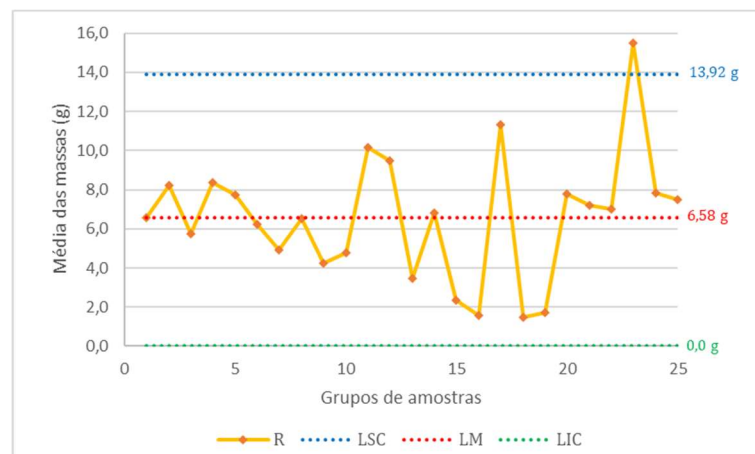
3.2 PARÂMETROS DE CONTROLE ESTATÍSTICO DO PROCESSO DE ENVASE

Os estudos de CEP para o processo de envase da máquina Prepac-Dermec 1000 foram desenvolvidos juntamente com um grupo de pesquisadores da Faculdade de Engenharia de Guaratinguetá – UNESP. Quando se iniciou a fase de ajustes, a máquina foi testada quanto ao

funcionamento básico e estabilidade operacional com total normalidade das funções básicas. Como a variável de estudo era o volume envasado, utilizou-se água como produto pela facilidade de fornecimento direto da rede de distribuição da faculdade, limpeza da máquina e do descarte das amostras envasadas, além de implicar em menor custo se comparado ao suco.

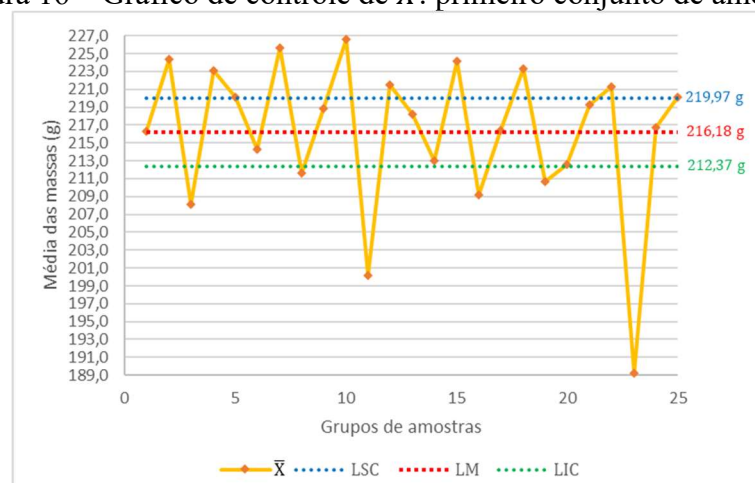
Um primeiro conjunto de dados foi coletado para se fazer uma análise dos parâmetros de CEP, onde se retirou 25 amostras contendo cinco elementos cada, em intervalos de, aproximadamente, cinco minutos. As 125 amostras foram pesadas utilizando a balança centesimal. A partir dos dados coletados, foram calculados os valores da amplitude R (amplitude amostral para monitorar a dispersão) e da média \bar{X} (média amostral) como sendo 6,58 g e 216,18 g, respectivamente (CRUZ, 2019). Com os resultados dessa amostragem, foram gerados os gráficos de controle de R e \bar{X} , apresentados nas Figuras 15 e 16:

Figura 15 – Gráfico de controle de R : primeiro conjunto de amostras.



Fonte: Adaptado de Cruz (2019).

Figura 16 – Gráfico de controle de \bar{X} : primeiro conjunto de amostras.

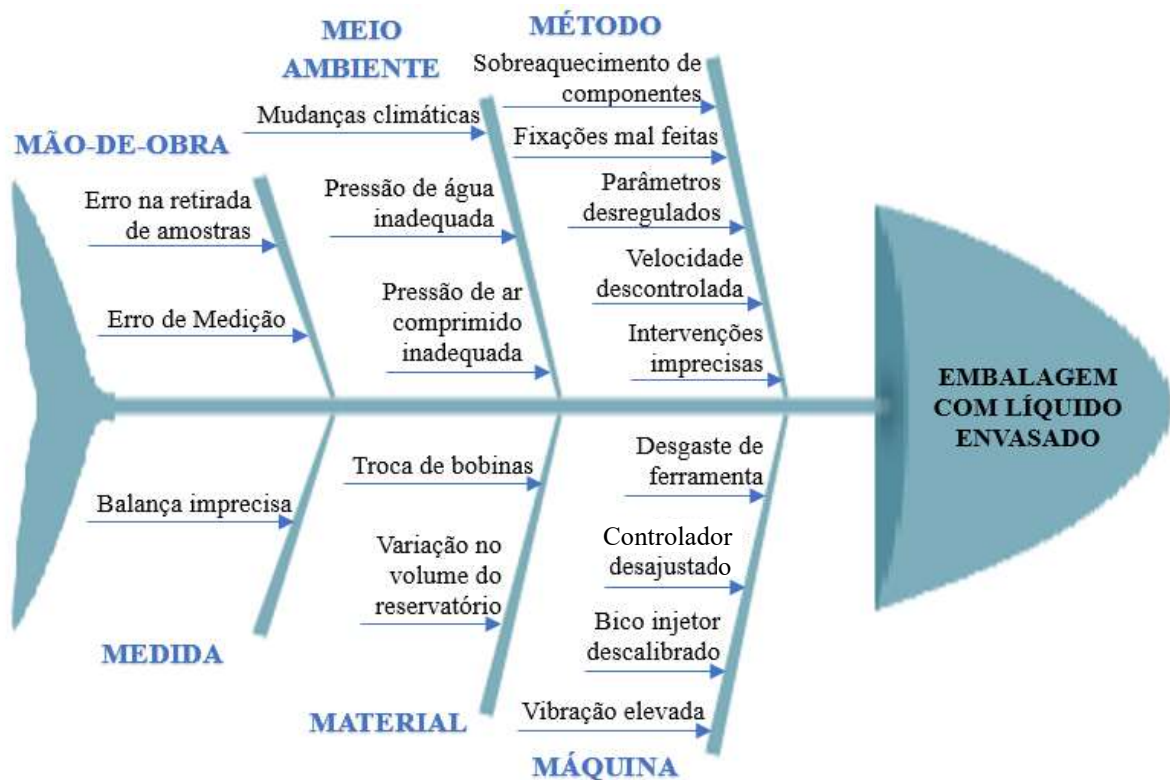


Fonte: Adaptado de Cruz (2019).

Pode-se verificar no gráfico de controle de R , a ocorrência de um ponto acima de LSC. Já no gráfico de controle de \bar{X} , observa-se que as massas de dez amostras estão acima de LSC e seis estão abaixo de LIC. Desta forma, constatou-se que a máquina não estava em controle, de acordo com os critérios apresentados no capítulo 2 deste trabalho e, sendo assim, foi necessário fazer uma revisão dos limites, bem como de todo o processo afim de identificar as possíveis causas desse diagnóstico (CRUZ, 2019).

Para auxiliar na análise das possíveis causas especiais atuantes no processo, foi elaborado o Diagrama de Ishikawa apresentado na Figura 17:

Figura 17 – Diagrama de Ishikawa do processo de envase.



Fonte: Adaptado de Lopes (2018).

A seguir são apresentadas as causas especiais e respectivas intervenções que foram realizadas no processo para melhorar seu ajuste e controle:

- Variação do volume d'água no reservatório de abastecimento: verificou-se uma inconstância entre o volume de entrada e saída de água no reservatório, o que causava o transbordamento de água do reservatório ou falta dela no decorrer do processo, influenciando, inclusive, na pressão. O problema foi reparado aumentando um pouco a entrada de água e instalando uma mangueira de escape sem derivação para extravasar o excesso de água em seu nível superior,

eliminando a variabilidade do nível do reservatório, conforme mostra a Figura 18.

- Tempo de ativação e resfriamento das resistências elétricas responsáveis pelo selo e corte das embalagens: foi feita uma regulagem no tempo de ativação das resistências, visto que ele é diretamente proporcional ao aquecimento que produzem. O calor em excesso derrete totalmente o selo ao invés de fechá-lo e, por outro lado, o calor insuficiente não fecha as embalagens ou não as corta para separá-las. Esse ajuste melhorou a qualidade dos selos das embalagens. Instalou-se o sistema de resfriamento em malha fechada independente, apresentado na Figura 19, para evitar o superaquecimento das resistências, principalmente quando a máquina opera por tempo prolongado.

Figura 18 – Mangueira de escape do reservatório.



Fonte: Lopes (2018).

Figura 19 – Sistema de resfriamento dos selos.



Fonte: Lopes (2018).

Após terem sido implementadas essas correções no processo de envase, foram coletados um segundo conjunto de amostras, nos mesmos padrões da primeira amostragem, para verificar se houve melhoria no controle da máquina. A Tabela 1 apresenta as massas das amostras coletadas, bem como a amplitude e a média de cada grupo:

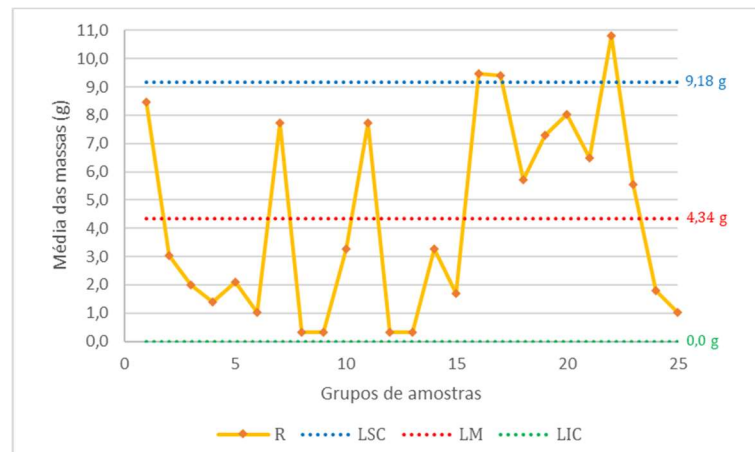
Tabela 1 – Massas de líquido envasado: segundo conjunto de amostras.

Grupos (m = 25)	Amostras (n = 5)					R (g)	\bar{X} (g)
1	205,34	196,89	204,42	202,64	199,76	8,45	201,81
2	202,68	201,39	202,18	202,02	199,63	3,05	201,58
3	201,22	202,50	201,80	202,56	203,21	1,99	202,26
4	202,67	202,46	202,00	201,70	201,28	1,39	202,02
5	202,34	202,82	203,27	201,18	201,37	2,09	202,20
6	202,14	202,83	201,79	202,03	202,06	1,04	202,17
7	202,27	203,47	201,49	207,55	199,83	7,72	202,92
8	200,76	200,51	200,84	200,61	200,72	0,33	200,69
9	200,32	200,50	200,63	200,40	200,44	0,31	200,46
10	202,14	204,43	201,17	201,76	202,36	3,26	202,37
11	202,27	203,47	201,49	207,55	199,83	7,72	202,92
12	200,76	200,51	200,84	200,61	200,72	0,33	200,69
13	200,32	200,50	200,63	200,40	200,44	0,31	200,46
14	202,14	204,43	201,17	201,76	202,36	3,26	202,37
15	202,65	203,93	202,23	203,90	203,64	1,70	203,27
16	203,69	194,35	203,81	203,31	203,63	9,46	201,76
17	202,32	200,75	210,15	204,16	204,44	9,40	204,36
18	204,10	202,38	205,28	204,14	208,09	5,71	204,80
19	201,94	202,62	201,95	209,23	209,24	7,30	205,00
20	201,37	201,89	202,62	209,39	203,13	8,02	203,68
21	202,30	203,41	204,77	198,29	201,96	6,48	202,15
22	195,24	202,87	206,06	204,89	203,78	10,82	202,57
23	204,29	205,03	204,24	199,48	200,05	5,55	202,62
24	202,94	204,52	202,71	203,44	202,79	1,81	203,28
25	201,28	201,40	201,27	201,44	202,29	1,02	201,54

Fonte: Elaborado pelo autor.

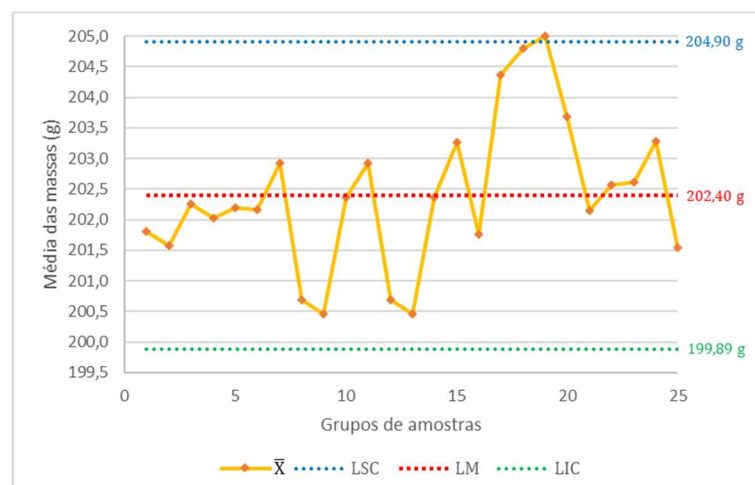
Foram gerados os gráficos de controle de R e \bar{X} para esse segundo conjunto de amostras e estão apresentados nas Figuras 20 e 21, respectivamente:

Figura 20 – Gráfico de controle de R : segundo conjunto de amostras.



Fonte: Elaborado pelo autor.

Figura 21 – Gráfico de controle de \bar{X} : segundo conjunto de amostras.



Fonte: Elaborado pelo autor.

Para este segundo conjunto de amostras, a média de R e de \bar{X} calculadas são, respectivamente, 4,34 g e 202,40 g. De acordo com Lopes (2018), o limite superior e inferior calculados para o gráfico de amplitude são, respectivamente, 9,18 g e 0 g, enquanto os limites superior e inferior para o gráfico de média são, respectivamente, 204,90 g e 199,89 g.

Analisando os gráficos de controle apresentados, verifica-se que houve uma melhora no controle do processo após terem sido feitas as intervenções nas causas especiais mencionadas anteriormente, obtendo-se uma redução de, aproximadamente, 72% nas ocorrências de pontos

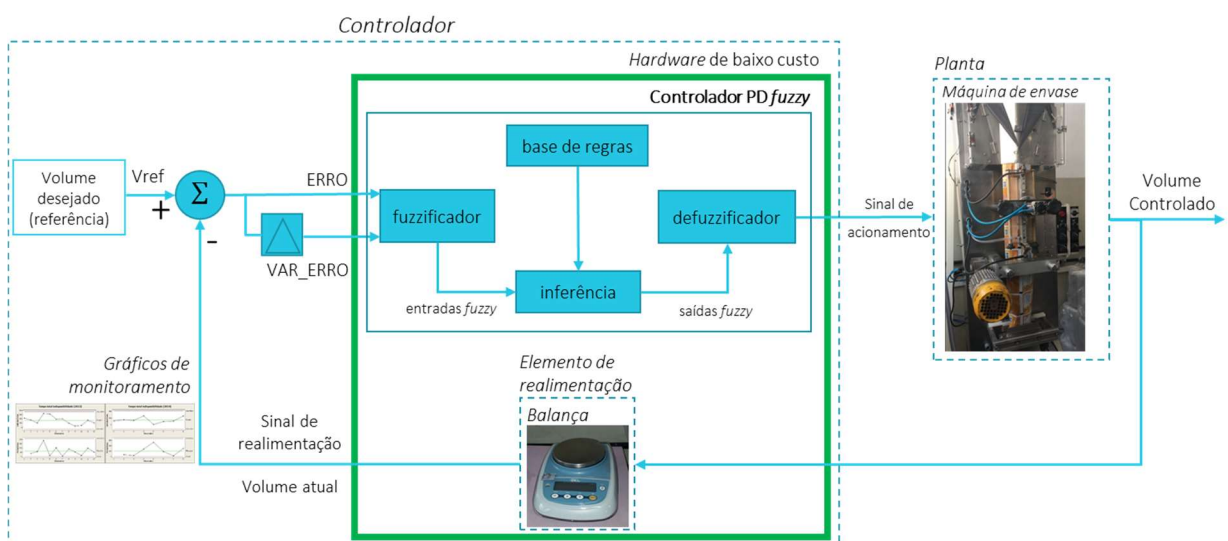
fora dos limites. Pode-se observar que no gráfico de controle da amplitude existem três pontos acima de LSC e no gráfico da média, um ponto acima de LSC, que poderiam indicar ainda a existência de causas especiais atuando no processo. Entretanto, de acordo com a pesquisa de Cruz (2019), considerando os testes de normalidade e as possibilidades de alarmes falsos para esse sistema, esse equipamento está em relativo controle e tem condições plenas de operar.

Neste contexto, com o propósito de contribuir ainda mais para a melhoria da estabilidade desse processo, propõe-se uma intervenção na máquina, mais precisamente no sistema de controle da válvula de envase, que consiste em substituir o atual modelo por um sistema em malha fechada, que utiliza um controlador *fuzzy* embarcado em Arduino. Espera-se que essa modificação seja capaz de reduzir perdas por unidades não conformes, custos com mão de obra e manutenção, além de melhorar o monitoramento do processo por meio da geração de gráfico em tempo real das massas medidas pela balança.

3.3 CONTROLADOR FUZZY EMBARCADO EM ARDUINO

O sistema de inferência *fuzzy* (FIS) foi elaborado para realizar o controle do tempo de abertura da válvula responsável por encher o saquinho. O tempo de abertura é diretamente proporcional ao volume envasado e, sendo assim, o volume abaixo do valor de referência é compensado por um tempo de abertura maior e vice-versa. A Figura 22 apresenta o diagrama esquemático do sistema de monitoramento e controle do processo de envase da máquina em estudo:

Figura 22 – Diagrama esquemático do monitoramento e controle do processo de envase.



Fonte: Elaborado pelo autor.

3.3.1 Sistema de inferência fuzzy do controlador em Matlab

Como pode ser visto na Figura 22, o controlador possui duas variáveis de entrada: Erro (ERRO) e Variação do Erro (VAR_ERRO), e uma de saída: Variação do Tempo de Abertura da Válvula (VTAV). O método de inferência utilizado é o Mamdani, também conhecido como MAX e MIN e, para a defuzzificação, o Método do Centroide. A definição dos universos de discurso das funções de pertinência, que podem ser vistos nas Figuras 23, 24 e 25, levou em consideração o limite volumétrico do saquinho, os resultados das amostras coletadas e, principalmente, o resultado da simulação em Arduino. Nas primeiras simulações, verificou-se que o reajuste desses parâmetros resultaria em um controle mais eficiente. A seguir são apresentadas as variáveis, com suas respectivas descrições, que foram utilizadas no FIS:

- **Variável de entrada “Erro (ERRO)”**: trata-se da diferença entre o valor da massa do saquinho envasado e o valor da massa de referência de 202,71 g, sendo:

MF 1 = “ENA”: Erro Negativo Alto (Massa muito abaixo do valor de referência).

MF 2 = “ENM”: Erro Negativo Médio (Massa abaixo do valor de referência).

MF 3 = “ENB”: Erro Negativo Baixo (Massa pouco abaixo do valor de referência).

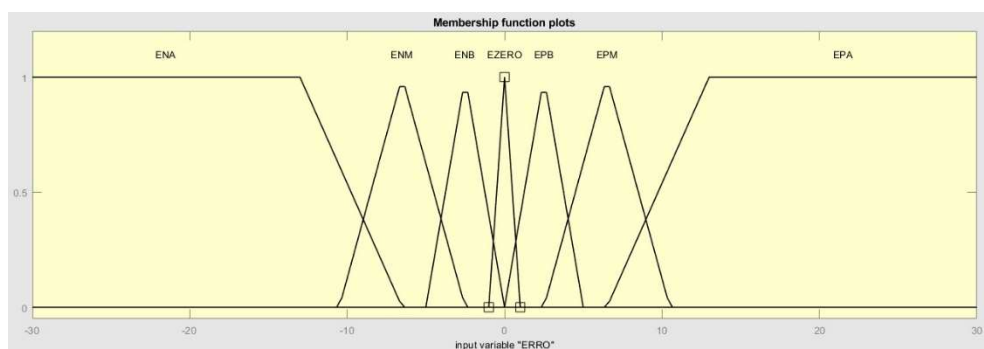
MF 4 = “EZERO”: Erro Zero (Massa igual ao valor de referência).

MF 5 = “EPB”: Erro Positivo Baixo (Massa pouco acima do valor de referência).

MF 6 = “EPM”: Erro Positivo Médio (Massa acima do valor de referência).

MF 7 = “EPA”: Erro Positivo Alto (Massa muito acima do valor de referência).

Figura 23 – Função de Pertinência de Entrada: ERRO.

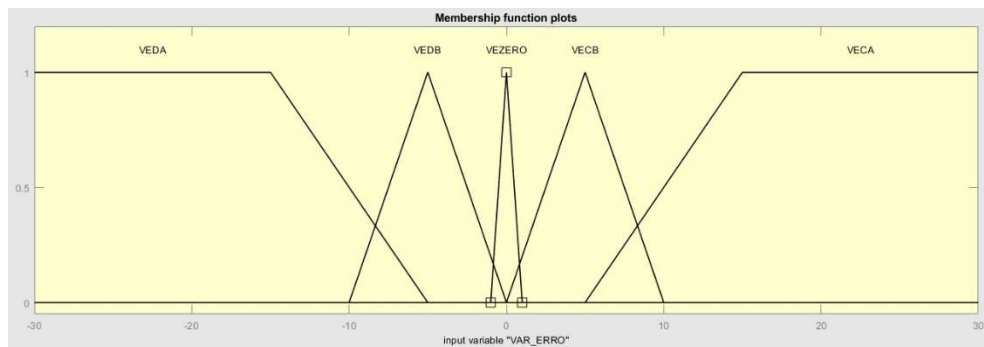


Fonte: Elaborado pelo autor.

- **Variável de entrada “Variação do Erro (VAR_ERRO)”**: trata-se da variável de entrada referente à variação no erro, ou seja, trata-se da diferença entre o erro antes do ajuste e após o ajuste, sendo:

- MF 1 = “VEDA”: Variação do Erro Decrescente Alta (O erro diminuiu muito após o ajuste).
- MF 2 = “VEDB”: Variação do Erro Decrescente Baixa (O erro diminuiu pouco após o ajuste).
- MF 3 = “VEZERO”: Variação do Erro Zero (O erro não variou após o ajuste).
- MF 4 = “VECB”: Variação do Erro Crescente Baixa (O erro aumentou pouco após o ajuste).
- MF 5 = “VECA”: Variação do Erro Crescente Alta (O erro aumentou muito após o ajuste).

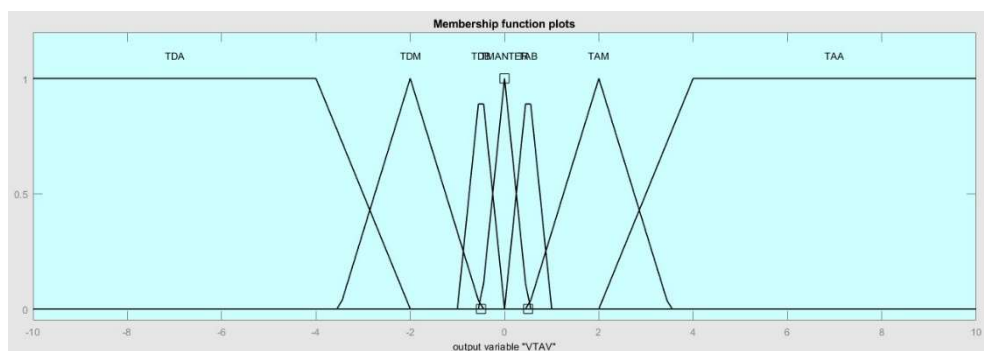
Figura 24 – Função de Pertinência de Entrada: VAR_ERRO.



Fonte: Elaborado pelo autor.

- **Variável de saída “Variação do Tempo de Abertura da Válvula (VTAV)”**: Variação do tempo de abertura da válvula, sendo:
 - MF 1 = “TAA”: Aumento de Tempo Alto (Aumenta muito o tempo de abertura da válvula).
 - MF 2 = “TAM”: Aumento de Tempo Médio (Aumenta o tempo de abertura da válvula).
 - MF 3 = “TAB”: Aumento de Tempo Baixo (Aumenta pouco o tempo de abertura da válvula).
 - MF 4 = “TMANTER”: Mantém o tempo (Mantém o tempo de abertura ajustado).
 - MF 5 = “TDB”: Diminuição de Tempo Baixa (reduz pouco o tempo de abertura da válvula).
 - MF 6 = “TDM”: Diminuição de Tempo Média (reduz o tempo de abertura da válvula).
 - MF 7 = “TDA”: Diminuição de Tempo Alta (reduz muito o tempo de abertura da válvula).

Figura 25 – Função de Pertinência de Saída: VTAV.



Fonte: Elaborado pelo autor.

A Figura 26 apresenta o conjunto de regras criado a partir das funções de pertinência:

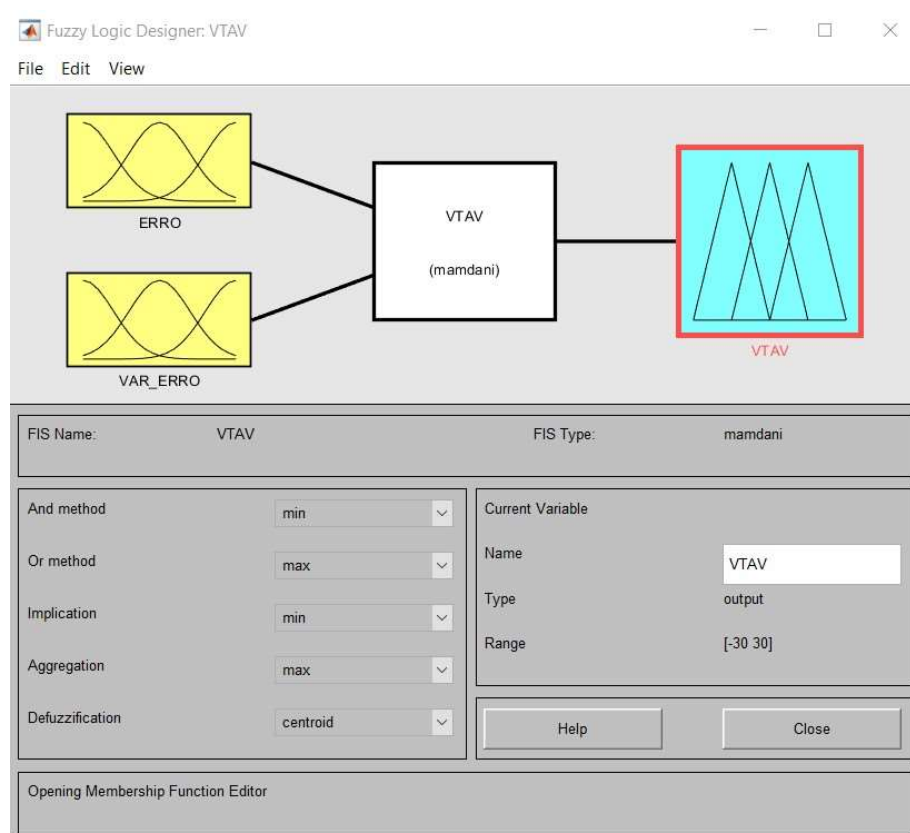
Figura 26 – Conjunto de regras¹.

		ERRO						
VARERRO		ENA	ENM	ENB	EZERO	EPB	EPM	EPA
VEDA		TAA	TAA	TAB	TAB	TMANTER	TMANTER	TDB
VEDB		TAA	TAM	TAB	TAB	TMANTER	TDB	TDM
VEZERO		TAA	TAB	TAB	TMANTER	TDB	TDB	TDA
VECB		TAM	TAB	TMANTER	TDB	TDB	TDM	TDA
VECA		TAB	TMANTER	TMANTER	TDB	TDB	TDA	TDA

Fonte: Elaborado pelo autor.

A Figura 27 apresenta o diagrama esquemático do controlador *fuzzy* no Matlab:

Figura 27 – Diagrama do controlador *fuzzy* no Matlab.



Fonte: Elaborado pelo autor.

¹ Descrição de algumas siglas utilizadas no conjunto de regras: ENA - Erro Negativo Alto; VEDA - Variação do Erro Decrescente Alta; TAA - Aumento de Tempo Alto (abertura da válvula).

O Quadro 3 apresenta o sistema de inferência *fuzzy* do controlador:

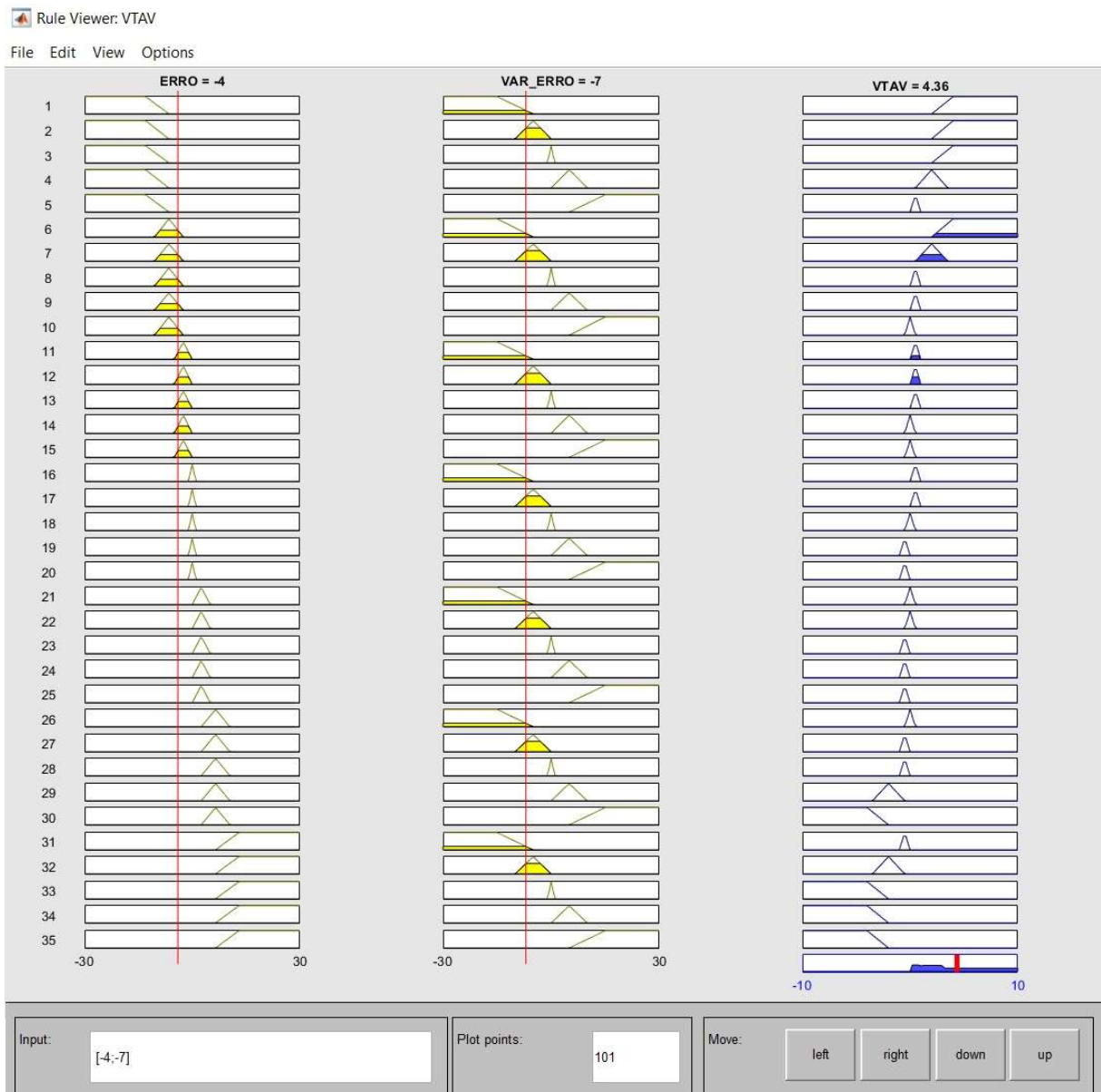
Quadro 3 – Sistema de inferência *fuzzy* do controlador.

REGRA	ENTRADA		SAÍDA
	ERRO	VARERRO	VTAV
Rule 01	SE o ERRO é ENA	E a VARERRO é VEDA	ENTÃO o VTAV é TAA
Rule 02	SE o ERRO é ENA	E a VARERRO é VEDB	ENTÃO o VTAV é TAA
Rule 03	SE o ERRO é ENA	E a VARERRO é VEZERO	ENTÃO o VTAV é TAA
Rule 04	SE o ERRO é ENA	E a VARERRO é VECB	ENTÃO o VTAV é TAM
Rule 05	SE o ERRO é ENA	E a VARERRO é VECA	ENTÃO o VTAV é TAB
Rule 06	SE o ERRO é ENM	E a VARERRO é VEDA	ENTÃO o VTAV é TAA
Rule 07	SE o ERRO é ENM	E a VARERRO é VEDB	ENTÃO o VTAV é TAM
Rule 08	SE o ERRO é ENM	E a VARERRO é VEZERO	ENTÃO o VTAV é TAB
Rule 09	SE o ERRO é ENM	E a VARERRO é VECB	ENTÃO o VTAV é TAB
Rule 10	SE o ERRO é ENM	E a VARERRO é VECA	ENTÃO o VTAV é TMANTER
Rule 11	SE o ERRO é ENB	E a VARERRO é VEDA	ENTÃO o VTAV é TAB
Rule 12	SE o ERRO é ENB	E a VARERRO é VEDB	ENTÃO o VTAV é TAB
Rule 13	SE o ERRO é ENB	E a VARERRO é VEZERO	ENTÃO o VTAV é TAB
Rule 14	SE o ERRO é ENB	E a VARERRO é VECB	ENTÃO o VTAV é TMANTER
Rule 15	SE o ERRO é ENB	E a VARERRO é VECA	ENTÃO o VTAV é TMANTER
Rule 16	SE o ERRO é EZERO	E a VARERRO é VEDA	ENTÃO o VTAV é TAB
Rule 17	SE o ERRO é EZERO	E a VARERRO é VEDB	ENTÃO o VTAV é TAB
Rule 18	SE o ERRO é EZERO	E a VARERRO é VEZERO	ENTÃO o VTAV é TMANTER
Rule 19	SE o ERRO é EZERO	E a VARERRO é VECB	ENTÃO o VTAV é TDB
Rule 20	SE o ERRO é EZERO	E a VARERRO é VECA	ENTÃO o VTAV é TDB
Rule 21	SE o ERRO é EPB	E a VARERRO é VEDA	ENTÃO o VTAV é TMANTER
Rule 22	SE o ERRO é EPB	E a VARERRO é VEDB	ENTÃO o VTAV é TMANTER
Rule 23	SE o ERRO é EPB	E a VARERRO é VEZERO	ENTÃO o VTAV é TDB
Rule 24	SE o ERRO é EPB	E a VARERRO é VECB	ENTÃO o VTAV é TDB
Rule 25	SE o ERRO é EPB	E a VARERRO é VECA	ENTÃO o VTAV é TDB
Rule 26	SE o ERRO é EPM	E a VARERRO é VEDA	ENTÃO o VTAV é TMANTER
Rule 27	SE o ERRO é EPM	E a VARERRO é VEDB	ENTÃO o VTAV é TDB
Rule 28	SE o ERRO é EPM	E a VARERRO é VEZERO	ENTÃO o VTAV é TDB
Rule 29	SE o ERRO é EPM	E a VARERRO é VECB	ENTÃO o VTAV é TDM
Rule 30	SE o ERRO é EPM	E a VARERRO é VECA	ENTÃO o VTAV é TDA
Rule 31	SE o ERRO é EPA	E a VARERRO é VEDA	ENTÃO o VTAV é TDB
Rule 32	SE o ERRO é EPA	E a VARERRO é VEDB	ENTÃO o VTAV é TDM
Rule 33	SE o ERRO é EPA	E a VARERRO é VEZERO	ENTÃO o VTAV é TDA
Rule 34	SE o ERRO é EPA	E a VARERRO é VECB	ENTÃO o VTAV é TDA
Rule 35	SE o ERRO é EPA	E a VARERRO é VECA	ENTÃO o VTAV é TDA

Fonte: Elaborado pelo autor.

A Figura 28 apresenta o conjunto de regras com as respectivas funções de pertinência no Matlab, sendo possível visualizar a saída VTAV a partir dos valores atribuídos à entrada ERRO e VAR_ERRO.

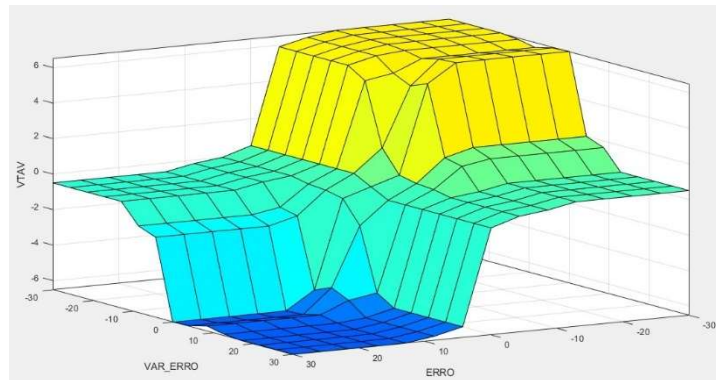
Figura 28 – Visualização da saída VTAV a partir do conjunto de regras.



Fonte: Elaborado pelo autor.

Após atribuída e testada a base de regras, foi gerada a superfície de resposta no Matlab conforme apresentada na Figura 29, a qual apresenta-se sem distorções abruptas ou grandes picos de valores.

Figura 29 – Superfície do resultado do modelo.



Fonte: Elaborado pelo autor.

Feita a validação do sistema de inferência *fuzzy* utilizando o Matlab, o próximo tópico apresenta como o controlador foi embarcado no Arduino Mega 2560.

3.3.2 Código do controlador fuzzy embarcado no Arduino Mega 2560

O Arduino Mega 2560 foi escolhido para embarcar o controlador *fuzzy*, por suas especificações serem compatíveis com o código desenvolvido. Para embarcar o controlador no Arduino, tomou-se como base os parâmetros elaborados, primeiramente, no Matlab. Utilizando a IDE do Arduino, desenvolvida na linguagem C++, em conjunto com a biblioteca *eFLL*, desenvolveu-se o código responsável por embarcar o controlador *fuzzy* no Arduino, o qual é apresentado integralmente no Apêndice A, e cujos passos construtivos são explicados a seguir:

- **Passo 1:** declaração das variáveis dos sistema:

```
boolean entra = true;
```

```
float output; // saída do controlador fuzzy (VTAV).
```

```
float Vref=202.71; // aqui é definido o peso de referência do saquinho.
```

```
float peso; // indicar o peso inicial?
```

```
float anterior=0;
```

```
float atual=0;
```

```
float entrada1=0;
```

```
float entrada2=0;
```

```
float precisao=0.01;
```

```
int iteracao=1; //contador de iterações
```

- **Passo 2:** construção da entrada do controlador, ENTRADA1 que refere-se a variável ERRO, a qual é composta por sete variáveis linguísticas, sendo apresentado como exemplo apenas o código para a variável ENA (as demais entradas são semelhantes):

```
// Variável de entrada "Erro (ERRO)"
FuzzyInput *ERRO = new FuzzyInput(1);
// MF 1 = "ENA": Erro Negativo Alto (Massa muito abaixo do valor de referência).
FuzzySet *ENA = new FuzzySet(-30, -30, -13, -6);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(ENA);
```

- **Passo 3:** construção da entrada do controlador, ENTRADA2 que refere-se a variável VAR_ERRO, a qual é composta por cinco variáveis linguísticas, exemplificada apenas pelo código para a variável VEDA (as demais entradas são semelhantes):

```
// Variável de entrada "Variação do Erro (VAR_ERRO)"
FuzzyInput *VARERRO = new FuzzyInput(2);
// MF 1 = "VEDA": Variação do Erro Decrescente Alta (O erro diminuiu muito após o ajuste).
FuzzySet *VEDA = new FuzzySet(-30, -30, -15, -5);
// Incluindo o FuzzySet no FuzzyInput
VARERRO->addFuzzySet(VEDA);
```

- **Passo 4:** construção do código para a saída do sistema, que refere-se a variação do tempo de abertura da válvula (VTAV) pelo objeto FuzzyOutput:

```
// Variável de saída "Variação do Tempo de Abertura da Válvula (VTAV)"
FuzzyOutput *VTAV = new FuzzyOutput(1);
// MF 1 = "TAA": Aumento de Tempo Alto (Aumenta muito o tempo de abertura da válvula).
FuzzySet *TDA = new FuzzySet(-10, -10, -4, -2);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet(TDA);
```

- **Passo 5:** construção do conjunto de regras apresentado no Quadro 3. Ao todo, são 35 regras diferentes, entretanto para simplificar a demonstração, apresenta-se apenas o trecho da linha do código referente a regra "Rule 01":

```
// Rule 01: SE o ERRO é ENA E a VARERRO é VEDA ENTÃO o VTAV é TAA
FuzzyRuleAntecedent *ENA_VEDA = new FuzzyRuleAntecedent();
```

```

ENA_VEDA->joinWithAND(ENA, VEDA);
FuzzyRuleConsequent* thenVTAV1 = new FuzzyRuleConsequent();
thenVTAV1->addOutput(TAA);
FuzzyRule* fuzzyRule1 = new FuzzyRule(1, ENA_VEDA, thenVTAV1);
fuzzy->addFuzzyRule(fuzzyRule1);

```

- **Passo 6:** código do sistema de controle *fuzzy* em malha fechada, responsável por fazer a aquisição da leitura da balança e manter a massa do saquinho no valor de referência:

```

//*****
void loop() {
//PARA FACILITAR FUTURAS CARGAS PINO DE PAUSE
if (!digitalRead(pinPause)) {
    estadoPause();
    return;
}
medida = scale.get_units(5); // ativa a balança
entra=true;
iteracao=1;
if (medida > pesoMin){
    Serial.println("Balança acionada. Ativando o controlador...");
    delay(2000);
    medida = scale.get_units(10); //aquisição de novas medidas (estabilidade da balança)
    digitalWrite(led_sistema,LOW);
    delay(100);
    digitalWrite(led_sistema,HIGH);
    delay(100);
    digitalWrite(led_sistema,LOW);
    delay(100);
    digitalWrite(led_sistema,HIGH);
    delay(100);
    Serial.println("");
}
Serial.println("*****");
//IMPRIME CONTAGEM DA MEDIDA
Serial.print("Medida ");

```

```

Serial.print(counter);
counter += 1;
delay(100);
Serial.print(": ");
//IMPRIME PESO EM GRAMAS
String medida_string; //serve para corrigir o ponto em vírgula
medida_string = String(medida*1000,2);
Serial.print(medida_string);
Serial.println(" g");
Serial.println("*****");
delay(2000);
if (entra) {
  peso = medida*1000;
  entrada1 = peso - Vref;
  anterior = entrada1;
  atual = entrada1;
  entrada2 = atual-anterior;
  entra = false;
}
//Loop de iterações enquanto o ERRO é maior ou menor que o intervalo
"precisao".
while ((entrada1 <-precisao)|| (entrada1>precisao)){
  fuzzy->setInput(1,entrada1);
  fuzzy->setInput(2,entrada2);
  //Imprime os dados de entrada
  Serial.print("Iteracao ");
  Serial.print(iteracao);
  Serial.print("-> ");
  Serial.print("Massa medida: ");
  Serial.println(peso);
  Serial.print("Erro: ");
  Serial.println(entrada1);
  Serial.print("Var_Erro: ");
  Serial.println(entrada2);
}

```



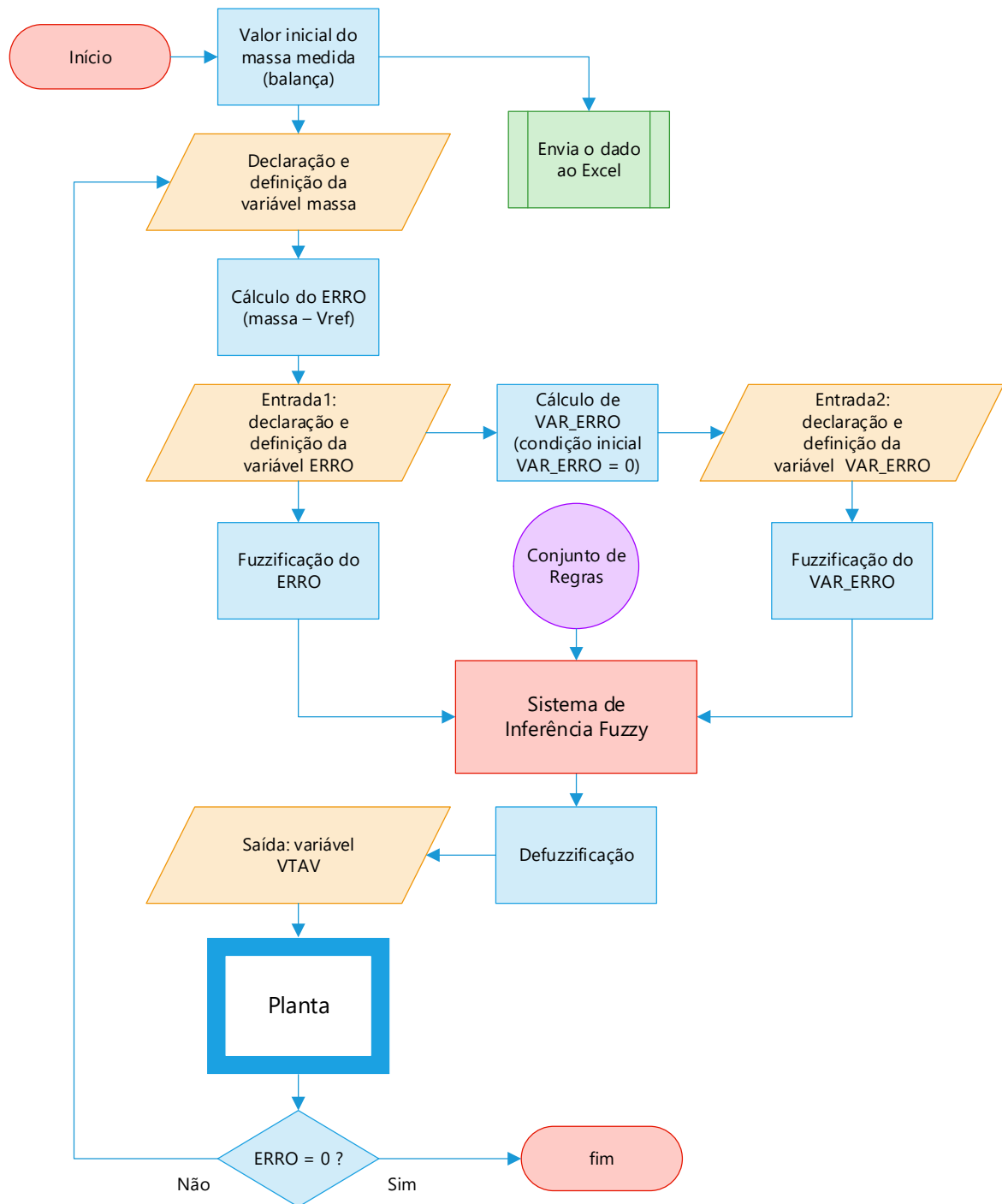
```

delay(200);           // wait for a second
}

```

A Figura 30 apresenta o fluxograma do código apresentado no passo 6, como forma de contribuir para o melhor entendimento do sistema que está sendo proposto nesse trabalho.

Figura 30 – Fluxograma do código do controlador *fuzzy*.



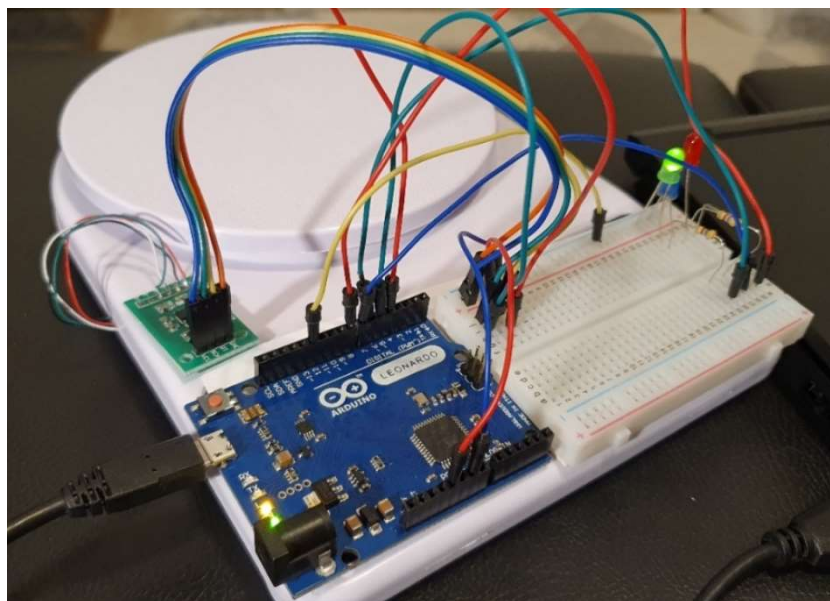
Fonte: Elaborado pelo autor.

3.3.3 Conexão da balança de medição e do Microsoft Excel ao Arduino Leonardo

Em qualquer processo produtivo, sabe-se que um sistema de monitoramento é essencial para que se possa garantir a qualidade do produto que será disponibilizado ao consumidor. Neste contexto, propõe-se um sistema para realizar o monitoramento do processo de envase, a partir de um gráfico gerado em tempo real pelo Microsoft Excel das massas medidas pela balança, cuja interface é feita a partir de um Arduino Leonardo, escolhido pela sua funcionalidade e praticidade de comunicação com o *software* Microsoft Excel.

Na Figura 31 tem-se a ilustração do protótipo construído para simular o sistema de monitoramento das massas medidas no processo de envase:

Figura 31 – Protótipo do sistema de monitoramento.



Fonte: Elaborado pelo autor.

O protótipo desenvolvido conta com o auxílio de quatro leds para identificar algumas situações, conforme descrito a seguir:

- Led vermelho: pisca para indicar que o sistema de medição está pausado;
- Led verde: pisca enquanto estiver iniciando o sistema, que inclui a calibração da tara da balança, e para de piscar, permanecendo aceso, quando a balança está sem carga e apta a realizar uma medição;
- Led azul: acende para indicar que o sensor da balança foi ativado pela existência de uma carga sobre ela e, se o sistema não estiver pausado, a massa medida é enviada

ao controlador e para o Microsoft Excel.

O *start* para a aquisição do dado da massa medida ocorre após a conclusão do processo de inicialização do sistema e calibração da balança e a partir de uma lógica que identifica se o peso que está sendo medido pela balança é diferente de zero, ou seja, assim que um objeto é colocado sobre a balança, a variável “*medida*” recebe o valor em gramas da massa que está sendo medida.

A seguir, apresenta-se o trecho do código embarcado no Arduino Leonardo (o código completo encontra-se no Apêndice F) responsável pela aquisição do valor da massa medida pela balança:

```
// OBJETO PARA MÓDULO DA BALANÇA
#include <HX711.h>
// DEFINIÇÕES DE PINOS
#define pinDT 4 //HX711
#define pinSCK 5 //HX711
// DEFINIÇÕES DA BALANÇA
#define pesoMin 0.05
#define pesoMax 300.0
#define escala 343000.0f
HX711 scale; //Instanciando o objeto balança
// DECLARAÇÃO DE VARIÁVEIS
float medida=0; //balança
//INICIO DA BALANÇA
scale.begin(pinDT, pinSCK); // CONFIGURANDO OS PINOS DA BALANÇA
scale.set_scale(escala); // ENVIANDO O VALOR DA ESCALA CALIBRADO
medida = scale.get_units(5); // ativa a balança
if (medida > pesoMin){
medida = scale.get_units(10);//realiza 10 medidas
}
```

A parte do código do Apêndice F que realiza o envio dos dados de massas medidas para o Microsoft Excel, é apresentado abaixo:

```
//OBJETO PARA EMULADOR DE TECLADO
#include <Keyboard.h>
//DECLARACAO DAS FUNCOES P/ ENVIAR AO EXCEL
void estadoPause();
```

```

void erroCritico();
void teclaEnter();
void teclaTAB();
//INICIO DO EMULADOR DE TECLADO (EXCEL)
    Keyboard.begin();
//IMPRIME CONTAGEM DA MEDIDA
    Keyboard.print("Medida ");
    Keyboard.print(counter);
    counter += 1;
    delay(100);
    teclaTAB();
//IMPRIME MASSA EM GRAMAS
    String medida_string; //serve para corrigir o ponto em vírgula
    medida_string = String(medida*1000,2);
    medida_string.replace(".",",");
    Keyboard.print(medida_string); // Envia a medida em gramas para o Excel
    delay(2000);
    teclaTAB();
    delay(100);
    teclaEnter();
    delay(1000); }
}
//3.FUNCAO PARA ENVIAR O APERTO DE UM ENTER
void teclaEnter() {
    Keyboard.press(KEY_RETURN); // PRESSIONA O ENTER
    delay(50); // ESPERA 50 MILISSEGUNDOS
    Keyboard.release(KEY_RETURN); // SOLTA O ENTER
}
//4.FUNCAO PARA ENVIAR O APERTO DE UM TAB
void teclaTAB() {
    Keyboard.press(KEY_TAB); // PRESSIONA O TAB
    delay(50); // ESPERA 50 MILISSEGUNDOS
    Keyboard.release(KEY_TAB); // SOLTA O TAB
}

```

O capítulo 4 apresenta os resultados obtidos com a simulação do código do controlador PD *fuzzy* em Arduino, bem como o gráfico gerado em Microsoft Excel a partir dos dados de leitura da balança.

4 ANÁLISE DE RESULTADOS

4.1 CONTROLADOR PD FUZZY

Para validar o controlador *fuzzy* que foi desenvolvido neste trabalho e embarcado no Arduino Mega 2560, foram realizadas diversas simulações e o resultado de quatro delas estão apresentados integralmente nos Apêndices B, C, D e E, sendo as entradas de massa medida, respectivamente, 192 g, 198 g, 203 g e 212 g.

Pode-se verificar no código apresentado no Apêndice A que foi utilizada uma precisão de erro de 0,01 para as iterações na busca pelo valor de referência de 202,71 g. Com essa precisão, o controlador permite uma variação na casa centesimal do valor de massa ajustado.

A Tabela 2 apresenta os seguintes parâmetros das simulações: massa medida pela balança (inicial), massa ajustada pelo controlador (final), número de iterações realizadas, tempo de ajuste:

Tabela 2 – Parâmetros das simulações do controlador *fuzzy* embarcado no Arduino.

SIMULAÇÃO	1	2	3	4
Massa medida (g)	193	198	203	212
Massa ajustada (g)	202,74	202,73	202,71	202,74
Número de iterações	34	29	9	36
Tempo de ajuste (ms)	3810	3355	1651	3957

Fonte: Elaborado pelo autor.

Se a precisão for modificada para 0,001, as iterações terminam apenas quando a massa de saída se iguala exatamente ao valor de referência de 202,71 g. Por exemplo, para a massa medido de 203 g com a precisão de 0,001, o tempo de ajuste passou a ser de 6222 ms, o que corresponde a um aumento de quase 277% em relação ao valor para a precisão de 0,01. Analisando o gráfico de controle, pode-se verificar que a variação na casa centesimal não prejudica o controle da máquina e ainda permite que o ajuste ocorra em menos tempo.

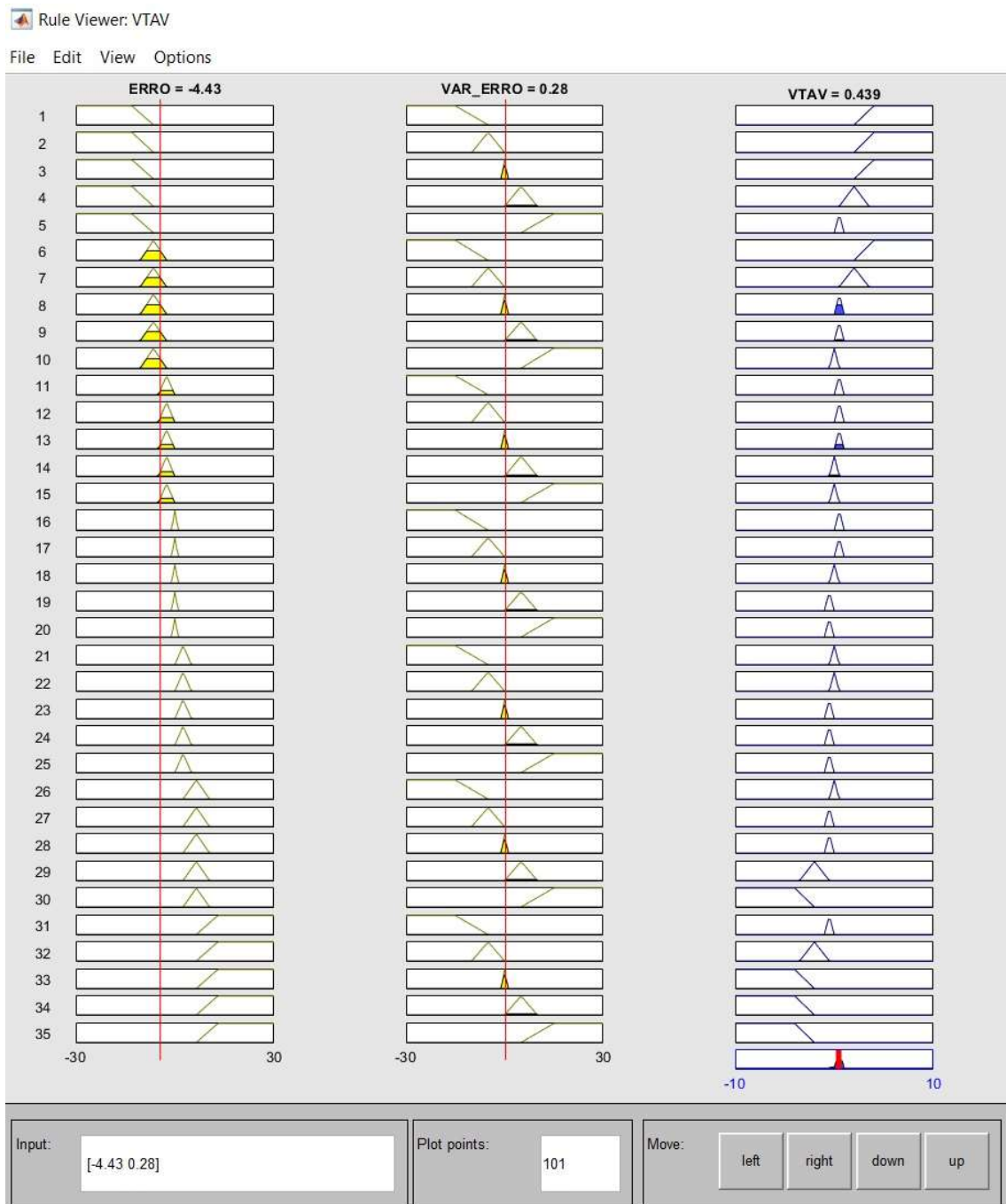
Para comparar os resultados obtidos com o controlador *fuzzy* no Matlab e no Arduino Mega 2560, implementou-se no Matlab os parâmetros da segunda iteração da simulação 2 e da terceira iteração da simulação 4 realizadas no Arduino. Os resultados estão apresentados nas Figuras 32 e 34 para o Arduino Mega 2560 e nas Figuras 33 e 35 para o Matlab, respectivamente:

Figura 32 – Segunda iteração da simulação 2 (Apêndice C): Arduino Mega 2560.

15:30:50,579 -> Massa medida: 198,28
 15:30:50,633 -> Erro: -4,43
 15:30:50,633 -> Var Erro: 0,28
 15:30:50,633 -> VTAV: 0,44

Fonte: Elaborado pelo autor.

Figura 33 – Segunda iteração da simulação 2: Matlab.



Fonte: Elaborado pelo autor.

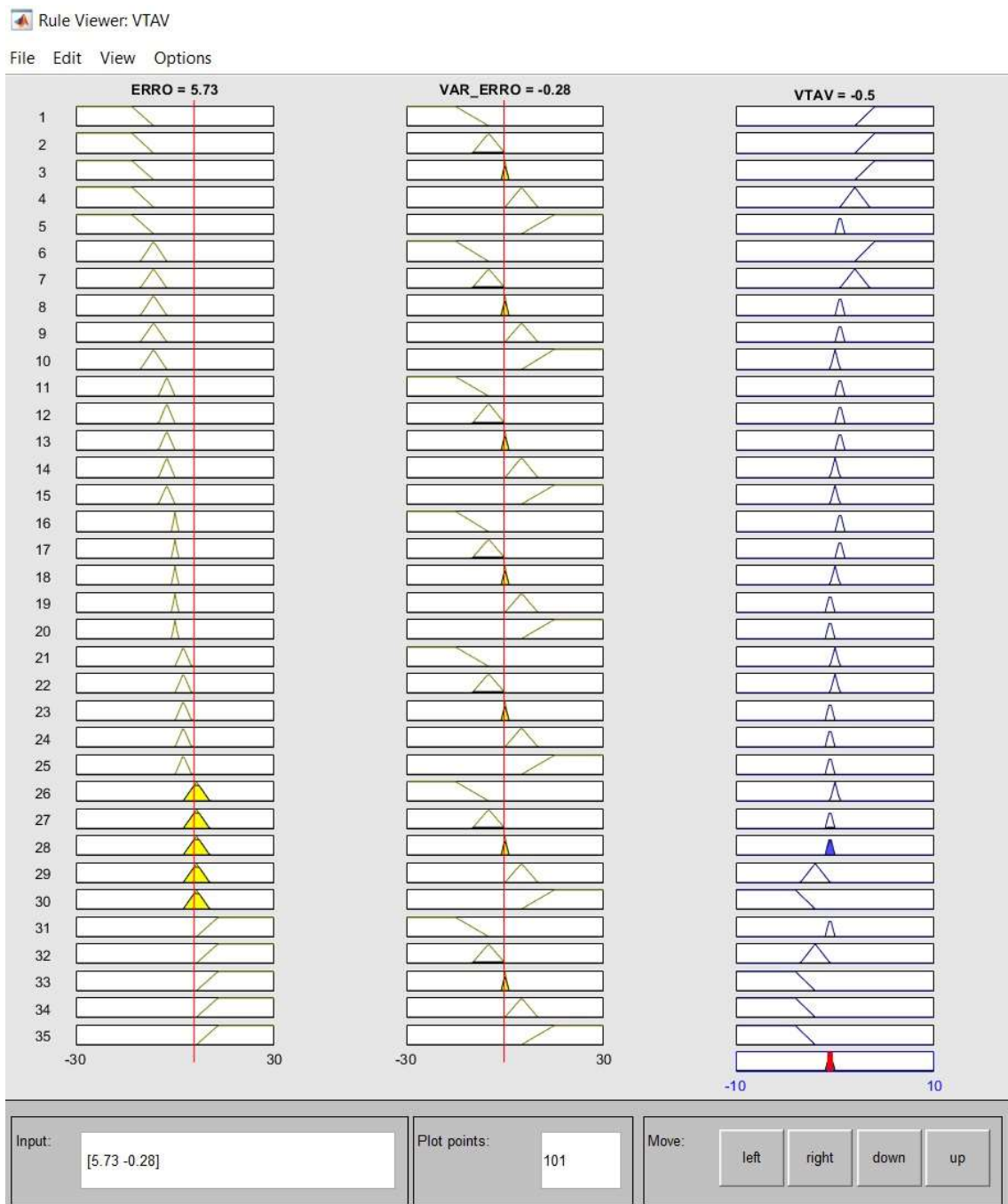
Figura 34 – Terceira iteração da simulação 4 (Apêndice E): Arduino Mega 2560.

```

16:44:41,334 -> Massa medida: 208,44
16:44:41,387 -> Erro: 5,73
16:44:41,387 -> Var Erro: -0,28
16:44:41,387 -> VTAV: -0,50
  
```

Fonte: Elaborado pelo autor.

Figura 35 – Terceira iteração da simulação 4: Matlab.

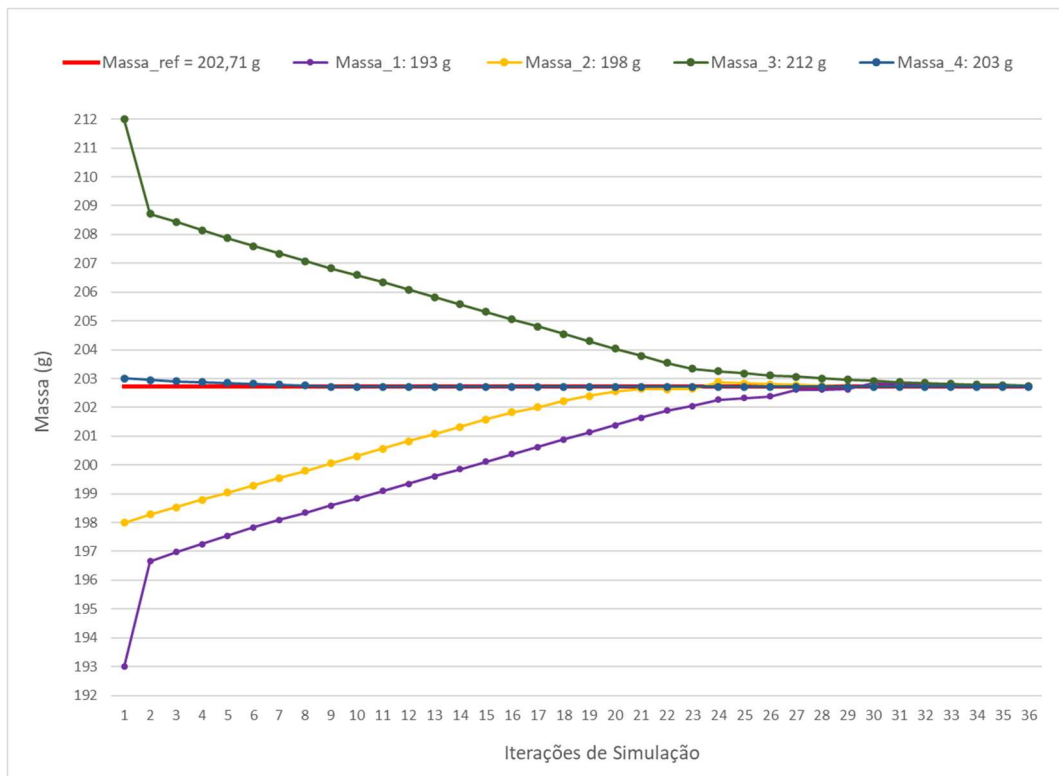


Fonte: Elaborado pelo autor.

Pode-se observar que foram obtidos os mesmos resultados, tanto na simulação em Arduino, quanto em Matlab, o que demonstra que a IDE do Arduino Mega 2560 está implementado corretamente o conjunto de regras elaborado no Matlab.

A Figura 36 apresenta o gráfico de resposta do controlador *fuzzy*, elaborado a partir dos resultados obtidos com as quatro simulações mencionadas anteriormente:

Figura 36: Gráfico de resposta do controlador *fuzzy*.



Fonte: Elaborado pelo autor.

Pelo gráfico apresentado na Figura 35, é possível observar que o controlador atua no sistema, convergindo as massas para o valor de referência. Quanto mais próximo é o valor da massa medida do valor de referência, menos iterações são realizadas para se conseguir o ajuste. Isso demonstra a validação do controlador proposto nesse trabalho, tendo em vista que os resultados obtidos com a simulação foram satisfatórios.

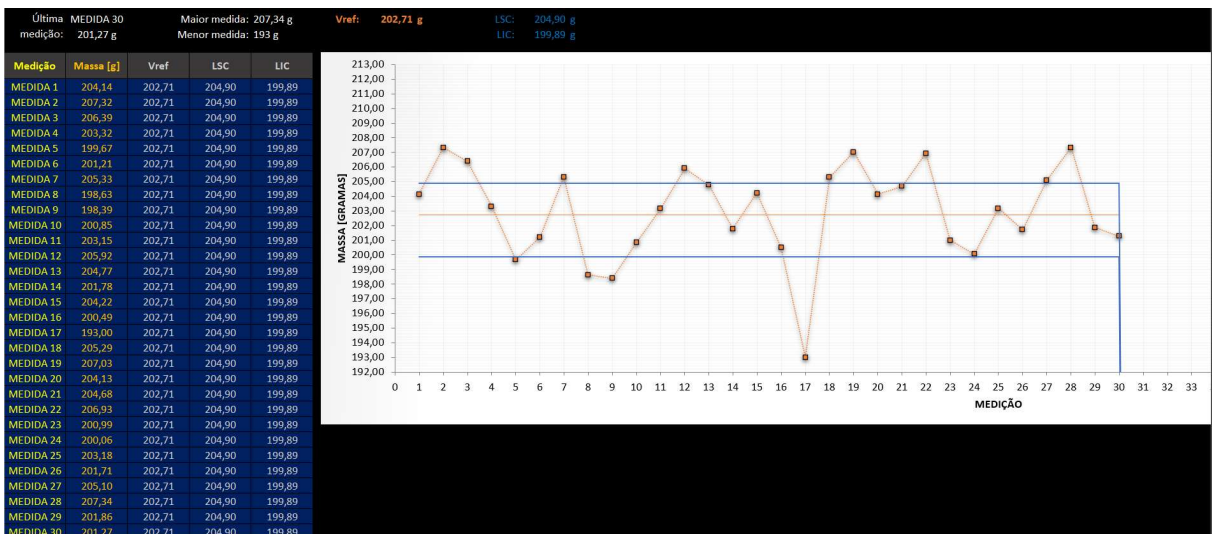
Destaca-se que cada vez que se faz a simulação para um mesmo valor de massa medido, não se obtém exatamente os mesmos resultados nas iterações, justamente por se tratar de um sistema *fuzzy* e os universos de discurso das funções permitem essa variação. Em função disso, o tempo de ajuste também pode apresentar uma pequena variação. Entretanto, o controlador se mostrou eficiente em todas os testes realizados.

4.2 GRÁFICO DE MONITORAMENTO

No sistema de monitoramento proposto, o Arduino Leonardo recebe os dados de leitura da balança e envia para o Microsoft Excel que, por sua vez, plota o gráfico em tempo real com as massas medidas, incluindo os limites de controle e os estabelecidos pelo Instituto Nacional de Metrologia, Qualidade e Tecnologia (Inmetro), para que o operador possa acompanhar, tanto o estado de controle do processo, quanto se as amostras atendem aos padrões de qualidade exigidos pelo órgão regulamentador.

A partir dos estudos de CEP, apresentado no subcapítulo 3.2, tem-se que os limites de controle desse processo de envase são: LIC = 199,89 g e LSC = 204,90 g. A Portaria do Inmetro n° 248, de 17 de julho de 2008, apresenta o Regulamento Técnico Metrológico que estabelece os critérios para verificação do conteúdo líquido de produtos pré-medidos com conteúdo nominal igual, comercializados nas grandezas de massa e volume. De acordo com esta Portaria do Inmetro, para conteúdos nominais entre 200 g e 300 g, a Tolerância Individual Permitida (TIP) é de ± 9 g em relação ao valor de referência (Vref), ou seja, para o processo de envase em estudo, Vref é 202,17g e as massas das embalagens podem variar de 193,17 g (TIPI – TIP inferior) a 211,71 g (TIPS – TIP superior). A amostragem deve ser reprovada se for verificada a ocorrência de mais de cinco unidades abaixo de 193,17 g, visto que quando o limite inferior não é respeitado, o consumidor é o principal prejudicado (INMETRO, 2008). A Figura 37 apresenta o gráfico de monitoramento das massas medidas plotado em Microsoft Excel em tempo real:

Figura 37: Gráfico de monitoramento do processo de envase em tempo real.



Fonte: Elaborado pelo autor.

Pode-se verificar que este gráfico é uma ferramenta capaz de agilizar a identificação de problemas no processo e a implementação de ações corretivas, de modo a contribuir para a redução de perdas de unidades não conformes e, assim, melhorar a qualidade, tanto do processo, quanto dos produtos fabricados.

5 CONSIDERAÇÕES FINAIS

Nas últimas décadas, as inovações tecnológicas impactaram o ser humano de tal forma que, hoje em dia, ninguém consegue imaginar como seria a vida sem conectividade e equipamentos eletroeletrônicos. As organizações industriais também estão inseridas nessa revolução e, cada vez mais, a mão de obra vem sendo suprida por máquinas automáticas, conectadas e eficientes, cujo propósito é elevar continuamente a qualidade dos processos, produtos e serviços.

Neste contexto, está sendo proposta uma inovação para o processo de envase da máquina Prepac-Dermec 1000, a partir do sistema de controle automático que utiliza controlador *fuzzy* embarcado em *hardware* de baixo custo. Pode-se dizer que os principais objetivos de se agregar automação e inteligência artificial a este processo é elevar a qualidade da produção e, ao mesmo tempo, reduzir os custos referentes a mão de obra, manutenção e perdas de unidades não conformes. O fato de se utilizar um dispositivo *open source* associado a disponibilidade gratuita da biblioteca *eFLL*, mostra que nem sempre uma inovação significa complexidade técnica e alto custo de investimento.

Quando se trabalha com lógica *fuzzy*, sabe-se que a experiência do projetista tem um peso relevante para se obter bons resultados. No caso deste trabalho, pode-se afirmar que seu próprio desenvolvimento foi uma importante ferramenta para aquisição de experiência, não só no que diz respeito a lógica *fuzzy*, mas, principalmente, no que se refere a construção dos códigos que foram embarcados nos Arduinos, tarefa que se traduziu em inúmeros procedimentos de ajustes e reajustes, até se alcançar o resultado desejado.

Os resultados obtidos com as simulações do controlador *fuzzy* embarcado em Arduino foram satisfatórios, verificando-se que o sistema realiza uma busca precisa e automática do controle quando uma perturbação é inserida, alcançando a estabilidade após realizar as iterações próprias de um sistema com *feedback control*.

Por apresentar os detalhamentos construtivos do controlador PD *fuzzy*, do procedimento para embarcá-lo no Arduino e de como conectar esse dispositivo à balança de medição e ao Microsoft Excel, este trabalho se traduz em uma fonte de consulta com potencial para auxiliar outros pesquisadores no desenvolvimento de trabalhos nesta área.

Como proposta para trabalhos futuros, sugere-se a implantação desse controlador na máquina Prepac-Dermec 1000, unindo todos os elementos (balança, controlador, Microsoft Excel) para validar experimentalmente o sistema de monitoramento e controle proposto.

REFERÊNCIAS

- ALVES, A. J. Eflil (eFLL): uma biblioteca fuzzy para Arduino e sistemas embarcados. *In*: ALVES, A. J. **Blog Zerokol**. Piauí, 28 set. 2012. Disponível em: <https://blog.zerokol.com/2012/09/arduinofuzzy-uma-biblioteca-fuzzy-para.html>. Acesso em: 15 jul. 2019.
- ARDUINO. **Getting started**. 2021. Disponível em: <https://www.arduino.cc/>. Acesso em: 23 abr. 2021.
- BANZI, M.; SHILOH, M. **Getting started with Arduino**. 3rd ed. Sebastopol: Maker Mídia, 2014.
- BAYER, F. M.; ARAÚJO, O. C. B. **Controle automático de processos**. 3.ed. Santa Maria: Universidade Federal Santa Maria, Colégio Técnico Industrial de Santa Maria, 2011. *E-book*. Disponível em: http://redeotec.mec.gov.br/images/stories/pdf/eixo_ctrl_proc_indust/tec_autom_ind/ctrl_auto_proc/161012_contr_aut_proc.pdf. Acesso em: 13 dez. 2020.
- COSTA, A. F. B.; EPPRECHT, E. K.; CARPINETTI, C. R. **Controle estatístico de qualidade**. 2.ed. São Paulo: Atlas; 2005.
- CRUZ, R. A. P. A. **Construção dos gráficos de Shewhart e avaliação de sua eficiência no controle de processos de envase**. 2019. Dissertação (Mestrado em Engenharia de Produção) – Faculdade de Engenharia de Guaratinguetá, Universidade Estadual Paulista, Guaratinguetá, 2019.
- FONSECA, R. P. F. **Criação de interface gráfica virtual aplicada a um sistema de controle de processos**. Dissertação (Mestrado em Engenharia Mecânica) – Instituto Superior de Engenharia de Coimbra, Portugal, Coimbra, 2016.
- GIL, A. C. **Como elaborar projetos de pesquisa**. 5. ed. São Paulo: Atlas, 2010.
- GYORI, C. A. M. **Lógica fuzzy como ferramenta de controle de processo**. 2019. Monografia (Especialização em Gestão da Produção) – Faculdade de Engenharia de Guaratinguetá, Universidade Estadual Paulista, Guaratinguetá, 2019.
- IBRAHIM, M. A. **Fuzzy logic for embedded systems applications**. Burlington: Elsevier, 2004.
- INMETRO - Instituto Nacional de Metrologia, Qualidade e Tecnologia. **Portaria Inmetro nº 248**, 17 jul. 2008. Duque de Caxias: INMETRO, 2008. Disponível em: <chrome-extension://efaidnbmnnnibpcajpcglclefindmkaj/viewer.html?pdfurl=http%3A%2F%2Fwww.inmetro.gov.br%2Flegislacao%2Frtac%2Fpdf%2Frtac001339.pdf&cLen=165912&chunk=true>. Acesso em: 6 maio 2021.
- KOTHARI, C. **Research methodology: methods and techniques**. 2nd ed. New Delhi: New Age International, 2013.

KOVACIC, Z.; BOGDAN, S. FAULKNER, L. **Fuzzy controller design: theory and applications**. Boca Raton: CRC Press, 2005.

LAKATOS, E. M.; MARCONI, M. A. **Fundamentos de metodologia científica**. 7. ed. São Paulo: Atlas, 2010.

LEAL, L. B. **Uma abordagem para estimação da qualidade de rotas em redes de sensores sem fio multi-sink baseada em sistemas fuzzy genéticos**. 2011. Dissertação (Mestrado em Informática Aplicada) – Universidade de Fortaleza, Fortaleza, 2011.

LOPES, D. V. **Correção de causas especiais no envase de saquinhos plásticos**. 2018. Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica) – Faculdade de Engenharia de Guaratinguetá, Universidade Estadual Paulista, Guaratinguetá, 2018.

LOTUFO, F. A. **Introdução ao Matlab e Simulink**. Guaratinguetá: Faculdade de Engenharia de Guaratinguetá, Universidade Estadual Paulista, 2009. Apostila. Disponível em: http://www.feg.unesp.br/Home/PaginasPessoais/proffranciscoantoniolotufo/lab1_mae_09.pdf. Acesso em: 18. ago. 2020.

MAMDANI, E. H.; ASSILIAN, S. An experiment in linguistic synthesis with a fuzzy logic controller. **International Journal of Man-Machine Studies**, London, v. 7, n. 1, p. 1-13, 1975. Disponível em: <https://www.sciencedirect.com/science/article/abs/pii/S0020737375800022>. Acesso em: 13 Jan. 2021.

MARQUES, J. E. S. Aplicação da lógica fuzzy no controle da velocidade de um protótipo móvel. **C&D-Revista Eletrônica da Fainor**, Vitória da Conquista, v. 9, n. 1, p. 15-25, 2016. Disponível em: <https://silo.tips/download/aplicacao-da-logica-fuzzy-no-controle-da-velocidade-de-um-prototipo-movel#>. Acesso em: 13 dez. 2020.

MATHWORKS. **Fuzzy logic toolbox**. Natick, MA: The MathWorks, 2021. Disponível em: <https://www.mathworks.com/products/fuzzy-logic.html>. Acesso em: 10 maio 2021.

MDIC - Ministério do Desenvolvimento, Indústria, Comércio Exterior e Serviços. **Jornada para a indústria 4.0**. Brasília: MDIC, 2019. Disponível em: <http://www.industria40.gov.br/>. Acesso em: 15 jul. 2020.

MENDEL, J. M. **Uncertain rule-based fuzzy logic systems: introduction and new directions**. Upper-Saddle River: Prentice-Hall, 2001.

MINCHALA, L.I.; PERALTA, J.; MATA-QUEVEDO, P.; ROJAS, J. An approach to industrial automation based on low-cost embedded platforms and open software. **Applied Sciences**. v. 10, n.14, 2020. Disponível em: <https://doi.org/10.3390/app10144696>. Acesso em: 21 out. 2021.

MONTGOMERY, D. C. **Introduction to statistical quality control**. 7th ed. Arizona: John Wiley & Sons, 2012.

MULLER, L. V. **Sistema de monitoramento da frequência cardíaca via lógica fuzzy em bicicleta ergométrica com microgeração de energia.** 2019. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Tecnológica Federal do Paraná, Curitiba, 2019.

NISE, N. S. **Engenharia de sistemas de controle.** 5. ed. Rio de Janeiro: LTC, 2011.

NOGUEIRA, J. C. B.; SERVEDIO, Y. **Implementação de controlador fuzzy em manipulador robótico.** 2015. Trabalho de Conclusão de Curso (Graduação em Engenharia de Controle e Automação) – Instituto Federal de Educação, Ciência e Tecnologia Fluminense, Campos de Goytacazes, 2015.

NOGUEIRA, M. M. **Aplicando lógica fuzzy no controle de robôs móveis usando dispositivos lógicos programáveis e a linguagem VHDL.** 2013. Dissertação (Mestrado em Engenharia Elétrica) – Faculdade de Engenharia de Ilha Solteira, Universidade Estadual Paulista, Ilha Solteira, 2013.

OGATA, K. **Engenharia de controle moderno.** 5. ed. São Paulo: Prentice Hall, 2010.

PEARCE, J. M. Economic savings for scientific free and open source technology: a review. **HardwareX**, Oxford, v. 8. 2020. Disponível em: <https://doi.org/10.1016/j.ohx.2020.e00139>. Acesso em: 21 Oct. 2021.

RIBEIRO, J. L. D.; TEN CATEN, C. S. **Controle estatístico de processo.** Porto Alegre: FEENG/UFRGS, 2012. *E-book*. Disponível em: http://www.producao.ufrgs.br/arquivos/disciplinas/388_apostilacep_2012.pdf. Acesso em: 7 ago. 2020.

RIZOL, P. M. S. R.; DIAS, R. A. Desmistificando a lógica fuzzy. *In*: CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA, 62., 2014, Juiz de Fora. **Anais [...]**. Juiz de Fora, 2014.

SALDANHA, P.; ROTHE, C. K.; BENEDETT, F. R.; PACHECO D. A. J.; JUNG, C. F.; TEN CATEN, C. S. Analisando a aplicação do controle estatístico de processos na indústria química: um estudo de caso. **Revista Spacios**, v. 34, n. 11, p. 17, 2013. Disponível em: <https://www.revistaespacios.com/a13v34n11/13341117.html>. Acesso em: 7 ago. 2020.

SILVA, G. C.; MALVEIRA, B. M.; CARNEIRO, J. R. G.; BRITO, P.P.; SILVA, T. A. Wear and thermal analysis of WC inserts in turning operations by fuzzy modeling. **Procedia CIRP**, Amsterdam, v. 58, p. 523-528, 2017. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2212827117304493>. Acesso em: 7 ago. 2020.

SOARES, G. P. **Aplicação do controle estatístico de processos em indústria de bebidas: um estudo de caso.** 2001. Dissertação (Mestrado em Engenharia de Produção) – Universidade Federal de Santa Catarina, Florianópolis, 2001. Disponível em: <http://repositorio.ufsc.br/xmlui/handle/123456789/82229>. Acesso em: 7 ago. 2020.

SUGENO, M. An introductory survey of fuzzy control information. **Information Sciences**, London, v. 36, p. 59-83. 1974. Disponível em: [https://doi.org/10.1016/0020-0255\(85\)90026-X](https://doi.org/10.1016/0020-0255(85)90026-X). Acesso em: 13 set. 2020.

SZESZ JUNIOR, A.; MONTEIRO JUNIOR, M.; DIAS, A. H.; MATHIAS, I. M.; CONTI, G. Embedded system in Arduino platform with fuzzy control to support the grain aeration decision. **Ciência Rural**, Santa Maria, v. 46, n. 11, p. 1917-1923, 2016. Disponível em: <https://www.scielo.br/j/cr/a/fWMWZm3rMbZt6V8bbXHjJbD/?lang=en>. Acesso em: 2 abr. 2021.

TANAKA, K. **An introduction to fuzzy logic for practical applications**. Kanazawa: Springer, 1996.

TANSCHHEIT, R. Sistemas fuzzy. *In*: OLIVEIRA JUNIOR, H.A. (org.). **Inteligência computacional**: aplicada à administração, economia e engenharia em Matlab. São Paulo: Thomson Learning, p. 229-264, 2007.

THOMSEN, A. O que é Arduino. *In*: THOMSEN, A. **Blog Arduino Developers**. [s. l.], 19 maio 2018. Disponível em: <https://desenvolvendoparaoarduino.wordpress.com/2018/05/19/o-que-e-arduino/>. Acesso em: 17 jun. 2020.

XUE, D.; CHEN, Y. **Modelling, analysis and design of control systems in Matlab and Simulink**. Shenyang: World Scientific Publishing, 2014.

ZADEH, L. A. Fuzzy sets. **Information and Control**, Berkeley, v. 8, n. 3, p. 338-353, 1965. Disponível em: [https://doi.org/10.1016/S0019-9958\(65\)90241-X](https://doi.org/10.1016/S0019-9958(65)90241-X). Acesso em: 13 out. 2020.

ZAKI, A. M.; EL-BARDINI, M.; SOLIMAN, F.A.S.; SHARAF, M. M. Embedded two level direct adaptive fuzzy controller for DC motor speed control. **Ain Shams Engineering Journal**, Cairo, v. 9, p. 65–75, 2018.

APÊNDICE A – CÓDIGO PARA EMBARCAR O CONTROLADOR FUZZY NO ARDUINO MEGA 2560

```

// MONITORAMENTO AUTOMÁTICO DE UM PROCESSO DE ENVASE UTILIZANDO CONTROLADOR
FUZZY EMBARCADO EM HARDWARE DE BAIXO CUSTO.
//Carlos A. M. Gyori, Marcela A. G. M. de Freitas e Paloma M. S. Rocha
Rizol
//Universidade Estadual Paulista (UNESP) "Júlio de Mesquita Filho", Campus
de Guaratinguetá, Brasil.
//{carlos.gyori, marcela.freitas, paloma.rizol}@unesp.br

//*****BIBLIOTECAS FUZZY*****
#include <Fuzzy.h>
#include <FuzzySet.h>
#include <FuzzyComposition.h>
#include <FuzzyIO.h>
#include <FuzzyInput.h>
#include <FuzzyOutput.h>
#include <FuzzyRule.h>
#include <FuzzyRuleAntecedent.h>
#include <FuzzyRuleConsequent.h>

//*****PARAMETROS FUZZY*****
boolean entra = true;
float output; // saída do controlador fuzzy (VTAV).
float Vref=202.71; // aqui é definido o peso de referência do saquinho.
float peso; // indicar o peso inicial?
float anterior=0;
float atual=0;
float entrada1=0;
float entrada2=0;
float precisao=0.01;
int iteracao=1; //contador de iterações

//*****FUZZY*****
//Instanciando o objeto fuzzy
Fuzzy *fuzzy = new Fuzzy();

//*****BALANÇA*****
// OBJETO PARA MÓDULO DA BALANÇA
#include <HX711.h>

// DEFINIÇÕES DE PINOS
#define pinDT 4 //HX711
#define pinSCK 5 //HX711

#define led_sistema 6 //verde
#define led_balanca 7 //azul
#define led_pause 8 //vermelho
#define pinPause 12 //pino de pause

// DEFINIÇÕES DA BALANÇA
#define pesoMin 0.1
#define pesoMax 300.0
#define escala 343000.0f

//Instanciando o objeto balança
HX711 scale;

```



```

// DECLARAÇÃO DE VARIÁVEIS
float medida=0; //balança
float medida1=0; //balança
int counter = 1; //contador de medições

//Pino de pause
void estadoPause();

//*****
//EXECUTADO QUANDO O ARDUINO FOR LIGADO
void setup() {
  Serial.begin(9600);
  Serial.println("Iniciando o sistema...");

  //INICIO DOS PINOS DO SISTEMA
  pinMode(pinPause, INPUT_PULLUP);
  pinMode(led_sistema, OUTPUT);
  pinMode(led_balanca, OUTPUT);
  pinMode(led_pause, OUTPUT);
  delay(1000);
  digitalWrite(led_sistema, HIGH);
  delay(1000);
  digitalWrite(led_balanca, HIGH);
  delay(1000);
  digitalWrite(led_pause, HIGH);
  delay(1000);
  digitalWrite(led_sistema, LOW);
  digitalWrite(led_balanca, LOW);
  digitalWrite(led_pause, LOW);

  //INICIO DA BALANÇA
  scale.begin(pinDT, pinSCK); // CONFIGURANDO OS PINOS DA BALANÇA
  scale.set_scale(escala); // ENVIANDO O VALOR DA ESCALA CALIBRADO
  digitalWrite(led_sistema, HIGH);
  delay(200);
  digitalWrite(led_sistema, LOW);
  delay(200);
  digitalWrite(led_sistema, HIGH);
  delay(200);
  digitalWrite(led_sistema, LOW);
  delay(200);
  digitalWrite(led_sistema, HIGH); //LIBERADO PARA REALIZAR A TARA
  digitalWrite(led_balanca, HIGH);
  scale.power_up(); // LIGANDO O SENSOR DA BALANÇA
  digitalWrite(led_sistema, LOW);
  delay(200);
  digitalWrite(led_balanca, LOW);
  delay(200);

  //TARA DA BALANÇA
  medida = scale.get_units(10); // SALVANDO NA VARIÁVEL O VALOR DA MÉDIA DE
10 MEDIDAS
  digitalWrite(led_sistema, HIGH);
  scale.tare(); // ZERA A BALANÇA CASO A MASSA SEJA MENOR QUE O VALOR MIN
  medida = 0;
  digitalWrite(led_sistema, LOW);
  delay(500);
  digitalWrite(led_sistema, HIGH);
  Serial.println("Sistema pronto. Aguardando pesagem.");
}

```

```

//*****ENTRADA1*****
// Variável de entrada "Erro (ERRO)"
FuzzyInput *ERRO = new FuzzyInput(1);

// MF 1 = "ENA": Erro Negativo Alto (Peso muito abaixo do valor de
referência).
FuzzySet *ENA = new FuzzySet(-30, -30, -13, -6);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(ENA);

// MF 2 = "ENM": Erro Negativo Médio (Peso abaixo do valor de referência).
FuzzySet *ENM = new FuzzySet(-10.5, -6.5, -6.5, -2.5);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(ENM);

// MF 3 = "ENB": Erro Negativo Baixo (Peso pouco abaixo do valor de
referência).
FuzzySet *ENB = new FuzzySet(-5, -2.5, -2.5, 0);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(ENB);

// MF 4 = "EZERO": Erro Zero (Peso igual ao valor de referência).
FuzzySet *EZERO = new FuzzySet(-1, 0, 0, 1);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(EZERO);

// MF 5 = "EPB": Erro Positivo Baixo (Peso pouco acima do valor de
referência).
FuzzySet *EPB = new FuzzySet(0, 2.5, 2.5, 5);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(EPB);

// MF 6 = "EPM": Erro Positivo Médio (Peso acima do valor de referência).
FuzzySet *EPM = new FuzzySet(2.5, 6.5, 6.5, 10.5);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(EPM);

// MF 7 = "EPA": Erro Positivo Alto (Peso muito acima do valor de
referência).
FuzzySet *EPA = new FuzzySet(6.5, 13, 30, 30);
// Incluindo o FuzzySet no FuzzyInput
ERRO->addFuzzySet(EPA);

// Atribuindo as entradas no FIS
fuzzy->addFuzzyInput(ERRO);

//*****ENTRADA2*****
// Variável de entrada "Variação do Erro (VAR_ERRO)"
FuzzyInput *VARERRO = new FuzzyInput(2);

// MF 1 = "VEDA": Variação do Erro Decrescente Alta (O erro diminuiu muito
após o ajuste).
FuzzySet *VEDA = new FuzzySet(-30, -30, -15, -5);
// Incluindo o FuzzySet no FuzzyInput
VARERRO->addFuzzySet(VEDA);

// MF 2 = "VEDB": Variação do Erro Decrescente Baixa (O erro diminuiu pouco
após o ajuste).

```

```

FuzzySet *VEDB = new FuzzySet(-10, -5, -5, 0);
// Incluindo o FuzzySet no FuzzyInput
VARERRO->addFuzzySet (VEDB);

// MF 3 = "VEZERO": Variação do Erro Zero (O erro não variou após o
ajuste).
FuzzySet *VEZERO = new FuzzySet(-1, 0, 0, 1);
// Incluindo o FuzzySet no FuzzyInput
VARERRO->addFuzzySet (VEZERO);

// MF 4 = "VECB": Variação do Erro Crescente Baixa (O erro aumentou pouco
após o ajuste).
FuzzySet *VECB = new FuzzySet(0, 5, 5, 10);
// Incluindo o FuzzySet no FuzzyInput
VARERRO->addFuzzySet (VECB);

// MF 5 = "VECA": Variação do Erro Crescente Alta (O erro aumentou muito
após o ajuste).
FuzzySet *VECA = new FuzzySet(5, 15, 30, 30);
// Incluindo o FuzzySet no FuzzyInput
VARERRO->addFuzzySet (VECA);

// Atribuindo as entradas no FIS
fuzzy->addFuzzyInput (VARERRO);

//*****SAIDA*****
// Variável de saída "Variação do Tempo de Abertura da Válvula (VTAV)"
FuzzyOutput *VTAV = new FuzzyOutput(1);

// MF 1 = "TAA": Aumento de Tempo Alto (Aumenta muito o tempo de abertura
da válvula).
FuzzySet *TDA = new FuzzySet(-10, -10, -4, -2);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet (TDA);

// MF 2 = "TAM": Aumento de Tempo Médio (Aumenta o tempo de abertura da
válvula).
FuzzySet *TDM = new FuzzySet(-3.5, -2, -2, -0.5);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet (TDM);

// MF 3 = "TAB": Aumento de Tempo Baixo (Aumenta pouco o tempo de abertura
da válvula).
FuzzySet *TDB = new FuzzySet(-1, -0.5, -0.5, 0);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet (TDB);

// MF 4 = "TMANTER": Mantém o tempo (Mantém o tempo de abertura ajustado).
FuzzySet *TMANTER = new FuzzySet(-0.5, 0, 0, 0.5);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet (TMANTER);

// MF 5 = "TDB": Diminuição de Tempo Baixa (reduz pouco o tempo de abertura
da válvula).
FuzzySet *TAB = new FuzzySet(0, 0.5, 0.5, 1);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet (TAB);

// MF 6 = "TDM": Diminuição de Tempo Média (reduz o tempo de abertura da
válvula).

```

```

FuzzySet *TAM = new FuzzySet(0.5, 2, 2, 3.5);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet(TAM);

// MF 7 = "TDA": Diminuição de Tempo Alta (reduz muito o tempo de abertura
da válvula).
FuzzySet *TAA = new FuzzySet(2, 4, 10, 10);
// Including the FuzzySet into FuzzyOutput
VTAV->addFuzzySet(TAA);

// Atribuindo a saída no FIS
fuzzy->addFuzzyOutput(VTAV);

//*****CONJUNTO DE REGRAS*****

// Rule 01: SE o ERRO é ENA E a VARERRO é VEDA ENTÃO o VTAV é TAA
FuzzyRuleAntecedent *ENA_VEDA = new FuzzyRuleAntecedent();
ENA_VEDA->joinWithAND(ENA, VEDA);
FuzzyRuleConsequent* thenVTAV1 = new FuzzyRuleConsequent();
thenVTAV1->addOutput(TAA);
FuzzyRule* fuzzyRule1 = new FuzzyRule(1, ENA_VEDA, thenVTAV1);
fuzzy->addFuzzyRule(fuzzyRule1);

// Rule 02: SE o ERRO é ENA E a VARERRO é VEDB ENTÃO o VTAV é TAA
FuzzyRuleAntecedent *ENA_VEDB = new FuzzyRuleAntecedent();
ENA_VEDB->joinWithAND(ENA, VEDB);
FuzzyRuleConsequent* thenVTAV2 = new FuzzyRuleConsequent();
thenVTAV2->addOutput(TAA);
FuzzyRule* fuzzyRule2 = new FuzzyRule(2, ENA_VEDB, thenVTAV2);
fuzzy->addFuzzyRule(fuzzyRule2);

// Rule 03: SE o ERRO é ENA E a VARERRO é VEZERO ENTÃO o VTAV é TAA
FuzzyRuleAntecedent *ENA_VEZERO = new FuzzyRuleAntecedent();
ENA_VEZERO->joinWithAND(ENA, VEZERO);
FuzzyRuleConsequent* thenVTAV3 = new FuzzyRuleConsequent();
thenVTAV3->addOutput(TAA);
FuzzyRule* fuzzyRule3 = new FuzzyRule(3, ENA_VEZERO, thenVTAV3);
fuzzy->addFuzzyRule(fuzzyRule3);

// Rule 04: SE o ERRO é ENA E a VARERRO é VECB ENTÃO o VTAV é TAM
FuzzyRuleAntecedent *ENA_VECB = new FuzzyRuleAntecedent();
ENA_VECB->joinWithAND(ENA, VECB);
FuzzyRuleConsequent* thenVTAV4 = new FuzzyRuleConsequent();
thenVTAV4->addOutput(TAM);
FuzzyRule* fuzzyRule4 = new FuzzyRule(4, ENA_VECB, thenVTAV4);
fuzzy->addFuzzyRule(fuzzyRule4);

// Rule 05: SE o ERRO é ENA E a VARERRO é VECA ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *ENA_VECA = new FuzzyRuleAntecedent();
ENA_VECA->joinWithAND(ENA, VECA);
FuzzyRuleConsequent* thenVTAV5 = new FuzzyRuleConsequent();
thenVTAV5->addOutput(TAB);
FuzzyRule* fuzzyRule5 = new FuzzyRule(5, ENA_VECA, thenVTAV5);
fuzzy->addFuzzyRule(fuzzyRule5);

// Rule 06: SE o ERRO é ENM E a VARERRO é VEDA ENTÃO o VTAV é TAA
FuzzyRuleAntecedent *ENM_VEDA = new FuzzyRuleAntecedent();
ENM_VEDA->joinWithAND(ENM, VEDA);
FuzzyRuleConsequent* thenVTAV6 = new FuzzyRuleConsequent();
thenVTAV6->addOutput(TAA);

```

```

    FuzzyRule* fuzzyRule6 = new FuzzyRule(6, ENM_VEDA, thenVTAV6);
    fuzzy->addFuzzyRule(fuzzyRule6);

// Rule 07: SE o ERRO é ENM E a VARERRO é VEDB ENTÃO o VTAV é TAM
FuzzyRuleAntecedent *ENM_VEDB = new FuzzyRuleAntecedent();
    ENM_VEDB->joinWithAND(ENM, VEDB);
    FuzzyRuleConsequent* thenVTAV7 = new FuzzyRuleConsequent();
    thenVTAV7->addOutput(TAM);
    FuzzyRule* fuzzyRule7 = new FuzzyRule(7, ENM_VEDB, thenVTAV7);
    fuzzy->addFuzzyRule(fuzzyRule7);

// Rule 08: SE o ERRO é ENM E a VARERRO é VEZERO ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *ENM_VEZERO = new FuzzyRuleAntecedent();
    ENM_VEZERO->joinWithAND(ENM, VEZERO);
    FuzzyRuleConsequent* thenVTAV8 = new FuzzyRuleConsequent();
    thenVTAV8->addOutput(TAB);
    FuzzyRule* fuzzyRule8 = new FuzzyRule(8, ENM_VEZERO, thenVTAV8);
    fuzzy->addFuzzyRule(fuzzyRule8);

// Rule 09: SE o ERRO é ENM E a VARERRO é VECB ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *ENM_VECB = new FuzzyRuleAntecedent();
    ENM_VECB->joinWithAND(ENM, VECB);
    FuzzyRuleConsequent* thenVTAV9 = new FuzzyRuleConsequent();
    thenVTAV9->addOutput(TAB);
    FuzzyRule* fuzzyRule9 = new FuzzyRule(9, ENM_VECB, thenVTAV9);
    fuzzy->addFuzzyRule(fuzzyRule9);

// Rule 10: SE o ERRO é ENM E a VARERRO é VECA ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *ENM_VECA = new FuzzyRuleAntecedent();
    ENM_VECA->joinWithAND(ENM, VECA);
    FuzzyRuleConsequent* thenVTAV10 = new FuzzyRuleConsequent();
    thenVTAV10->addOutput(TMANTER);
    FuzzyRule* fuzzyRule10 = new FuzzyRule(10, ENM_VECA, thenVTAV10);
    fuzzy->addFuzzyRule(fuzzyRule10);

// Rule 11: SE o ERRO é ENB E a VARERRO é VEDA ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *ENB_VEDA = new FuzzyRuleAntecedent();
    ENB_VEDA->joinWithAND(ENB, VEDA);
    FuzzyRuleConsequent* thenVTAV11 = new FuzzyRuleConsequent();
    thenVTAV11->addOutput(TAB);
    FuzzyRule* fuzzyRule11 = new FuzzyRule(11, ENB_VEDA, thenVTAV11);
    fuzzy->addFuzzyRule(fuzzyRule11);

// Rule 12: SE o ERRO é ENB E a VARERRO é VEDB ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *ENB_VEDB = new FuzzyRuleAntecedent();
    ENB_VEDB->joinWithAND(ENB, VEDB);
    FuzzyRuleConsequent* thenVTAV12 = new FuzzyRuleConsequent();
    thenVTAV12->addOutput(TAB);
    FuzzyRule* fuzzyRule12 = new FuzzyRule(12, ENB_VEDB, thenVTAV12);
    fuzzy->addFuzzyRule(fuzzyRule12);

// Rule 13: SE o ERRO é ENB E a VARERRO é VEZERO ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *ENB_VEZERO = new FuzzyRuleAntecedent();
    ENB_VEZERO->joinWithAND(ENB, VEZERO);
    FuzzyRuleConsequent* thenVTAV13 = new FuzzyRuleConsequent();
    thenVTAV13->addOutput(TAB);
    FuzzyRule* fuzzyRule13 = new FuzzyRule(13, ENB_VEZERO, thenVTAV13);
    fuzzy->addFuzzyRule(fuzzyRule13);

// Rule 14: SE o ERRO é ENB E a VARERRO é VECB ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *ENB_VECB = new FuzzyRuleAntecedent();

```

```

    ENB_VECB->joinWithAND(ENB, VECB);
    FuzzyRuleConsequent* thenVTAV14 = new FuzzyRuleConsequent();
    thenVTAV14->addOutput(TMANTER);
    FuzzyRule* fuzzyRule14 = new FuzzyRule(14, ENB_VECB, thenVTAV14);
    fuzzy->addFuzzyRule(fuzzyRule14);

// Rule 15: SE o ERRO é ENB E a VARERRO é VECA ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *ENB_VECA = new FuzzyRuleAntecedent();
    ENB_VECA->joinWithAND(ENB, VECA);
    FuzzyRuleConsequent* thenVTAV15 = new FuzzyRuleConsequent();
    thenVTAV15->addOutput(TMANTER);
    FuzzyRule* fuzzyRule15 = new FuzzyRule(15, ENB_VECA, thenVTAV15);
    fuzzy->addFuzzyRule(fuzzyRule15);

// Rule 16: SE o ERRO é EZERO E a VARERRO é VEDA ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *EZERO_VEDA = new FuzzyRuleAntecedent();
    EZERO_VEDA->joinWithAND(EZERO, VEDA);
    FuzzyRuleConsequent* thenVTAV16 = new FuzzyRuleConsequent();
    thenVTAV16->addOutput(TAB);
    FuzzyRule* fuzzyRule16 = new FuzzyRule(16, EZERO_VEDA, thenVTAV16);
    fuzzy->addFuzzyRule(fuzzyRule16);

// Rule 17: SE o ERRO é EZERO E a VARERRO é VEDB ENTÃO o VTAV é TAB
FuzzyRuleAntecedent *EZERO_VEDB = new FuzzyRuleAntecedent();
    EZERO_VEDB->joinWithAND(EZERO, VEDB);
    FuzzyRuleConsequent* thenVTAV17 = new FuzzyRuleConsequent();
    thenVTAV17->addOutput(TAB);
    FuzzyRule* fuzzyRule17 = new FuzzyRule(17, EZERO_VEDB, thenVTAV17);
    fuzzy->addFuzzyRule(fuzzyRule17);

// Rule 18: SE o ERRO é EZERO E a VARERRO é VEZERO ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *EZERO_VEZERO = new FuzzyRuleAntecedent();
    EZERO_VEZERO->joinWithAND(EZERO, VEZERO);
    FuzzyRuleConsequent* thenVTAV18 = new FuzzyRuleConsequent();
    thenVTAV18->addOutput(TMANTER);
    FuzzyRule* fuzzyRule18 = new FuzzyRule(18, EZERO_VEZERO, thenVTAV18);
    fuzzy->addFuzzyRule(fuzzyRule18);

// Rule 19: SE o ERRO é EZERO E a VARERRO é VECB ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EZERO_VECB = new FuzzyRuleAntecedent();
    EZERO_VECB->joinWithAND(EZERO, VECB);
    FuzzyRuleConsequent* thenVTAV19 = new FuzzyRuleConsequent();
    thenVTAV19->addOutput(TDB);
    FuzzyRule* fuzzyRule19 = new FuzzyRule(19, EZERO_VECB, thenVTAV19);
    fuzzy->addFuzzyRule(fuzzyRule19);

// Rule 20: SE o ERRO é EZERO E a VARERRO é VECA ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EZERO_VECA = new FuzzyRuleAntecedent();
    EZERO_VECA->joinWithAND(EZERO, VECA);
    FuzzyRuleConsequent* thenVTAV20 = new FuzzyRuleConsequent();
    thenVTAV20->addOutput(TDB);
    FuzzyRule* fuzzyRule20 = new FuzzyRule(20, EZERO_VECA, thenVTAV20);
    fuzzy->addFuzzyRule(fuzzyRule20);

// Rule 21: SE o ERRO é EPB E a VARERRO é VEDA ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *EPB_VEDA = new FuzzyRuleAntecedent();
    EPB_VEDA->joinWithAND(EPB, VEDA);
    FuzzyRuleConsequent* thenVTAV21 = new FuzzyRuleConsequent();
    thenVTAV21->addOutput(TMANTER);
    FuzzyRule* fuzzyRule21 = new FuzzyRule(21, EPB_VEDA, thenVTAV21);
    fuzzy->addFuzzyRule(fuzzyRule21);

```

```

// Rule 22: SE o ERRO é EPB E a VARERRO é VEDB ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *EPB_VEDB = new FuzzyRuleAntecedent();
EPB_VEDB->joinWithAND(EPB, VEDB);
FuzzyRuleConsequent* thenVTAV22 = new FuzzyRuleConsequent();
thenVTAV22->addOutput(TMANTER);
FuzzyRule* fuzzyRule22 = new FuzzyRule(22, EPB_VEDB, thenVTAV22);
fuzzy->addFuzzyRule(fuzzyRule22);

// Rule 23: SE o ERRO é EPB E a VARERRO é VEZERO ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EPB_VEZERO = new FuzzyRuleAntecedent();
EPB_VEZERO->joinWithAND(EPB, VEZERO);
FuzzyRuleConsequent* thenVTAV23 = new FuzzyRuleConsequent();
thenVTAV23->addOutput(TDB);
FuzzyRule* fuzzyRule23 = new FuzzyRule(23, EPB_VEZERO, thenVTAV23);
fuzzy->addFuzzyRule(fuzzyRule23);

// Rule 24: SE o ERRO é EPB E a VARERRO é VECB ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EPB_VECB = new FuzzyRuleAntecedent();
EPB_VECB->joinWithAND(EPB, VECB);
FuzzyRuleConsequent* thenVTAV24 = new FuzzyRuleConsequent();
thenVTAV24->addOutput(TDB);
FuzzyRule* fuzzyRule24 = new FuzzyRule(24, EPB_VECB, thenVTAV24);
fuzzy->addFuzzyRule(fuzzyRule24);

// Rule 25: SE o ERRO é EPB E a VARERRO é VECA ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EPB_VECA = new FuzzyRuleAntecedent();
EPB_VECA->joinWithAND(EPB, VECA);
FuzzyRuleConsequent* thenVTAV25 = new FuzzyRuleConsequent();
thenVTAV25->addOutput(TDB);
FuzzyRule* fuzzyRule25 = new FuzzyRule(25, EPB_VECA, thenVTAV25);
fuzzy->addFuzzyRule(fuzzyRule25);

// Rule 26: SE o ERRO é EPM E a VARERRO é VEDA ENTÃO o VTAV é TMANTER
FuzzyRuleAntecedent *EPM_VEDA = new FuzzyRuleAntecedent();
EPM_VEDA->joinWithAND(EPM, VEDA);
FuzzyRuleConsequent* thenVTAV26 = new FuzzyRuleConsequent();
thenVTAV26->addOutput(TMANTER);
FuzzyRule* fuzzyRule26 = new FuzzyRule(26, EPM_VEDA, thenVTAV26);
fuzzy->addFuzzyRule(fuzzyRule26);

// Rule 27: SE o ERRO é EPM E a VARERRO é VEDB ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EPM_VEDB = new FuzzyRuleAntecedent();
EPM_VEDB->joinWithAND(EPM, VEDB);
FuzzyRuleConsequent* thenVTAV27 = new FuzzyRuleConsequent();
thenVTAV27->addOutput(TDB);
FuzzyRule* fuzzyRule27 = new FuzzyRule(27, EPM_VEDB, thenVTAV27);
fuzzy->addFuzzyRule(fuzzyRule27);

// Rule 28: SE o ERRO é EPM E a VARERRO é VEZERO ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EPM_VEZERO = new FuzzyRuleAntecedent();
EPM_VEZERO->joinWithAND(EPM, VEZERO);
FuzzyRuleConsequent* thenVTAV28 = new FuzzyRuleConsequent();
thenVTAV28->addOutput(TDB);
FuzzyRule* fuzzyRule28 = new FuzzyRule(28, EPM_VEZERO, thenVTAV28);
fuzzy->addFuzzyRule(fuzzyRule28);

// Rule 29: SE o ERRO é EPM E a VARERRO é VECB ENTÃO o VTAV é TDM
FuzzyRuleAntecedent *EPM_VECB = new FuzzyRuleAntecedent();
EPM_VECB->joinWithAND(EPM, VECB);
FuzzyRuleConsequent* thenVTAV29 = new FuzzyRuleConsequent();

```

```

    thenVTAV29->addOutput(TDM);
    FuzzyRule* fuzzyRule29 = new FuzzyRule(29, EPM_VECA, thenVTAV29);
    fuzzy->addFuzzyRule(fuzzyRule29);

// Rule 30: SE o ERRO é EPM E a VARERRO é VECA ENTÃO o VTAV é TDA
FuzzyRuleAntecedent *EPM_VECA = new FuzzyRuleAntecedent();
EPM_VECA->joinWithAND(EPM, VECA);
FuzzyRuleConsequent* thenVTAV30 = new FuzzyRuleConsequent();
thenVTAV30->addOutput(TDA);
FuzzyRule* fuzzyRule30 = new FuzzyRule(30, EPM_VECA, thenVTAV30);
fuzzy->addFuzzyRule(fuzzyRule30);

// Rule 31: SE o ERRO é EPA E a VARERRO é VEDA ENTÃO o VTAV é TDB
FuzzyRuleAntecedent *EPA_VEDA = new FuzzyRuleAntecedent();
EPA_VEDA->joinWithAND(EPA, VEDA);
FuzzyRuleConsequent* thenVTAV31 = new FuzzyRuleConsequent();
thenVTAV31->addOutput(TDB);
FuzzyRule* fuzzyRule31 = new FuzzyRule(31, EPA_VEDA, thenVTAV31);
fuzzy->addFuzzyRule(fuzzyRule31);

// Rule 32: SE o ERRO é EPA E a VARERRO é VEDEB ENTÃO o VTAV é TDM
FuzzyRuleAntecedent *EPA_VEDEB = new FuzzyRuleAntecedent();
EPA_VEDEB->joinWithAND(EPA, VEDEB);
FuzzyRuleConsequent* thenVTAV32 = new FuzzyRuleConsequent();
thenVTAV32->addOutput(TDM);
FuzzyRule* fuzzyRule32 = new FuzzyRule(32, EPA_VEDEB, thenVTAV32);
fuzzy->addFuzzyRule(fuzzyRule32);

// Rule 33: SE o ERRO é EPA E a VARERRO é VEZERO ENTÃO o VTAV é TDA
FuzzyRuleAntecedent *EPA_VEZERO = new FuzzyRuleAntecedent();
EPA_VEZERO->joinWithAND(EPA, VEZERO);
FuzzyRuleConsequent* thenVTAV33 = new FuzzyRuleConsequent();
thenVTAV33->addOutput(TDA);
FuzzyRule* fuzzyRule33 = new FuzzyRule(33, EPA_VEZERO, thenVTAV33);
fuzzy->addFuzzyRule(fuzzyRule33);

// Rule 34: SE o ERRO é EPA E a VARERRO é VECB ENTÃO o VTAV é TDA
FuzzyRuleAntecedent *EPA_VECB = new FuzzyRuleAntecedent();
EPA_VECB->joinWithAND(EPA, VECB);
FuzzyRuleConsequent* thenVTAV34 = new FuzzyRuleConsequent();
thenVTAV34->addOutput(TDA);
FuzzyRule* fuzzyRule34 = new FuzzyRule(34, EPA_VECB, thenVTAV34);
fuzzy->addFuzzyRule(fuzzyRule34);

// Rule 35: SE o ERRO é EPA E a VARERRO é VECA ENTÃO o VTAV é TDA
FuzzyRuleAntecedent *EPA_VECA = new FuzzyRuleAntecedent();
EPA_VECA->joinWithAND(EPA, VECA);
FuzzyRuleConsequent* thenVTAV35 = new FuzzyRuleConsequent();
thenVTAV35->addOutput(TDA);
FuzzyRule* fuzzyRule35 = new FuzzyRule(35, EPA_VECA, thenVTAV35);
fuzzy->addFuzzyRule(fuzzyRule35);
}

//*****
*****
void loop() {

//PARA FACILITAR FUTURAS CARGAS PINO DE PAUSE
if (!digitalRead(pinPause)) {
    estadoPause();
    return;
}

```



```

    }

    medida = scale.get_units(5); // ativa a balança
    entra=true;
    iteracao=1;

    if (medida > pesoMin){
        Serial.println("Balança acionada. Ativando o controlador...");
        delay(2000);
        medida = scale.get_units(10);//aquisição de novas medidas
(estabilidade da balança)
        digitalWrite(led_sistema,LOW);
        delay(100);
        digitalWrite(led_sistema,HIGH);
        delay(100);
        digitalWrite(led_sistema,LOW);
        delay(100);
        digitalWrite(led_sistema,HIGH);
        delay(100);
        Serial.println("");
    }

Serial.println("*****
*");

    //IMPRIME CONTAGEM DA MEDIDA
    Serial.print("Medida ");
    Serial.print(counter);
    counter += 1;
    delay(100);
    Serial.print(": ");

    //IMPRIME PESO EM GRAMAS
    String medida_string; //serve para corrigir o ponto em vírgula
    medida_string = String(medida*1000,2);
    Serial.print(medida_string);
    Serial.println(" g");

Serial.println("*****
*");

    delay(2000);

    if (entra) {
        peso = medida*1000;
        entradal = peso - Vref;
        anterior = entradal;
        atual = entradal;
        entrada2 = atual-anterior;
        entra = false;
    }

    //Loop de iterações enquanto o ERRO é maior ou menor que o
intervalo "precisao".
    while ((entradal <-precisao)|| (entradal>precisao)){
        fuzzy->setInput(1,entradal);
        fuzzy->setInput(2,entrada2);

        //Imprime os dados de entrada
        Serial.print("Iteracao ");
        Serial.print(iteracao);
        Serial.print("-> ");
        Serial.print("Massa medida: ");

```

```

Serial.println(peso);
Serial.print("Erro: ");
Serial.println(entrada1);
Serial.print("Var_Erro: ");
Serial.println(entrada2);
iteracao += 1;

//Rodando a fuzzificação
fuzzy->fuzzify();

//Rodando a defuzzificação
output = fuzzy->defuzzify(1);

//Imprimindo os resultados
Serial.print("VTAV: ");
Serial.println(output);
Serial.println("-----");

//Guarda a informação do ERRO anterior
anterior = entrada1;

//Cálculo do novo peso reajustado
if (output < 0) {
    peso = peso-abs(output)*0.56623; // constante
    utilizada para calcular variação de tempo de saída em peso
    entrada1 = peso-Vref;
    entrada2 = entrada1-anterior;
} else {
    peso = peso+abs(output)*0.56623; // constante
    utilizada para calcular variação de tempo de saída em peso
    entrada1 = peso-Vref;
    entrada2 = entrada1-anterior;
}
}
}

//*****
//*****
//FUNCAO PARA AGUARDAR ANTES DE INICIAR O SETUP PARA NAO PREJUDICAR FUTURAS
CARGAS
void estadoPause() {
    pinMode(led_pause, OUTPUT);
    digitalWrite(led_pause, HIGH); // turn the LED on (HIGH is the voltage
level)
    delay(500); // wait for a second
    digitalWrite(led_pause, LOW); // turn the LED off by making the
voltage LOW
    delay(200); // wait for a second
}

```

APÊNDICE B – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 193 g

```
15:06:36,361 -> Iniciando o sistema...
15:06:43,722 -> Sistema pronto. Aguardando pesagem.
15:06:55,535 -> Balança acionada. Ativando o controlador...
15:06:58,854 -> Medida 1: 193,00
15:07:00,868 -> *****
15:07:00,940 -> Massa medida: 193,00
15:07:01,843 -> Erro: -9,71
15:07:01,890 -> Var Erro: 0,00
15:07:01,890 -> VTAT: 6,46
15:07:01,890 -> -----
15:07:01,943 -> Massa medida: 196,66
15:07:01,943 -> Erro: -6,05
15:07:01,943 -> Var Erro: 3,66
15:07:01,990 -> VTAT: 0,57
15:07:01,990 -> -----
15:07:01,990 -> Massa medida: 196,98
15:07:02,043 -> Erro: -5,73
15:07:02,043 -> Var Erro: 0,32
15:07:02,043 -> VTAT: 0,50
15:07:02,090 -> -----
15:07:02,090 -> Massa medida: 197,26
15:07:02,144 -> Erro: -5,45
15:07:02,144 -> Var Erro: 0,28
15:07:02,144 -> VTAT: 0,50
15:07:02,144 -> -----
15:07:02,191 -> Massa medida: 197,55
15:07:02,191 -> Erro: -5,16
15:07:02,244 -> Var Erro: 0,28
15:07:02,244 -> VTAT: 0,50
15:07:02,244 -> -----
15:07:02,291 -> Massa medida: 197,83
15:07:02,291 -> Erro: -4,88
15:07:02,291 -> Var Erro: 0,28
15:07:02,344 -> VTAT: 0,45
15:07:02,344 -> -----
15:07:02,344 -> Massa medida: 198,09
15:07:02,391 -> Erro: -4,62
15:07:02,391 -> Var Erro: 0,26
15:07:02,391 -> VTAT: 0,45
15:07:02,444 -> -----
15:07:02,444 -> Massa medida: 198,34
15:07:02,444 -> Erro: -4,37
15:07:02,491 -> Var Erro: 0,25
```

15:07:02,491 -> VTAT: 0,45
15:07:02,491 -> -----
15:07:02,544 -> Massa medida: 198,59
15:07:02,544 -> Erro: -4,12
15:07:02,544 -> Var Erro: 0,25
15:07:02,607 -> VTAT: 0,44
15:07:02,607 -> -----
15:07:02,607 -> Massa medida: 198,84
15:07:02,645 -> Erro: -3,87
15:07:02,645 -> Var Erro: 0,25
15:07:02,645 -> VTAT: 0,45
15:07:02,692 -> -----
15:07:02,692 -> Massa medida: 199,10
15:07:02,745 -> Erro: -3,61
15:07:02,745 -> Var Erro: 0,25
15:07:02,745 -> VTAT: 0,45
15:07:02,745 -> -----
15:07:02,792 -> Massa medida: 199,35
15:07:02,792 -> Erro: -3,36
15:07:02,846 -> Var Erro: 0,25
15:07:02,846 -> VTAT: 0,45
15:07:02,846 -> -----
15:07:02,846 -> Massa medida: 199,61
15:07:02,886 -> Erro: -3,10
15:07:02,886 -> Var Erro: 0,25
15:07:02,946 -> VTAT: 0,45
15:07:02,946 -> -----
15:07:02,946 -> Massa medida: 199,86
15:07:02,993 -> Erro: -2,85
15:07:02,993 -> Var Erro: 0,25
15:07:02,993 -> VTAT: 0,45
15:07:03,046 -> -----
15:07:03,046 -> Massa medida: 200,11
15:07:03,046 -> Erro: -2,60
15:07:03,093 -> Var Erro: 0,25
15:07:03,093 -> VTAT: 0,45
15:07:03,093 -> -----
15:07:03,146 -> Massa medida: 200,37
15:07:03,146 -> Erro: -2,34
15:07:03,146 -> Var Erro: 0,25
15:07:03,193 -> VTAT: 0,45
15:07:03,193 -> -----
15:07:03,193 -> Massa medida: 200,62
15:07:03,247 -> Erro: -2,09
15:07:03,247 -> Var Erro: 0,25
15:07:03,247 -> VTAT: 0,45

15:07:03,293 -> -----
15:07:03,293 -> Massa medida: 200,88
15:07:03,293 -> Erro: -1,83
15:07:03,347 -> Var Erro: 0,25
15:07:03,347 -> VTAT: 0,45
15:07:03,347 -> -----
15:07:03,394 -> Massa medida: 201,13
15:07:03,394 -> Erro: -1,58
15:07:03,394 -> Var Erro: 0,25
15:07:03,447 -> VTAT: 0,45
15:07:03,447 -> -----
15:07:03,447 -> Massa medida: 201,38
15:07:03,494 -> Erro: -1,33
15:07:03,494 -> Var Erro: 0,25
15:07:03,494 -> VTAT: 0,45
15:07:03,547 -> -----
15:07:03,547 -> Massa medida: 201,64
15:07:03,547 -> Erro: -1,07
15:07:03,594 -> Var Erro: 0,25
15:07:03,594 -> VTAT: 0,45
15:07:03,594 -> -----
15:07:03,648 -> Massa medida: 201,89
15:07:03,648 -> Erro: -0,82
15:07:03,648 -> Var Erro: 0,25
15:07:03,696 -> VTAT: 0,26
15:07:03,696 -> -----
15:07:03,696 -> Massa medida: 202,04
15:07:03,748 -> Erro: -0,67
15:07:03,748 -> Var Erro: 0,15
15:07:03,748 -> VTAT: 0,39
15:07:03,795 -> -----
15:07:03,795 -> Massa medida: 202,26
15:07:03,848 -> Erro: -0,45
15:07:03,848 -> Var Erro: 0,22
15:07:03,848 -> VTAT: 0,11
15:07:03,848 -> -----
15:07:03,895 -> Massa medida: 202,32
15:07:03,895 -> Erro: -0,39
15:07:03,948 -> Var Erro: 0,06
15:07:03,948 -> VTAT: 0,11
15:07:03,948 -> -----
15:07:03,995 -> Massa medida: 202,38
15:07:03,995 -> Erro: -0,33
15:07:03,995 -> Var Erro: 0,06
15:07:04,049 -> VTAT: 0,41
15:07:04,049 -> -----

15:07:04,049 -> Massa medida: 202,61
15:07:04,096 -> Erro: -0,10
15:07:04,096 -> Var Erro: 0,23
15:07:04,096 -> VTAT: -0,01
15:07:04,149 -> -----
15:07:04,149 -> Massa medida: 202,61
15:07:04,149 -> Erro: -0,10
15:07:04,196 -> Var Erro: -0,01
15:07:04,196 -> VTAT: 0,04
15:07:04,196 -> -----
15:07:04,249 -> Massa medida: 202,63
15:07:04,249 -> Erro: -0,08
15:07:04,249 -> Var Erro: 0,02
15:07:04,296 -> VTAT: 0,38
15:07:04,296 -> -----
15:07:04,349 -> Massa medida: 202,84
15:07:04,349 -> Erro: 0,13
15:07:04,349 -> Var Erro: 0,22
15:07:04,349 -> VTAT: -0,06
15:07:04,396 -> -----
15:07:04,396 -> Massa medida: 202,81
15:07:04,449 -> Erro: 0,10
15:07:04,449 -> Var Erro: -0,03
15:07:04,449 -> VTAT: -0,05
15:07:04,449 -> -----
15:07:04,496 -> Massa medida: 202,79
15:07:04,496 -> Erro: 0,08
15:07:04,550 -> Var Erro: -0,03
15:07:04,550 -> VTAT: -0,05
15:07:04,550 -> -----
15:07:04,596 -> Massa medida: 202,76
15:07:04,596 -> Erro: 0,05
15:07:04,596 -> Var Erro: -0,03
15:07:04,650 -> VTAT: -0,04
15:07:04,650 -> -----
15:07:04,650 -> Massa medida: 202,74
15:07:04,697 -> Erro: 0,03
15:07:04,697 -> Var Erro: -0,02
15:07:04,697 -> VTAT: -0,04
15:07:04,750 -> -----

APÊNDICE C – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 198 g

```
15:30:25,361 -> Iniciando o sistema...
15:30:32,722 -> Sistema pronto. Aguardando pesagem.
15:30:44,315 -> Balança acionada. Ativando o controlador...
15:30:47,854 -> Medida 1: 198,00
15:30:49,568 -> *****
15:30:49,630 -> Massa medida: 198,00
15:30:50,533 -> Erro: -4,71
15:30:50,533 -> Var Erro: 0,00
15:30:50,579 -> VTAV: 0,50
15:30:50,579 -> -----
15:30:50,579 -> Massa medida: 198,28
15:30:50,633 -> Erro: -4,43
15:30:50,633 -> Var Erro: 0,28
15:30:50,633 -> VTAV: 0,44
15:30:50,680 -> -----
15:30:50,680 -> Massa medida: 198,53
15:30:50,733 -> Erro: -4,18
15:30:50,733 -> Var Erro: 0,25
15:30:50,733 -> VTAV: 0,45
15:30:50,733 -> -----
15:30:50,780 -> Massa medida: 198,79
15:30:50,780 -> Erro: -3,92
15:30:50,834 -> Var Erro: 0,25
15:30:50,834 -> VTAV: 0,45
15:30:50,834 -> -----
15:30:50,881 -> Massa medida: 199,04
15:30:50,881 -> Erro: -3,67
15:30:50,881 -> Var Erro: 0,25
15:30:50,934 -> VTAV: 0,45
15:30:50,934 -> -----
15:30:50,934 -> Massa medida: 199,29
15:30:50,981 -> Erro: -3,42
15:30:50,981 -> Var Erro: 0,25
15:30:50,981 -> VTAV: 0,45
15:30:51,034 -> -----
15:30:51,034 -> Massa medida: 199,55
15:30:51,034 -> Erro: -3,16
15:30:51,081 -> Var Erro: 0,25
15:30:51,081 -> VTAV: 0,45
15:30:51,081 -> -----
15:30:51,134 -> Massa medida: 199,80
15:30:51,134 -> Erro: -2,91
15:30:51,134 -> Var Erro: 0,25
```

15:30:51,181 -> VTAV: 0,45
15:30:51,181 -> -----
15:30:51,181 -> Massa medida: 200,06
15:30:51,235 -> Erro: -2,65
15:30:51,235 -> Var Erro: 0,25
15:30:51,235 -> VTAV: 0,45
15:30:51,281 -> -----
15:30:51,281 -> Massa medida: 200,31
15:30:51,335 -> Erro: -2,40
15:30:51,335 -> Var Erro: 0,25
15:30:51,335 -> VTAV: 0,45
15:30:51,335 -> -----
15:30:51,382 -> Massa medida: 200,56
15:30:51,382 -> Erro: -2,15
15:30:51,382 -> Var Erro: 0,25
15:30:51,435 -> VTAV: 0,45
15:30:51,435 -> -----
15:30:51,435 -> Massa medida: 200,82
15:30:51,482 -> Erro: -1,89
15:30:51,482 -> Var Erro: 0,25
15:30:51,536 -> VTAV: 0,45
15:30:51,536 -> -----
15:30:51,536 -> Massa medida: 201,07
15:30:51,582 -> Erro: -1,64
15:30:51,582 -> Var Erro: 0,25
15:30:51,582 -> VTAV: 0,45
15:30:51,582 -> -----
15:30:51,636 -> Massa medida: 201,32
15:30:51,636 -> Erro: -1,39
15:30:51,684 -> Var Erro: 0,25
15:30:51,684 -> VTAV: 0,45
15:30:51,684 -> -----
15:30:51,736 -> Massa medida: 201,58
15:30:51,736 -> Erro: -1,13
15:30:51,736 -> Var Erro: 0,25
15:30:51,782 -> VTAV: 0,45
15:30:51,782 -> -----
15:30:51,782 -> Massa medida: 201,83
15:30:51,836 -> Erro: -0,88
15:30:51,836 -> Var Erro: 0,25
15:30:51,836 -> VTAV: 0,30
15:30:51,836 -> -----
15:30:51,883 -> Massa medida: 202,00
15:30:51,883 -> Erro: -0,71
15:30:51,937 -> Var Erro: 0,17
15:30:51,937 -> VTAV: 0,38

15:30:51,937 -> -----
15:30:51,983 -> Massa medida: 202,22
15:30:51,983 -> Erro: -0,49
15:30:51,983 -> Var Erro: 0,21
15:30:52,036 -> VTAV: 0,31
15:30:52,036 -> -----
15:30:52,036 -> Massa medida: 202,39
15:30:52,084 -> Erro: -0,32
15:30:52,084 -> Var Erro: 0,17
15:30:52,084 -> VTAV: 0,28
15:30:52,137 -> -----
15:30:52,137 -> Massa medida: 202,55
15:30:52,137 -> Erro: -0,16
15:30:52,184 -> Var Erro: 0,16
15:30:52,184 -> VTAV: 0,17
15:30:52,184 -> -----
15:30:52,237 -> Massa medida: 202,64
15:30:52,237 -> Erro: -0,07
15:30:52,237 -> Var Erro: 0,10
15:30:52,284 -> VTAV: -0,02
15:30:52,284 -> -----
15:30:52,337 -> Massa medida: 202,63
15:30:52,337 -> Erro: -0,08
15:30:52,337 -> Var Erro: -0,01
15:30:52,384 -> VTAV: 0,03
15:30:52,384 -> -----
15:30:52,384 -> Massa medida: 202,65
15:30:52,437 -> Erro: -0,06
15:30:52,437 -> Var Erro: 0,02
15:30:52,437 -> VTAV: 0,38
15:30:52,437 -> -----
15:30:52,484 -> Massa medida: 202,87
15:30:52,484 -> Erro: 0,16
15:30:52,538 -> Var Erro: 0,22
15:30:52,538 -> VTAV: -0,06
15:30:52,538 -> -----
15:30:52,584 -> Massa medida: 202,83
15:30:52,584 -> Erro: 0,12
15:30:52,584 -> Var Erro: -0,03
15:30:52,638 -> VTAV: -0,05
15:30:52,638 -> -----
15:30:52,638 -> Massa medida: 202,80
15:30:52,685 -> Erro: 0,09
15:30:52,685 -> Var Erro: -0,03
15:30:52,685 -> VTAV: -0,05
15:30:52,738 -> -----

15:30:52,738 -> Massa medida: 202,78
15:30:52,738 -> Erro: 0,07
15:30:52,785 -> Var Erro: -0,03
15:30:52,785 -> VTAV: -0,04
15:30:52,785 -> -----
15:30:52,838 -> Massa medida: 202,75
15:30:52,838 -> Erro: 0,04
15:30:52,838 -> Var Erro: -0,02
15:30:52,885 -> VTAV: -0,04
15:30:52,885 -> -----
15:30:52,938 -> Massa medida: 202,73
15:30:52,938 -> Erro: 0,02
15:30:52,938 -> Var Erro: -0,02
15:30:52,985 -> VTAV: -0,04
15:30:52,985 -> -----

APÊNDICE D – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 203 g

```
15:57:26,342 -> Iniciando o sistema...
15:57:33,702 -> Sistema pronto. Aguardando pesagem.
15:57:45,533 -> Balança acionada. Ativando o controlador...
15:57:48,854 -> Medida 1: 203,00
15:57:50,168 -> *****
16:57:50,285 -> Massa medida: 203,00
16:57:51,188 -> Erro: 0,29
16:57:51,188 -> Var Erro: 0,00
16:57:51,235 -> VTAV: -0,09
16:57:51,235 -> -----
16:57:51,289 -> Massa medida: 202,95
16:57:51,289 -> Erro: 0,24
16:57:51,289 -> Var Erro: -0,05
16:57:51,289 -> VTAV: -0,07
16:57:51,336 -> -----
16:57:51,336 -> Massa medida: 202,91
16:57:51,389 -> Erro: 0,20
16:57:51,389 -> Var Erro: -0,04
16:57:51,389 -> VTAV: -0,06
16:57:51,389 -> -----
16:57:51,436 -> Massa medida: 202,88
16:57:51,436 -> Erro: 0,17
16:57:51,489 -> Var Erro: -0,04
16:57:51,489 -> VTAV: -0,06
16:57:51,489 -> -----
16:57:51,536 -> Massa medida: 202,84
16:57:51,536 -> Erro: 0,13
16:57:51,536 -> Var Erro: -0,03
16:57:51,589 -> VTAV: -0,05
16:57:51,589 -> -----
16:57:51,589 -> Massa medida: 202,81
16:57:51,636 -> Erro: 0,10
16:57:51,636 -> Var Erro: -0,03
16:57:51,636 -> VTAV: -0,05
16:57:51,670 -> -----
16:57:51,670 -> Massa medida: 202,79
16:57:51,704 -> Erro: 0,08
16:57:51,704 -> Var Erro: -0,03
16:57:51,752 -> VTAV: -0,05
16:57:51,752 -> -----
16:57:51,789 -> Massa medida: 202,76
16:57:51,789 -> Erro: 0,05
16:57:51,789 -> Var Erro: -0,03
```

16:57:51,836 -> VTAV: -0,04
16:57:51,836 -> -----
16:57:51,889 -> Massa medida: 202,74
16:57:51,889 -> Erro: 0,03
16:57:51,889 -> Var Erro: -0,02
16:57:51,936 -> VTAV: -0,04
16:57:51,936 -> -----

APÊNDICE E – RESULTADO DA SIMULAÇÃO PARA MASSA MEDIDA DE 212 g

```
16:44:16,366 -> Iniciando o sistema...
16:44:23,720 -> Sistema pronto. Aguardando pesagem.
16:44:35,527 -> Balança acionada. Ativando o controlador...
16:44:38,544 -> Medida 1: 212,00
16:44:40,165 -> *****
16:44:40,283 -> Massa medida: 212,00
16:44:41,186 -> Erro: 9,29
16:44:41,186 -> Var Erro: 0,00
16:44:41,233 -> VTAV: -5,79
16:44:41,233 -> -----
16:44:41,233 -> Massa medida: 208,72
16:44:41,287 -> Erro: 6,01
16:44:41,287 -> Var Erro: -3,28
16:44:41,287 -> VTAV: -0,50
16:44:41,334 -> -----
16:44:41,334 -> Massa medida: 208,44
16:44:41,387 -> Erro: 5,73
16:44:41,387 -> Var Erro: -0,28
16:44:41,387 -> VTAV: -0,50
16:44:41,387 -> -----
16:44:41,434 -> Massa medida: 208,15
16:44:41,434 -> Erro: 5,44
16:44:41,488 -> Var Erro: -0,28
16:44:41,488 -> VTAV: -0,50
16:44:41,488 -> -----
16:44:41,535 -> Massa medida: 207,87
16:44:41,535 -> Erro: 5,16
16:44:41,535 -> Var Erro: -0,28
16:44:41,588 -> VTAV: -0,50
16:44:41,588 -> -----
16:44:41,588 -> Massa medida: 207,59
16:44:41,635 -> Erro: 4,88
16:44:41,635 -> Var Erro: -0,28
16:44:41,635 -> VTAV: -0,45
16:44:41,669 -> -----
16:44:41,669 -> Massa medida: 207,33
16:44:41,704 -> Erro: 4,62
16:44:41,704 -> Var Erro: -0,25
16:44:41,751 -> VTAV: -0,44
16:44:41,751 -> -----
16:44:41,751 -> Massa medida: 207,08
16:44:41,788 -> Erro: 4,37
16:44:41,788 -> Var Erro: -0,25
```

16:44:41,835 -> VTAV: -0,44
16:44:41,835 -> -----
16:44:41,889 -> Massa medida: 206,83
16:44:41,889 -> Erro: 4,12
16:44:41,889 -> Var Erro: -0,25
16:44:41,889 -> VTAV: -0,43
16:44:41,936 -> -----
16:44:41,936 -> Massa medida: 206,59
16:44:41,989 -> Erro: 3,88
16:44:41,989 -> Var Erro: -0,25
16:44:41,989 -> VTAV: -0,44
16:44:42,036 -> -----
16:44:42,036 -> Massa medida: 206,34
16:44:42,036 -> Erro: 3,63
16:44:42,089 -> Var Erro: -0,25
16:44:42,089 -> VTAV: -0,45
16:44:42,089 -> -----
16:44:42,136 -> Massa medida: 206,09
16:44:42,136 -> Erro: 3,38
16:44:42,136 -> Var Erro: -0,25
16:44:42,189 -> VTAV: -0,45
16:44:42,189 -> -----
16:44:42,189 -> Massa medida: 205,83
16:44:42,236 -> Erro: 3,12
16:44:42,236 -> Var Erro: -0,25
16:44:42,236 -> VTAV: -0,45
16:44:42,290 -> -----
16:44:42,290 -> Massa medida: 205,58
16:44:42,290 -> Erro: 2,87
16:44:42,337 -> Var Erro: -0,26
16:44:42,337 -> VTAV: -0,45
16:44:42,337 -> -----
16:44:42,390 -> Massa medida: 205,32
16:44:42,390 -> Erro: 2,61
16:44:42,390 -> Var Erro: -0,26
16:44:42,437 -> VTAV: -0,45
16:44:42,437 -> -----
16:44:42,490 -> Massa medida: 205,06
16:44:42,490 -> Erro: 2,35
16:44:42,490 -> Var Erro: -0,26
16:44:42,537 -> VTAV: -0,45
16:44:42,537 -> -----
16:44:42,537 -> Massa medida: 204,81
16:44:42,590 -> Erro: 2,10
16:44:42,590 -> Var Erro: -0,26
16:44:42,590 -> VTAV: -0,45

16:44:42,637 -> -----
16:44:42,637 -> Massa medida: 204,55
16:44:42,637 -> Erro: 1,84
16:44:42,690 -> Var Erro: -0,26
16:44:42,690 -> VTAV: -0,45
16:44:42,690 -> -----
16:44:42,737 -> Massa medida: 204,30
16:44:42,737 -> Erro: 1,59
16:44:42,737 -> Var Erro: -0,26
16:44:42,790 -> VTAV: -0,45
16:44:42,790 -> -----
16:44:42,790 -> Massa medida: 204,04
16:44:42,837 -> Erro: 1,33
16:44:42,837 -> Var Erro: -0,25
16:44:42,891 -> VTAV: -0,44
16:44:42,891 -> -----
16:44:42,891 -> Massa medida: 203,79
16:44:42,937 -> Erro: 1,08
16:44:42,937 -> Var Erro: -0,25
16:44:42,937 -> VTAV: -0,44
16:44:42,937 -> -----
16:44:42,991 -> Massa medida: 203,54
16:44:42,991 -> Erro: 0,83
16:44:43,037 -> Var Erro: -0,25
16:44:43,037 -> VTAV: -0,35
16:44:43,037 -> -----
16:44:43,091 -> Massa medida: 203,34
16:44:43,091 -> Erro: 0,63
16:44:43,091 -> Var Erro: -0,20
16:44:43,137 -> VTAV: -0,15
16:44:43,137 -> -----
16:44:43,137 -> Massa medida: 203,26
16:44:43,191 -> Erro: 0,55
16:44:43,191 -> Var Erro: -0,08
16:44:43,191 -> VTAV: -0,15
16:44:43,238 -> -----
16:44:43,238 -> Massa medida: 203,18
16:44:43,291 -> Erro: 0,47
16:44:43,291 -> Var Erro: -0,08
16:44:43,291 -> VTAV: -0,12
16:44:43,291 -> -----
16:44:43,338 -> Massa medida: 203,11
16:44:43,338 -> Erro: 0,40
16:44:43,338 -> Var Erro: -0,07
16:44:43,391 -> VTAV: -0,10
16:44:43,391 -> -----

16:44:43,391 -> Massa medida: 203,06
16:44:43,439 -> Erro: 0,35
16:44:43,439 -> Var Erro: -0,06
16:44:43,491 -> VTAV: -0,09
16:44:43,491 -> -----
16:44:43,491 -> Massa medida: 203,00
16:44:43,538 -> Erro: 0,29
16:44:43,538 -> Var Erro: -0,05
16:44:43,538 -> VTAV: -0,08
16:44:43,592 -> -----
16:44:43,592 -> Massa medida: 202,96
16:44:43,592 -> Erro: 0,25
16:44:43,639 -> Var Erro: -0,04
16:44:43,639 -> VTAV: -0,07
16:44:43,639 -> -----
16:44:43,692 -> Massa medida: 202,92
16:44:43,692 -> Erro: 0,21
16:44:43,692 -> Var Erro: -0,04
16:44:43,739 -> VTAV: -0,06
16:44:43,739 -> -----
16:44:43,739 -> Massa medida: 202,88
16:44:43,792 -> Erro: 0,17
16:44:43,792 -> Var Erro: -0,04
16:44:43,792 -> VTAV: -0,06
16:44:43,839 -> -----
16:44:43,839 -> Massa medida: 202,85
16:44:43,893 -> Erro: 0,14
16:44:43,893 -> Var Erro: -0,03
16:44:43,893 -> VTAV: -0,05
16:44:43,893 -> -----
16:44:43,940 -> Massa medida: 202,82
16:44:43,940 -> Erro: 0,11
16:44:43,993 -> Var Erro: -0,03
16:44:43,993 -> VTAV: -0,05
16:44:43,993 -> -----
16:44:43,993 -> Massa medida: 202,79
16:44:44,040 -> Erro: 0,08
16:44:44,040 -> Var Erro: -0,03
16:44:44,093 -> VTAV: -0,05
16:44:44,093 -> -----
16:44:44,093 -> Massa medida: 202,77
16:44:44,140 -> Erro: 0,06
16:44:44,140 -> Var Erro: -0,03
16:44:44,140 -> VTAV: -0,04
16:44:44,193 -> -----
16:44:44,193 -> Massa medida: 202,74

16:44:44,193 -> Erro: 0,03
16:44:44,240 -> Var Erro: -0,02
16:44:44,240 -> VTAV: -0,04
16:44:44,240 -> -----

APÊNDICE F – CÓDIGO PARA COMUNICAR O ARDUINO LEONARDO COM A BALANÇA E O MICROSOFT EXCEL

```

// MONITORAMENTO AUTOMÁTICO DE UM PROCESSO DE ENVASE UTILIZANDO CONTROLADOR
FUZZY EMBARCADO EM HARDWARE DE BAIXO CUSTO.
//Carlos A. M. Gyori, Marcela A. G. M. de Freitas e Paloma M. S. Rocha
Rizol
//Universidade Estadual Paulista (UNESP) "Júlio de Mesquita Filho", Campus
de Guaratinguetá, Brasil.
//{carlos.gyori, marcela.freitas, paloma.rizol}@unesp.br

//*****EXCEL*****
//OBJETO PARA EMULADOR DE TECLADO
#include <Keyboard.h>

//*****BALANÇA*****
// OBJETO PARA MÓDULO DA BALANÇA
#include <HX711.h>

// DEFINIÇÕES DE PINOS
#define pinDT 4 //HX711
#define pinSCK 5 //HX711
#define led_sistema 6 //verde
#define led_balanca 7 //azul
#define led_pause 8 //vermelho
#define pinPause 12 //pino de pause

// DEFINIÇÕES DA BALANÇA
#define pesoMin 0.05
#define pesoMax 300.0
#define escala 343000.0f

//Instanciando o objeto balança
HX711 scale;

// DECLARAÇÃO DE VARIÁVEIS
float medida=0; //balança
float medida1=0; //balança
int counter = 1; //contador de medições

//DECLARACAO DAS FUNCOES P/ ENVIAR AO EXCEL
void estadoPause();
void erroCritico();
void teclaEnter();
void teclaTAB();

//EXECUTADO QUANDO O ARDUINO FOR LIGADO
void setup() {
  Serial.begin(57600);

//INICIO DOS PINOS DO SISTEMA
  pinMode(pinPause, INPUT_PULLUP);
  pinMode(led_sistema,OUTPUT);
  pinMode(led_balanca,OUTPUT);
  pinMode(led_pause,OUTPUT);
  delay(1000);
  digitalWrite(led_sistema,HIGH);
  delay(1000);
  digitalWrite(led_balanca,HIGH);
  delay(1000);

```

```

digitalWrite(led_pause,HIGH);
delay(1000);
digitalWrite(led_sistema,LOW);
digitalWrite(led_balanca,LOW);
digitalWrite(led_pause,LOW);

//INICIO DA BALANÇA
scale.begin(pinDT, pinSCK); // CONFIGURANDO OS PINOS DA BALANÇA
scale.set_scale(escala); // ENVIANDO O VALOR DA ESCALA CALIBRADO
digitalWrite(led_sistema,HIGH);
delay(200);
digitalWrite(led_sistema,LOW);
delay(200);
digitalWrite(led_sistema,HIGH);
delay(200);
digitalWrite(led_sistema,LOW);
delay(200);

//INICIO DO EMULADOR DE TECLADO (EXCEL)
Keyboard.begin();

digitalWrite(led_sistema,HIGH); //LIBERADO PARA REALIZAR A TARA
digitalWrite(led_balanca,HIGH);
scale.power_up(); // LIGANDO O SENSOR DA BALANÇA
digitalWrite(led_sistema,LOW);
delay(200);
digitalWrite(led_balanca,LOW);
delay(200);

medida = scale.get_units(10); // SALVANDO NA VARIÁVEL O VALOR DA MÉDIA DE
10 MEDIDAS //TARA DA BALANÇA
digitalWrite(led_sistema,HIGH);
scale.tare(); // ZERA A BALANÇA CASO A MASSA SEJA MENOR QUE O VALOR MIN
medida = 0;
digitalWrite(led_sistema,LOW);
delay(500);
digitalWrite(led_sistema,HIGH);
}

void loop() {

medida = scale.get_units(5); // ativa a balança

//PARA FACILITAR FUTURAS CARGAS PINO DE PAUSE
if (!digitalRead(pinPause)) {
estadoPause();
return;
}

if (medida > pesoMin){
delay(1000);
medida = scale.get_units(10);//realiza 10 medidas
digitalWrite(led_sistema,LOW);
delay(100);
digitalWrite(led_sistema,HIGH);
delay(100);
digitalWrite(led_sistema,LOW);
delay(100);
digitalWrite(led_sistema,HIGH);
delay(100);
}
}

```

```

//IMPRIME CONTAGEM DA MEDIDA
Keyboard.print("Medida ");
Keyboard.print(counter);
counter += 1;
delay(100);
teclaTAB();

//IMPRIME MASSA EM GRAMAS
String medida_string; //serve para corrigir o ponto em vírgula
medida_string = String(medida*1000,2);
medida_string.replace(".",",");
Keyboard.print(medida_string); // Envia a medida em gramas
delay(2000);
teclaTAB();
delay(100);
teclaEnter();
delay(1000);
}
}

//1.FUNCAO PARA AGUARDAR ANTES DE INICIAR O SETUP PARA NAO PROJUDICAR
FUTURAS CARGAS
void estadoPause() {
  pinMode(led_pause, OUTPUT);
  digitalWrite(led_pause, HIGH); // turn the LED on (HIGH is the voltage
level)
  delay(500); // wait for a second
  digitalWrite(led_pause, LOW); // turn the LED off by making the
voltage LOW
  delay(200); // wait for a second
}

//2.FUNCAO PARA AVISO DE ERRO CRITICO ATRAVES DO LED INTERNO DO ARDUINO
void erroCritico() {
  pinMode(LED_BUILTIN, OUTPUT);
  while (true) {
    digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the
voltage level)
    delay(100); // wait for a second
    digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the
voltage LOW
    delay(100); // wait for a second
  }
}

//3.FUNCAO PARA ENVIAR O APERTO DE UM ENTER
void teclaEnter() {
  Keyboard.press(KEY_RETURN); // PRESSIONA O ENTER
  delay(50); // ESPERA 50 MILISSEGUNDOS
  Keyboard.release(KEY_RETURN); // SOLTA O ENTER
}

//4.FUNCAO PARA ENVIAR O APERTO DE UM TAB
void teclaTAB() {
  Keyboard.press(KEY_TAB); // PRESSIONA O TAB
  delay(50); // ESPERA 50 MILISSEGUNDOS
  Keyboard.release(KEY_TAB); // SOLTA O TAB
}
}

```