INSTITUTO FEDERAL DE CIÊNCIA E TECNOLOGIA DE PERNAMBUCO

Campus Garanhuns

Bacharelado em Engenharia Elétrica

ARIOSVALDO ZACARIAS DE BARROS FILHO

APLICAÇÃO DE MÉTODOS NUMÉRICOS PARA O AUMENTO DA EFICIÊNCIA ENERGÉTICA DO CONTROLADOR DE TEMPERATURA DE UMA ESTUFA

Garanhuns - PE

2022

ARIOSVALDO ZACARIAS DE BARROS FILHO

APLICAÇÃO DE MÉTODOS NUMÉRICOS PARA O AUMENTO DA EFICIÊNCIA ENERGÉTICA DO CONTROLADOR DE TEMPERATURA DE UMA ESTUFA

Trabalho de conclusão de curso apresentado a Coordenação do Curso Superior Bacharelado em Engenharia Elétrica do Instituto Federal de Ciência e Tecnologia de Pernambuco, como requisito para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Me. Robson Dias Ramalho

Garanhuns - PE

2022

B277a Barros Filho, Ariosvaldo Zacarias de.
Aplicação de métodos numéricos para o aumento da eficiência energética do controlador de temperatura de uma estufa / Ariosvaldo Zacarias de Barros Filho ; orientador Robson Dias Ramalho, 2022.
97 f. : il.

Orientador: Robson Dias Ramalho.
Trabalho de Conclusão de Curso (Graduação) – Instituto Federal de Pernambuco. Pró-Reitoria de Ensino. Diretoria de Ensino. Campus Garanhuns. Coordenação do Curso Superior em Engenharia. Curso de Bacharelado em Engenharia Elétrica, 2022.

1. Controle de temperatura – Modelos matemáticos. 2. Controladores elétricos. 3. Energia elétrica - Conservação. I. Título.

CDD 536.5

Riane Melo de Freitas Alves –CRB4/1897

APLICAÇÃO DE MÉTODOS NUMÉRICOS PARA O AUMENTO DA EFICIÊNCIA ENERGÉTICA DO CONTROLADOR DE TEMPERATURA DE UMA ESTUFA

Trabalho aprovado. Garanhuns, 20/12/2022.

_____

Docente-Orientador: Prof. Me. Robson Dias Ramalho

_____

Examinador 1: Prof. Dr. Diego Soares Lopes

_____

Examinador 2: Profa. Dra. Danubia Soares Pires

Garanhuns - PE

2022

AGRADECIMENTOS

Agradeço a Deus por ter me ajudado chegar até aqui.

“Mas buscai primeiro o seu reino e a sua justiça, e todas estas coisas vos serão acrescentadas.”

(Mateus 6:33)

RESUMO

Este trabalho apresenta alguns tipos de controladores utilizados para realizar o controle de temperatura de um protótipo de estufa. O protótipo foi disponibilizado pelo Laboratório de Pneumática e Supervisórios do IFPE *Campus* Garanhuns, de forma que foi possível comparar o desempenho dos controladores, sob a ótica da eficiência energética e estresse mecânico dos atuadores que participam da variação de temperatura da estufa. Foram utilizados dois atuadores, a lâmpada halógena para realizar o aquecimento e uma ventoinha para fazer a dissipação de calor. É aplicado um controlador cujo funcionamento é baseado nos métodos numéricos, de modo que seja capaz de rastrear a potência ideal da lâmpada a fim de manter a temperatura no nível desejado. Para coletar os dados de temperatura, um sensor envia os dados para um microcontrolador, fazendo então a regulação da tensão através de um módulo eletrônico (*Dimmer*). Este componente regula a potência da lâmpada, permitindo assim manter a temperatura desejada na estufa utilizando o mínimo de energia. Este sistema, além de apresentar um menor tempo de resposta, tanto no regime estacionário como no permanente, pode se adaptar a cenários onde há distúrbios externos.

Palavras-chave: Controlador de temperatura, métodos numéricos, eficiência energética, estufa.

ABSTRACT

This work presents some types of controllers used to control the temperature of a greenhouse prototype. The prototype was made available by the Pneumatics and Supervisory Laboratory of the IFPE Campus Garanhuns, so that it was possible to compare the performance of the controllers, from the perspective of energy efficiency and mechanical stress of the actuators that participate in the temperature variation of the greenhouse. Two actuators were used, the halogen lamp for heating and a fan for heat dissipation. A controller is applied whose operation is based on numerical methods, so that it is able to track the ideal power of the lamp in order to maintain the temperature at the desired level. To collect temperature data, a sensor sends the data to a microcontroller, then regulating the voltage through an electronic module (*Dimmer*). This component regulates the power of the lamp, thus allowing to maintain the desired temperature in the stove using a minimum of energy. This system, in addition to presenting a shorter response time, both in stationary and permanent regimes, can adapt to scenarios where there are external disturbances.

Keywords: Temperature controller, numerical methods, energy efficiency, greenhouse.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

α	Ângulo de disparo (Radiano)
AA	Ângulo Atual
AMa	Ângulo Máximo
AMi	Ângulo Mínimo
ATX	Advanced Technology Extended
CA	Corrente Alternada
CC	Corrente Contínua
f	Frequência
FP	Fator de potência
FT	Função de transferência
FTs	Funções de transferências
G(s)	Representação de um sistema
I	Corrente
IDE	Integrated Development Environment
K	Ganho de um sistema
Kd	Ganho derivativo
Ki	Ganho integrativo
Kp	Ganho proporcional
NA	Normalmente Aberto
NF	Normalmente Fechado
P	Potência
PD	Proporcional Derivativo
PI	Proporcional Integral
PID	Proporcional Integral Derivativo
PWM	Pulse Width Modulation (Modulação por largura de pulso)
R	Resistência
τ	Constante de tempo
Td	Tempo derivativo
Ti	Tempo integrativo
TRIAC	triode for alternating current (triodo de corrente alternada)
U(s)	Sinal de Entrada de um sistema

V	Tensão
Y(s)	Sinal de saída de um sistema

SUMÁRIO

1 INTRODUÇÃO

Este capítulo aborda os conceitos de controle, automação e eficiência energética, é apresentada a justificativa que motivou a criação do trabalho, além dos objetivos gerais e específicos do trabalho.

1.1 Uma visão sobre controle, automação e eficiência energética

Em concordância com Junior (2019), com a modernização dos processos industriais, houve a necessidade da implantação dos modelos automáticos, permitindo que a intervenção humana fosse nula ou mínima, através da integração de sensores e controladores.

Segundo Aguirre (2007), um dos principais mecanismos para aumentar a eficiência dos sistemas industriais é a utilização dos modelos matemáticos para a construção de um sistema de controle. Isso permite a otimização nos processos, além de garantir que o sistema funcione com um maior grau de confiabilidade, visto que a resposta do controle realizado por um ser humano apresenta maior probabilidade de erros, se comparado com um controlador baseado na eletrônica ou métodos computacionais.

De acordo com Dorf (2001), um sistema de controle é a integração de dispositivos que, quando acionados em conjunto, fornecerá uma resposta de saída ao sistema. De modo geral, esses dispositivos podem ser definidos como: Sensor, atuador e controlador, onde sensor fornecerá a resposta em tempo real de determinado sistema, de modo que o controlador fará a leitura do sinal do sensor e baseado na sua programação interna, acionará o atuador de modo a alterar o estado atual da planta.

Conforme Ogata (2004), um sistema de controle é fundamental para qualquer área da engenharia e indispensável para sistemas que utilizam a robótica e qualquer processo industrial que envolva o controle de variáveis como: temperatura, pressão, umidade, viscosidade, vazão e qualquer parâmetro que possa ser medido por um sensor eletrônico.

Na implementação de um sistema automático, busca-se reduzir os custos energéticos do processo a partir do controle dos componentes do sistema, além de garantir uma resposta mais rápida ao sistema.

1.2 Motivação

Com a modernização de máquinas e outros dispositivos elétricos, além de processos industriais, há uma tendência de otimização da eficiência energética, visto que os gastos com energia elétrica de uma indústria apresentam custos expressivos.

Desta forma, este trabalho busca propor um método alternativo na construção de um controlador utilizando métodos numéricos aplicado ao controle de uma estufa. Assim, o controlador de temperatura será capaz de manter a energia térmica dentro da estufa, com alta estabilidade e baixo consumo energético.

Além do controlador rastreador, este trabalho apresentará os métodos de construção de controladores clássicos que são amplamente utilizados em processos industriais e residenciais, como o controlador *On-Off*, *On-Off* com intervalo diferencial e o controlador PI. Com isso, qualquer pessoa que tenha um sistema com um microcontrolador, sensor e atuador, poderá utilizar este trabalho como guia para a réplica ou ampliação do trabalho na área de controle e automação.

1.3 Objetivos

1.3.1 Objetivo geral

Desenvolver um controlador de temperatura baseado nos métodos numéricos, de modo que apresente uma alta eficiência energética, observando e comparando o comportamento de outros controladores clássicos e analisando o desempenho individual de cada controlador.

1.3.2 Objetivos específicos

- Realizar a revisão bibliográfica referente aos controladores empregados em sistemas onde há o controle de temperatura.
- Implementar os controladores: duas posições, duas posições com intervalo diferencial, PI e o controlador rastreador baseado nos métodos numéricos, através de códigos desenvolvidos na IDE do *Arduino*.
- Testar cada um dos controladores separadamente.
- Obter os dados referentes ao comportamento da temperatura da estufa em relação ao tempo, de cada um dos controladores.
- Comparar a resposta de cada um dos controladores construídos com o controlador rastreador, através dos dados de temperatura dos controladores.

1.4 Estrutura do trabalho

A estrutura do trabalho é baseada em cinco capítulos.

No capítulo 1 foram abordados os conceitos de controle, automação e eficiência energética, é apresentado a justificativa que motivou a criação do trabalho, além dos objetivos gerais e específicos do trabalho.

No capítulo 2 são apresentadas as partes que compõem a estufa, assim como também a identificação das suas características e suas funções.

No capítulo 3 é discutido sobre os sistemas de controle, com aprofundamento na malha aberta e malha fechada.

No capítulo 4 é detalhada a metodologia que foi utilizada no desenvolvimento do projeto para a criação dos controladores além de como foram realizados os ensaios de cada controlador.

No capítulo 5 são apresentados os resultados obtidos com cada controlador, analisado os dados obtidos e feita a comparação entre eles.

No capítulo 6 são apresentadas a conclusão do trabalho e sugestões para trabalhos futuros.

2 COMPONENTES FÍSICOS DA ESTUFA

Este capítulo apresenta as partes que compõem a estufa, assim como a identificação das suas principais características e suas funções.

2.1 Estrutura da estufa

Em concordância com Knight (2009), sabe-se que o calor é a energia térmica transferida entre a matéria, esta energia pode ser dissipada por três formas: condução, radiação e convecção. Deste modo, a estrutura da estufa busca reduzir os três tipos de dissipação.

A estrutura é construída em madeira que funciona como um isolante térmico, visto que sua baixa condutividade térmica faz com que a temperatura externa não cause alterações na temperatura interna de forma significativa, tornando possível reduzir a dissipação do calor por condução.

O papel alumínio que reveste a estrutura interna da estufa faz com que a radiação emitida pela lâmpada se mantenha dentro da estufa, diminuindo as perdas térmicas pelo aquecimento do compensado de madeira.

Segundo Kittle (1973), a convecção é o movimento de subida ou descida de um fluido devido a variação de sua densidade, ocasionado pela agitação das moléculas do fluído por meio da variação de temperatura. Considerando que a estufa seja operada com a tampa fechada, a perda por convecção é reduzida.

É importante ressaltar que a estufa apresenta algumas frestas entre a tampa e a caixa e na parte superior onde se situa a ventoinha, estas aberturas permitem que uma pequena parcela de energia térmica seja dissipada de forma não controlada. Todavia, essa perda é fundamental para simular os distúrbios e perdas de um sistema real, de modo que o nosso controlador seja capaz de lidar com esses ruídos.

Na parte externa da estufa, há um espaço reservado para o circuito responsável pela obtenção, processamento e envio de dados, de modo que é necessário a conexão elétrica do microcontrolador ao computador através de um cabo USB, como também do relé a fonte ATX.

Na figura 1 pode-se visualizar a estrutura externa da estufa, baseada em uma caixa de compensado de madeira com espessura de 1,4 cm, comprimento de 36,6 cm, largura de 21,3 cm, altura de 21,3 cm e o volume interno de 18,94 litros. Na figura 2 nota-se a estrutura interna da estufa, com revestimento de alumínio para a redução da perda da energia térmica via radiação. Na figura 3 é possível ver a parte da estufa responsável pela alocação dos componentes eletrônicos que farão o envio

de sinais para o acionamento dos atuadores e o recebimento dos dados de temperatura.

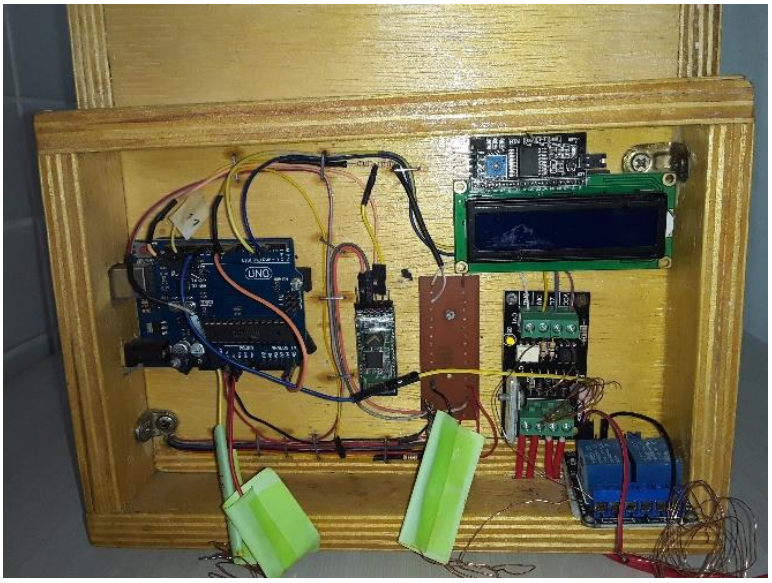
Figura 1 - Estrutura externa da estufa.

Fonte: autoria própria (2022).

Figura 2 - Estrutura interna da estufa.

Fonte: autoria própria (2022).

Figura 3 - Circuito responsável pelo controle da temperatura interna da estufa.



Fonte: autoria própria (2022).

2.2 Arduino UNO

Segundo McRoberts (2018, p. 22) “Um *Arduino* é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele”. Do ponto de vista da sua estrutura, o *Arduino* é uma placa de circuito montada que utiliza componentes eletrônicos, microcontroladores como o ATmega8 ou o ATmega168 e memória *flash* em sua composição. Tal dispositivo é de baixo custo e possui código aberto, permitindo desenvolver e testar projetos com mais facilidade.

Com o *Arduino* (*Hardware*) e a IDE *Arduino* (*Software*) é possível realizar a construção de um sistema baseado na lógica e em algoritmos, de modo que é possível receber e enviar dados.

Neste projeto, a entrada utilizou os pinos digitais, a este pino foi ligado o sensor de temperatura DS18B20, o qual receberá os dados via sinais de modulação por largura de pulso, também conhecido por PWM (*Pulse-Width Modulation*). As saídas também são digitais, as quais fazem o acionamento do módulo *Dimmer* e do relé. O *Arduino* também será responsável por integrar o sensor e os atuadores com a programação realizada através do software IDE *Arduino*.

O *Arduino* é responsável pela transformação dos sinais digitais do sensor de temperatura para graus celsius (ºC). Através de uma biblioteca de códigos tem-se a conversão do valor da tensão ou corrente emitida pelo sensor em valores que sejam possíveis a compreensão dos dados.

Através da lógica construída, baseada nos controladores propostos, tem-se o acionamento dos atuadores do módulo *Dimmer* que é responsável pelo acionamento

da lâmpada conforme a potência pré-estabelecida no ângulo de disparo do triodo de corrente alternada (TRIAC), já o relé é responsável pela conexão elétrica da ventoinha com a fonte de alimentação.

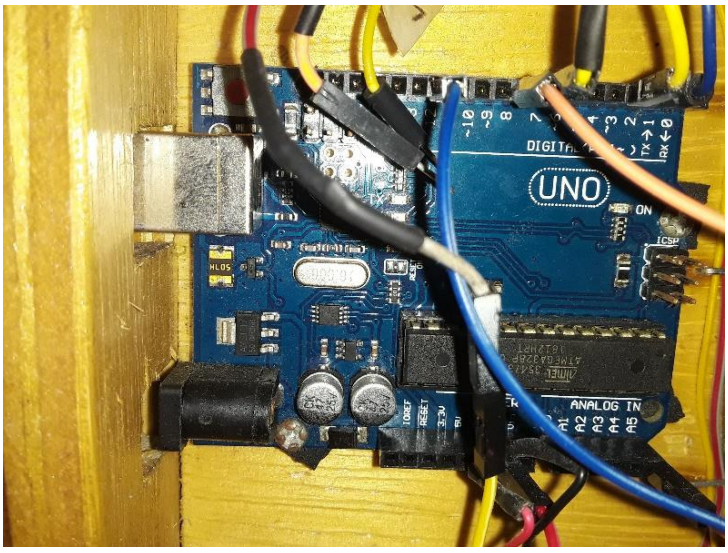
O modelo utilizado no projeto foi o *Arduino* UNO R3, onde as especificações técnicas podem ser observadas na tabela 1.

Tabela 1 - Especificações técnicas do Arduino UNO R3.

Especificações técnicas	Tipo ou valor
Microcontrolador	ATmega328P
Tensão de operação	5 V
Tensão de entrada (recomendada)	7 - 12 V
Tensão de entrada (limite)	6 - 20 V
Pinos digitais de entrada e saída	14 (6 suportam PWM)
Pinos digitais I/O PWM	6
Pinos de entrada analógica	6
Corrente contínua por pino I/O	20 mA
Corrente contínua por pino 3,3 V	50 mA
Memória Flash	32 kB (ATmega328P)
SRAM	1 kB (ATmega328P)
EEPROM	16 Mhz
Velocidade de Clock	16Mhz
Dimensões	68,6 x 53,4 mm
Peso	25 g

Fonte: Arduino (2022).

Figura 4 - Arduino UNO.



Fonte: autoria própria (2022).

2.3 Arduino IDE

De acordo com Monk (2016), o *Arduino* de Ambiente de Desenvolvimento Integrado, do inglês *Integrated Development Environment* (IDE) é o *Software* de código aberto, responsável pela verificação e compilação dos códigos desenvolvidos e a integração com a placa do *Arduino*.

Em concordância com Banzi (2015), a linguagem utilizada é baseada em funções de C e C++. Como o sistema é de código aberto, há inúmeras bibliotecas desenvolvidas com diversas aplicações. As bibliotecas são códigos com aplicações específicas, desenvolvidos por qualquer pessoa e disponibilizados na internet, com isso é possível facilitar a programação de diversos projetos, apenas importando tais bibliotecas.

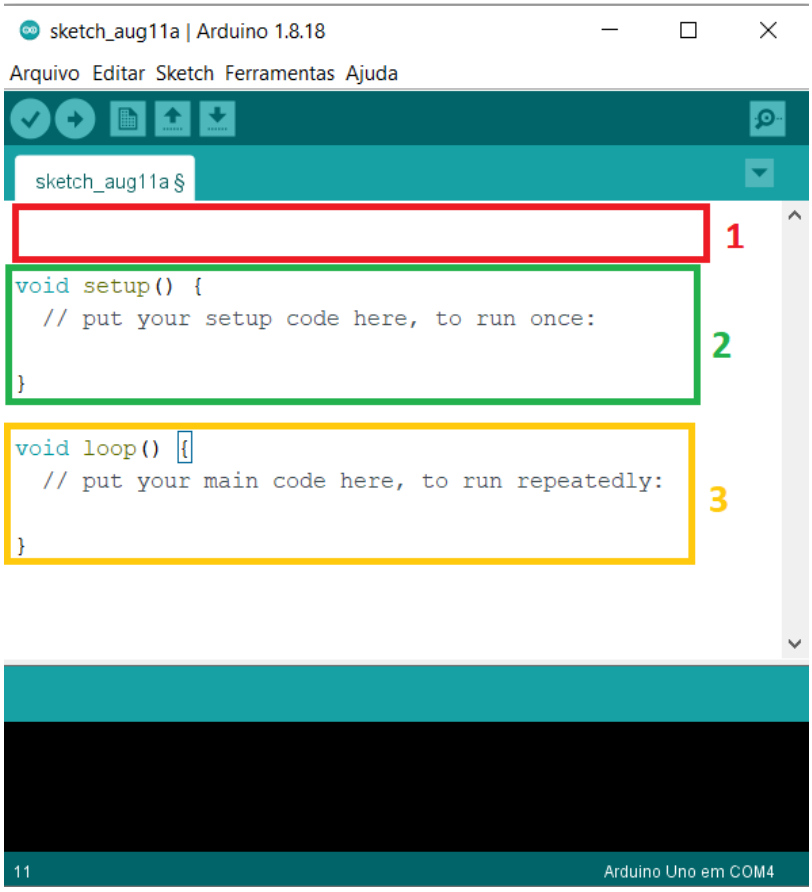
O *Arduino* IDE é capaz de realizar a verificação, através de uma função que analisa se há algum tipo de erro na escrita do código, é importante ressaltar que a verificação não analisará erros do ponto de vista da lógica do programa. A compilação converte o código escrito em C para uma linguagem em que o processador é capaz de executar os comandos.

Segundo McRoberts (2018), o IDE pode ser dividido em três partes: Na parte superior a *Toolbar* é responsável pela configuração do sistema, no centro fica o código ou *Sketch Window* responsável por fazer a escrita do código, na parte inferior fica a janela de mensagens responsável pela a apresentação dos erros outro tipo de notificação.

Como mostrado na figura 5, dentro do código, podemos dividir em três partes.

1. Área onde se deve colocar as variáveis globais, bibliotecas e funções.
2. Dentro da função *void setup*, colocamos os códigos que serão lidos apenas uma vez pela pelo *Arduino* IDE, também é nesta área que que colocamos as funções referentes a conexão serial do *Arduino* com o computador.
3. Dentro da função *void loop*, estarão os códigos que serão executados de forma contínua e infinitamente na forma de um loop, enquanto o *Arduino* estiver energizado.

Figura 5 - IDE Arduino.



Fonte: autoria própria (2022).

2.3 Sensor de temperatura – DS18B20

O sensor de temperatura que é utilizado neste projeto é o modelo DS18B20 do tipo sonda.

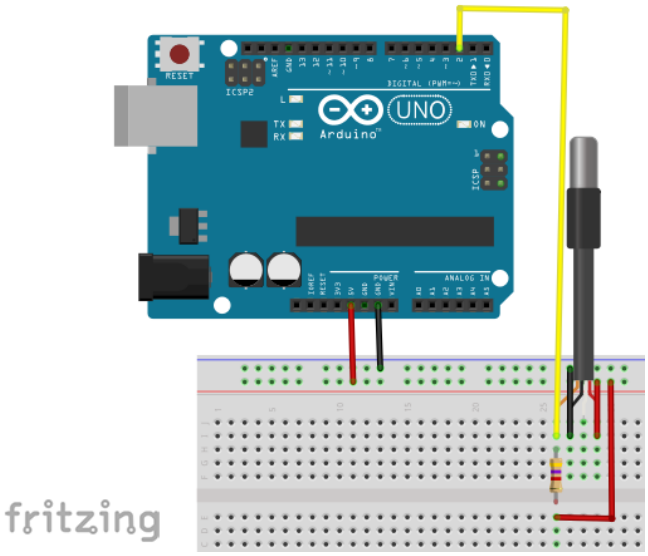
De acordo com a folha de dados do componente (*Datasheet*), trata-se de um sensor digital capaz de realizar aferições de -55º até 125 ºC, podendo ser utilizado para medir a temperatura do ar ou de líquidos, com precisão de ± 0,5 ºC.

As especificações técnicas do DS18B20 são descritas logo abaixo:

- Chip: DS18B20;
- Tensão de operação: 3 - 5,5 V;
- Faixa de medição: -55°C a +125 °C;
- Precisão: ± 0.5 °C entre -10 °C e +85 °C;
- Ponta de aço inoxidável;
- Dimensão ponta de aço: 6 x 50 mm;
- Dimensão do cabo: 100 cm;
- Interface de 1 fio.

O diagrama elétrico do sensor de temperatura pode ser observado através da figura 6. Na figura 7 nota-se o sensor DS18B20 posicionada no interior da estufa.

Figura 6 - Diagrama elétrico do Arduino com o DS18B20.



Fonte: Curto Circuito (2021).

Figura 7 - Sensor de temperatura no interior da estufa.

Fonte: autoria própria (2022).

2.4 Lâmpada halógena

De acordo com o INEE (2022), a lâmpada incandescente apresenta uma eficiência energética de 8%, de modo que apenas 8% da potência elétrica total é convertida em potência luminosa, enquanto os 92% restantes da potência da lâmpada são convertidos em energia térmica através do efeito Joule.

Considerando os dados da fabricante Foxlux (2022), sobre a lâmpada halógena, (figura 8), uma lâmpada halógena de 100 W pode substituir uma lâmpada incandescente de 150 W de potência, de modo que a potência luminosa será mantida. E, pode-se afirmar que a lâmpada halógena deste fabricante apresenta uma eficiência de 50% em relação a lâmpada incandescente.

Com estes dados podemos deduzir que a potência elétrica que será convertida em potência luminosa de uma lâmpada incandescente de 150 W, será:

$$Potência\ luminosa_{Lâmpada\ incandescente\ (150\ W)} = 150 * 8\% = 12\ W \quad (1)$$

E sabe-se, que uma lâmpada halógena de 100 W, consegue fornecer a mesma potência luminosa de uma lâmpada incandescente de 150 W, com isso é possível encontrar a eficiência de uma lâmpada halógena:

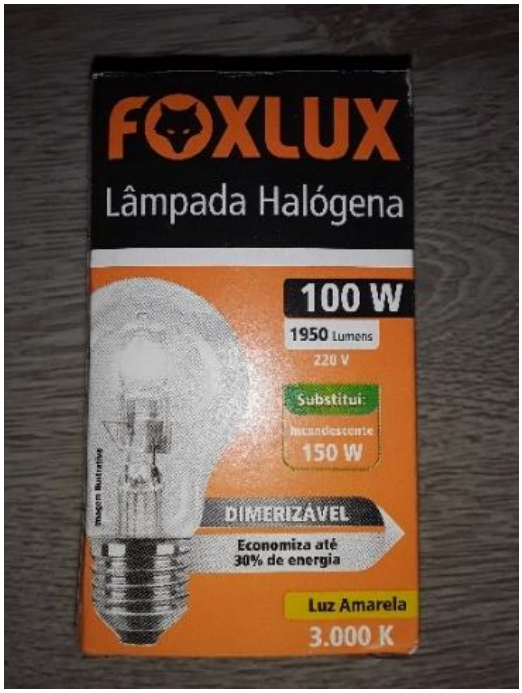
$$Potência\ luminosa_{Lâmpada\ halógena\ (100\ W)} = 100 * Eficiência_{Lâmpada\ halógena} = 12\ W \quad (2)$$

Com isso:

$$Eficiência_{Lâmpada\ halógena} = 12\% \quad (3)$$

Desta forma, a lâmpada halógena apresenta uma eficiência energética de 12%, de modo que apenas 12% da potência elétrica total é convertida em potência luminosa, enquanto os 88% restantes da potência da lâmpada são convertidos em energia térmica através do efeito Joule.

Figura 8 - Modelo da lâmpada utilizada.



Fonte: autoria própria (2022).

Na figura 9, é possível visualizar a lâmpada halógena posicionada no interior da estufa.

Figura 9 - Lâmpada halógena no interior da estufa.

Fonte: autoria própria (2022).

É importante ressaltar que a alimentação da lâmpada com a rede elétrica não será realizada de forma direta, visto que se precisa ter o controle de ligar e desligar a lâmpada, além de realizar a variação da potência, com isso foi utilizado um módulo *Dimmer* para fazer conexão elétrica da lâmpada com a rede elétrica.

A resistência da lâmpada pode ser encontrada de acordo com a equação abaixo.

$$R_{Lâmpada} = \frac{V^2}{P} \tag{4}$$

$$R_{Lâmpada} = \frac{(220)^2}{100} = 484\ \Omega \tag{5}$$

Onde:

$R_{Lâmpada}$ = Resistência da lâmpada

V = Tensão nos terminais da lâmpada

P = Potência nominal da lâmpada

2.5 Módulo Dimmer

2.5.1 Funcionamento do TRIAC

Para compreender o funcionamento do módulo *Dimmer*, é fundamental entender o funcionamento de um TRIAC, que é o componente responsável pela variação da tensão de um Dimmer.

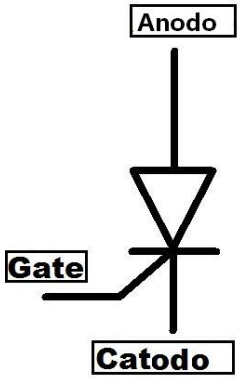
De acordo com Hart (2016), através da conexão e desconexão de chaves eletrônicas, é possível construir um controlador capaz de alterar a tensão, corrente e potência de uma carga CA com uma fonte CA.

Há várias chaves eletrônicas com características singulares, de modo que suas características são próprias para determinados tipos de aplicações. Uma dessas chaves é o tiristor, que é um dispositivo semicondutor de grande importância na eletrônica de potência.

Segundo Ahmed (1998), os tiristores são chaves eletrônicas que conseguem converter e controlar grandes cargas, utilizando uma baixa potência. O regime de chaveamento de tal dispositivo, é controlado através de três terminais, de modo que a corrente é conduzida do terminal ânodo ao terminal cátodo, e controlada através de impulsos elétricos através do terminal denominado *gate*.

Na figura 10, nota-se a estrutura de um tiristor, e os seus três terminais: Ânodo, Cátodo e *Gate*.

Figura 10 - Esquema de um Tiristor.



Fonte: Riva (2013).

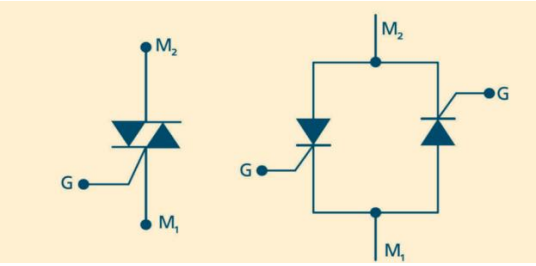
O *triode for alternating current* (TRIAC), traduzido para o português como triodo de corrente alternada, é um tiristor capaz de conduzir corrente elétrica nos dois sentidos de polarização, sendo possível fazer o controle em onda completa. Através de uma baixa corrente no terminal de disparo (*gate*), é possível fazer o fechamento do circuito, permitindo uma alta passagem de corrente, dependendo do modelo do TRIAC.

Na figura 11, é possível notar a estrutura externa do componente TRIAC, enquanto na figura 12 é possível visualizar o diagrama da estrutura interna do TRIAC.

Figura 11 – TRIAC.

Fonte: Baú da Eletrônica (2022).

Figura 12 - Estrutura do TRIAC.



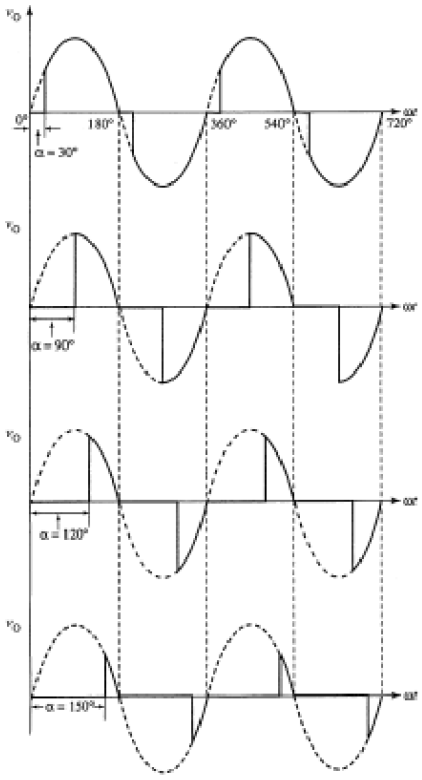
Fonte: e-tec Brasil (2011).

É necessário que haja a identificação quando a senoide passa por zero, visto que o envio de um pulso elétrico para o terminal *gate* precisa estar sincronizado com a senoide. O circuito responsável pela detecção deste evento é chamado de detector de *zero-crossing*, onde o módulo *Dimmer* também possui tal circuito em sua estrutura.

O intervalo de tempo do momento em que a senoide passa por zero até o momento em que será enviado o pulso elétrico para o terminal *gate*, será o controle de tensão, que é denominado de ângulo de disparo ou ângulo de corte.

Na figura 13, nota-se o comportamento da senoide em relação ao ângulo de disparo.

Figura 13 - Corte de onda da senoide.



Fonte: Ahmed (2009).

Nota-se que o ângulo de corte da onda senoidal varia de 0º até 180º, de modo que quanto mais próximo de 180º, maior será o corte da onda senoidal, como a tensão instantânea será reduzida, consequentemente a tensão eficaz também será reduzida. Nas equações: 1,2,3 e 4, podemos ver os valores da tensão e corrente eficaz, da potência de saída e do fator de potência.

Valor da tensão eficaz:

$$V_{0\,(RMS)} = V_i \left\{1 - \frac{\alpha}{\pi} + \frac{sen(2\alpha)}{2\pi}\right\}^{1/2} \quad (6)$$

Valor da corrente eficaz:

$$I_{0\,(RMS)} = \frac{V_i}{R} \left\{1 - \frac{\alpha}{\pi} + \frac{sen(2\alpha)}{2\pi}\right\}^{1/2} \quad (7)$$

Valor da potência média:

$$P_{0\,(avg)} = I^2_{0\,(RMS)} * R = \frac{V^2_{0\,(RMS)}}{R} \quad (8)$$

Fator de potência:

$$FP = \frac{P_{Saída}}{P_{Entrada}} = \left\{1 - \frac{\alpha}{\pi} + \frac{sen(2\alpha)}{2\pi}\right\}^{1/2} \quad (9)$$

Onde:

$V_{0\,(RMS)}$ = Tensão de saída eficaz;

$I_{0\,(RMS)}$ = Corrente de saída eficaz;

$P_{0\,(avg)}$ = Potência média de saída;

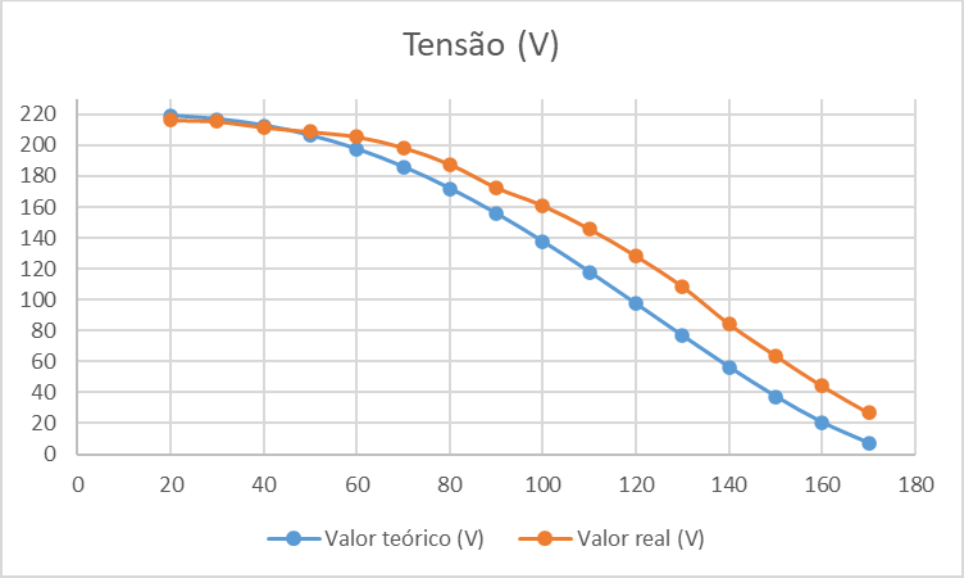
FP = Fator de potência (Relação da potência de saída com a entrada);

α = Ângulo de disparo ou ângulo de corte (Radiano).

Com o auxílio de um voltímetro, foi realizada a medição da tensão nos terminais de saída do módulo *Dimmer*, visto que valores reais de tensão em um circuito geralmente apresenta uma pequena discrepância em relação ao valor teórico, devido a precisão dos componentes elétricos e eletrônicos.

Na figura 14, é possível notar a curva que representa a variação da tensão (V) em relação ao ângulo de disparo (º), onde a curva azul representa os valores teóricos de tensão enquanto a curva laranja representa os valores medidos de tensão.

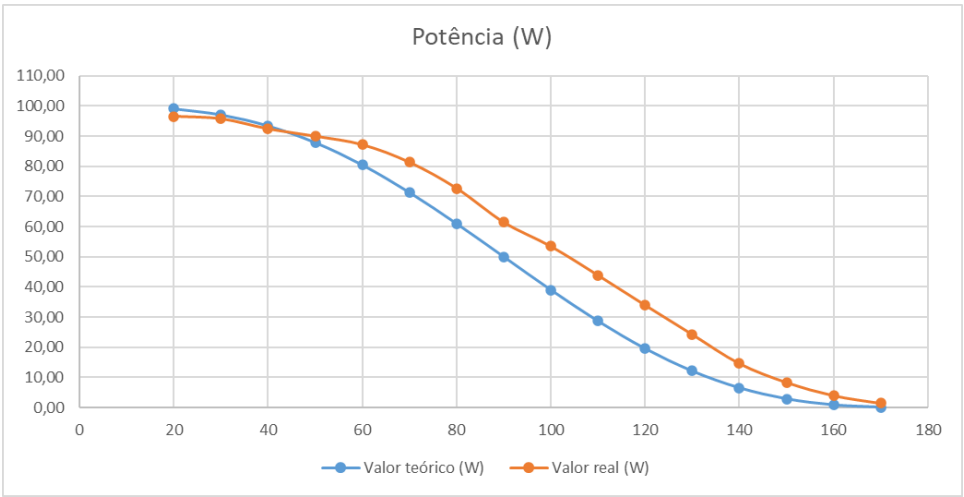
Figura 14 - Valor teórico da tensão comparado com o valor real da tensão em relação a variação do ângulo.



Fonte: autoria própria (2022).

Conhecendo a tensão do Dimmer com a variação do ângulo de disparo, e conhecendo a resistência da lâmpada, é possível encontrar a potência real, que pode ser vista na figura 15.

Figura 15 - Valor teórico da potência comparado com o valor real da tensão em relação a variação do ângulo.



Fonte: autoria própria (2022).

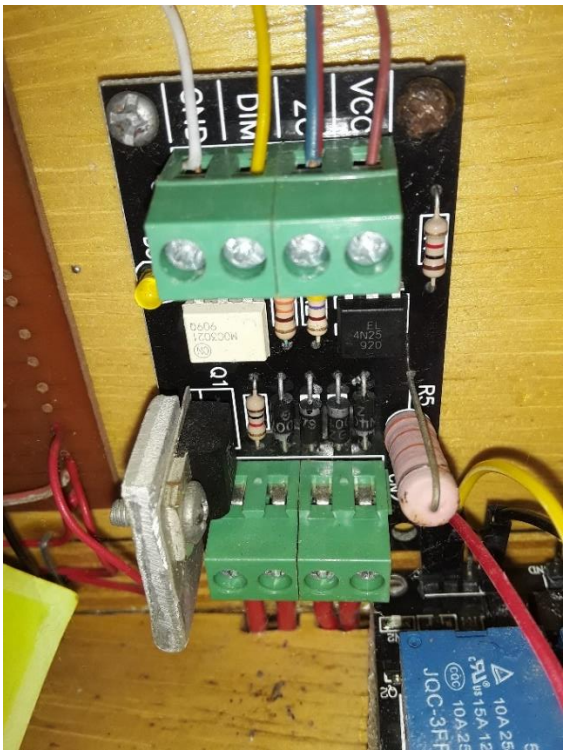
É perceptível que os valores teóricos e medidos apresentam um certo nível de proximidade que já era esperada devido a precisão do módulo *Dimmer*, contudo essa pequena discrepância é suficiente para causar uma alteração significativa em

nossos testes. Logo, para este estudo, será utilizado os valores reais da potência, para as análises.

2.5.2 Funcionamento do módulo Dimmer

O módulo *Dimmer* do *Arduino* contém um TRIAC, de modo que é possível fazer a variação da tensão de saída, e consequentemente a potência de saída. Na figura 14 nota-se a módulo *Dimmer* inserido na parte externa da estufa.

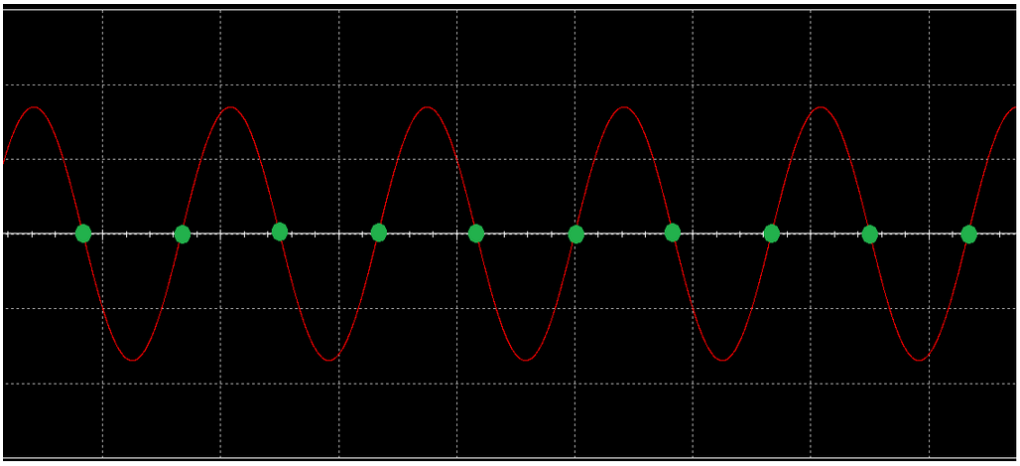
Figura 16 - Módulo Dimmer.



Fonte: autoria própria (2022).

O módulo *Dimmer* envia um pulso para o *Arduino* sempre que a tensão da rede atinge zero, desta forma, um código pode ser utilizado afim de enviar o pulso de ativação para o terminal de gatilho do TRIAC, conforme determinado tempo. Esse tempo é baseado no ângulo de disparo, assim é possível fazer o controle da tensão, e consequentemente a potência da lâmpada. Na figura 17, nota-se os pontos em que a senoide tem o valor igual a zero, neste momento há o envio de um pulso elétrico do *Dimmer* para o *Arduino*.

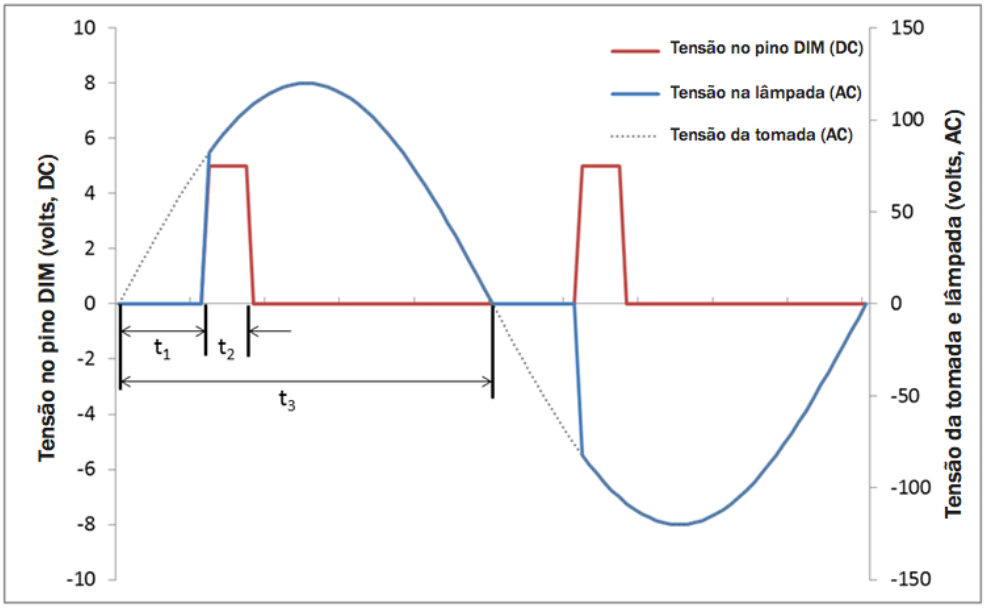
Figura 17 - Momento em que o valor da senoide é igual a zero.



Fonte: Laboratório de Garagem (2016).

Na figura 18, podemos notar o funcionamento da variação de tensão pelo módulo *Dimmer*. A variável t_1 é o tempo necessário para fazer o acionamento do TRIAC, esse acionamento é feito através de um pulso elétrico que dura um tempo representado por t_2, enquanto t_3 representa o tempo total do meio-ciclo da onda senoidal da rede.

Figura 18 - Funcionamento da variação da tensão pelo módulo Dimmer.



Fonte: BR-Arduino (2016).

Para se calcular o tempo necessário de ativação do TRIAC para cada ângulo de disparo, primeiramente deve-se considerar o tempo de cada meio-ciclo.

$$Meio - ciclo = \frac{1}{2f} = \frac{1}{120\ Hz} = 8{,}33\ ms \quad (10)$$

Onde:

f = frequência da senoide da rede elétrica.

Dividindo o tempo de meio-ciclo por 180º obtém-se o coeficiente, aqui chamado de k, o qual quando multiplicado por qualquer ângulo de 0º até 180º, resultará no tempo necessário de espera necessário para a ativação do TRIAC.

$$k = \frac{8{,}333}{180} = 0{,}04627 \quad (11)$$

Com isto, caso deseje saber o tempo necessário para a ativação do TRIAC (t_1) considerando um ângulo de 90º, tem-se:

$$Tempo\ de\ ativação\ do\ TRIAC = k * ângulo = 0{,}04627 * 90 = 4{,}1643\ ms \quad (12)$$

2.6 Ventoinha

Na parte superior da estufa, há uma ventoinha de 3 W de potência, alimentada por uma fonte ATX (*Advanced Technology Extended*) externa de 12 V, cuja função é fazer a dissipação do ar quente do interior da estufa.

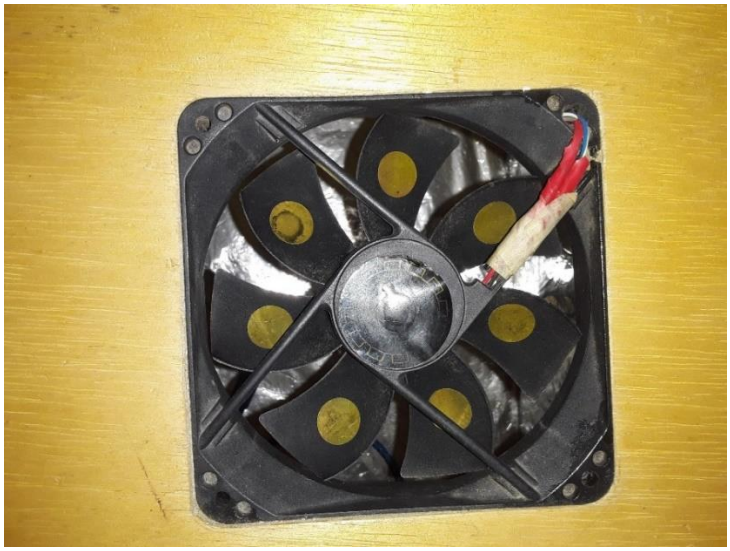
Quando a tampa da estufa está fechada, que é o estado de operação necessário para o funcionamento adequado da estufa, a corrente de ar que fluirá pela ventoinha será menor do que quando a tampa está aberta, isso se deve ao fato de que quando a estufa está fechada o fluxo de ar é de dentro para fora, também há diminuição da pressão interna, fazendo com que o ar externo seja forçado para dentro da estufa através das pequenas frestas ao redor da estufa, e também pela abertura superior onde se situa a ventoinha.

Apesar de o fluxo de ar pela ventoinha ser baixo quando a tampa está fechada, é ainda extremamente relevante para dispersão do ar quente do interior da estufa, e assim provocar uma significativa diminuição de temperatura.

A abertura superior onde se situa a ventoinha também provoca uma perda considerável de energia térmica devido ao efeito da convecção, pois o ar quente fica menos denso e consequentemente tende a subir e escapar pela abertura da ventoinha. Todavia a lâmpada é capaz de suprir tal perda até determinada temperatura, além de que essa perda simulará os ruídos de um sistema real, e será útil para os testes dos controladores, simulando então essa instabilidade do sistema.

Na figura 19, é possível notar o posicionamento da ventoinha na parte superior da estufa.

Figura 19 - Ventoinha da estufa.



Fonte: autoria própria (2022).

2.7 Relé

Segundo Petruzella (2014), o relé é um dispositivo eletromecânico que funciona como um interruptor, fazendo com que haja a abertura ou fechamento de um circuito elétrico. Com ele é possível conectar e desconectar um dispositivo de uma fonte CA ou CC de alta potência com um sinal CC de baixa potência.

De acordo com Braga (2017), para fazer o acionamento do relé, é necessário energizar a sua bobina interna, de modo que quando há a passagem de corrente, é criado uma campo magnético fazendo com que o enrolamento funcione como um eletroímã, e assim fazer com que o contator seja atraído em direção a bobina e por consequência fechando o circuito.

Quando o enrolamento é desenergizado, o contator volta para a posição original através da força exercida por uma mola.

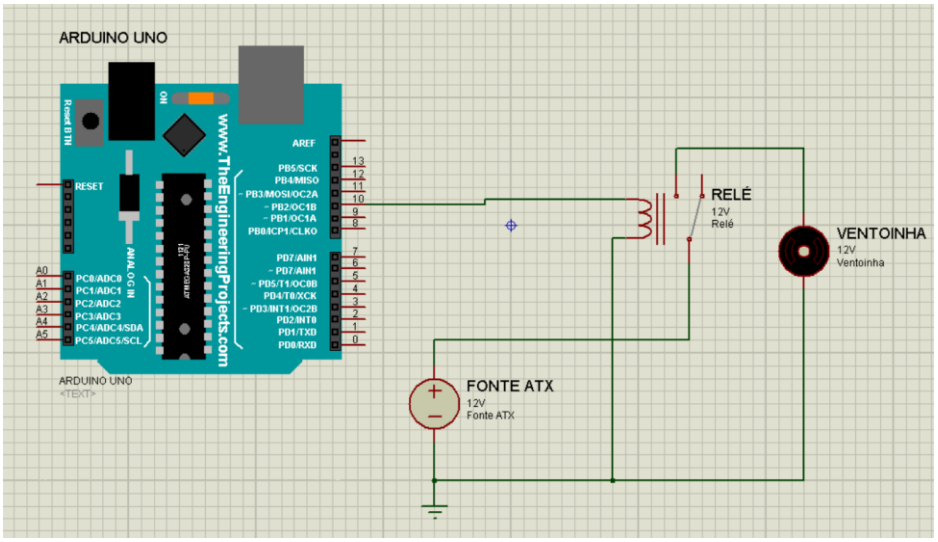
Neste projeto, foi utilizado um módulo relé para *Arduino* de dois canais, de modo que apenas um canal foi utilizado, sua função é realizar a abertura e o fechamento do circuito de alimentação da ventoinha com a fonte ATX.

O relé apresenta três terminais relacionados aos comutadores, onde um terminal é conectado a fonte alimentadora, e os outros dois representam o contato normalmente aberto (NA) e o contato normalmente fechado (NF); e há também dois terminais relacionados ao controle do chaveamento do sistema, onde um terminal é

conectado ao terra (*Ground*) do *Arduino*, e o outro terminal é conectado em uma porta digital, de modo que, dependendo do estado da porta (Ligado ou desligado), haverá a acionamento ou desligamento elétrico do enrolamento do relé e, consequentemente, a alteração da posição do contato, de modo a abrir ou fechar o circuito da ventoinha.

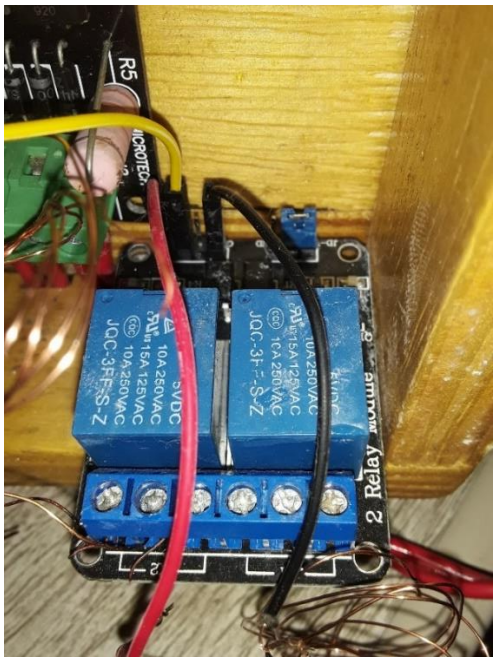
Na figura 20, nota-se o diagrama de acionamento da ventoinha com o relé, e na figura 21, o posicionamento do relé na parte externa da estufa.

Figura 20 - Diagrama do acionamento da ventoinha com o relé.



Fonte: autoria própria (2022).

Figura 21 – Relé.



Fonte: autoria própria (2022).

3 MODELO MATEMÁTICO DO SISTEMA DE CONTROLE

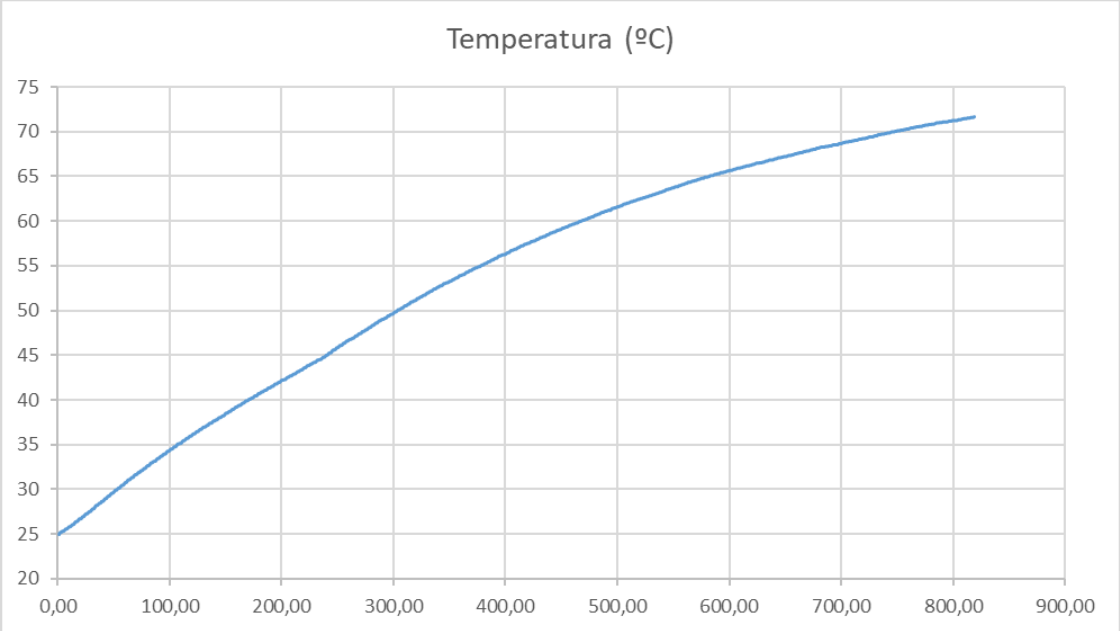
Neste capítulo é discutido sobre a modelagem matemática dos sistemas de controle, detalhando o comportamento da malha aberta e malha fechada.

3.1 Comportamento da temperatura no interior da estufa

Para conhecer o comportamento da temperatura deste projeto, foi realizada a obtenção dos dados de temperatura, considerando a temperatura ambiente e inicial de 25 ºC, e acionando a lâmpada em sua potência elétrica máxima de 100 W, e com a tampa da estufa fechada.

Os dados foram armazenados em uma planilha do programa *Microsoft Excel*. E foi utilizado um gráfico suavizado para a visualização do comportamento da estufa que pode ser visto na figura 22.

Figura 22 - Aumento da temperatura (ºC) do interior da estufa em relação ao tempo (segundos).



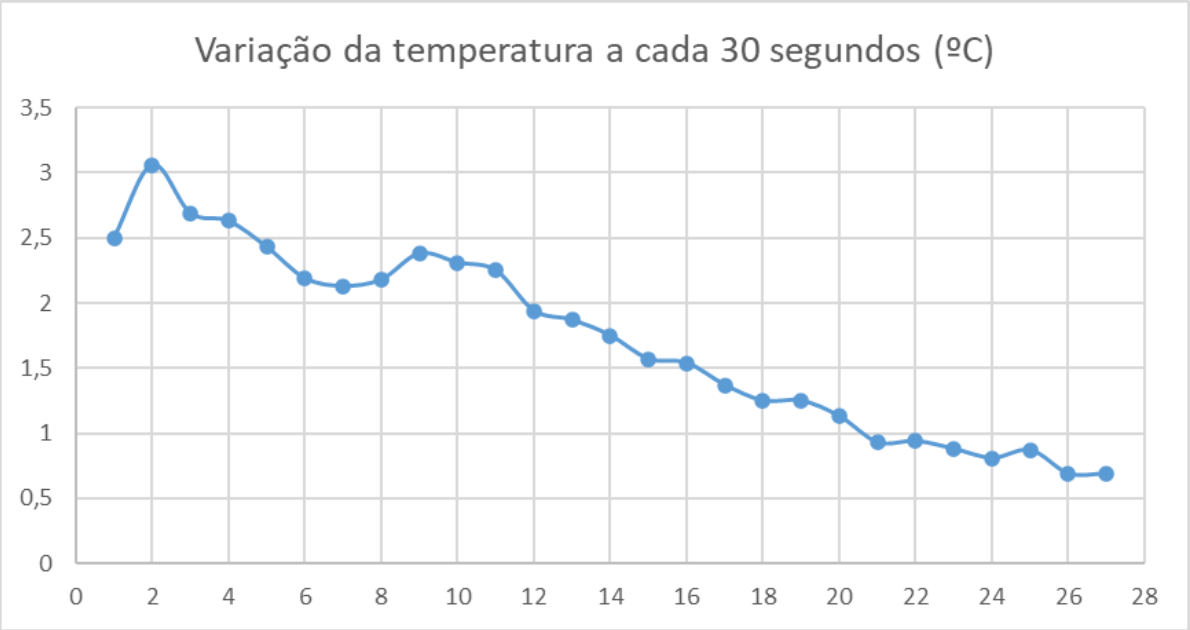
Fonte: autoria própria (2022).

Como pode ser visto na figura 22, não foi possível obter os dados de temperatura no regime permanente, visto que quando a temperatura excedia os 71,69 ºC, o *Arduino* apenas registrava o valor -127 ºC no *Excel*.

Não foi possível encontrar o fator que ocasionava este erro, visto que a temperatura máxima que o sensor de temperatura poderia ler era de 125 ºC, além de que não havia aquecimento no *Arduino*.

Para resolver o problema, obteve-se a variação de temperatura a cada trinta segundos, como se pode ver na figura 23. É notório que, inicialmente, o comportamento da curva apresenta uma característica periódica, contudo, quanto mais tempo passa, mais estável fica a curva do gráfico, tendendo a uma linha reta.

Figura 23 - Variação da temperatura (ºC) a cada 30 segundos.



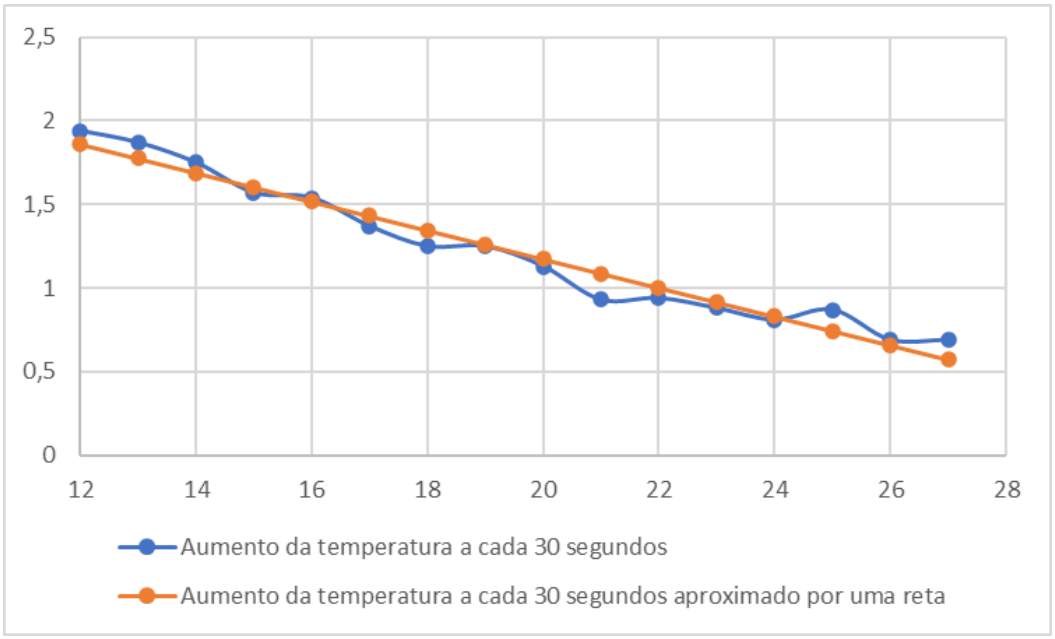
Fonte: autoria própria (2022).

3.2 Estimação através do Método dos Mínimos Quadrados

Como o gráfico da figura 23, a partir dos 360 segundos (12 no eixo horizontal) está próximo de uma reta, foi utilizado o método dos mínimos quadrados para fazer uma aproximação dos pontos através de uma reta que melhor se aproxime dos pontos e assim estimar os dados referentes a variação de temperatura no futuro.

A utilização do método dos mínimos quadrados, se deve ao fato da reta gerada por esta técnica, apresenta o menor erro relativo médio. Na figura 24, é possível notar os pontos medidos (Azul), com os pontos gerados pela reta com o método dos mínimos quadrados (Laranja).

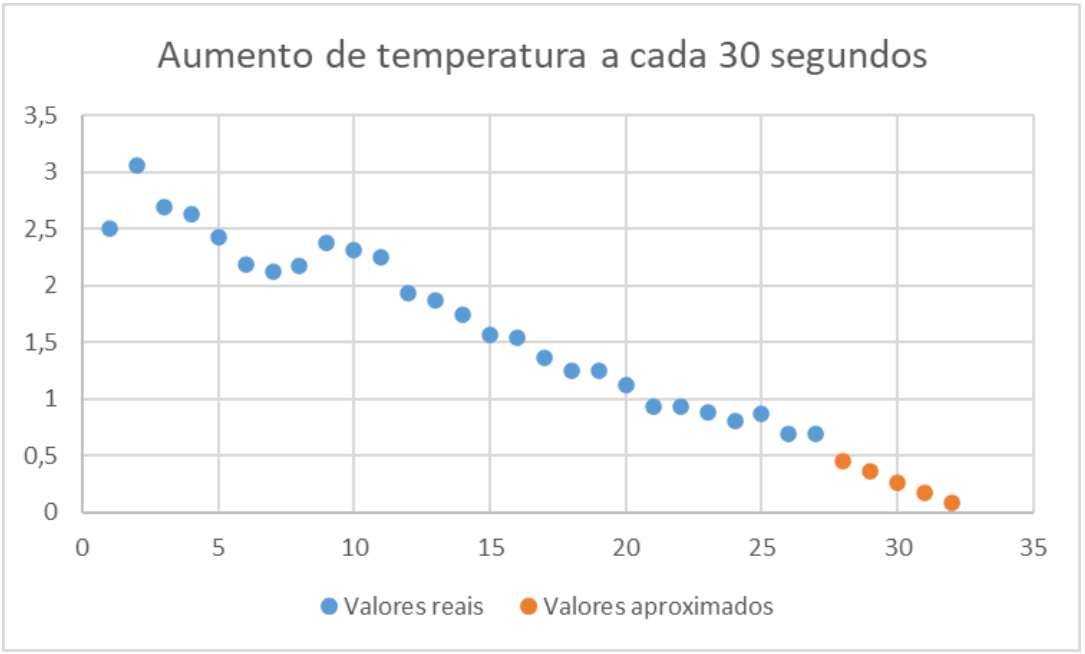
Figura 24 - Aproximação dos valores reais do aumento de temperatura através de uma reta.



Fonte: autoria própria (2022).

A reta obtida apresenta um coeficiente de determinação de 0,963 e um erro relativo médio de 6,21%. Com base na reta aproximada, é possível encontrar a variação de temperatura a partir do instante 840 segundos até 960 segundos. Na figura 25, os pontos em azul representam os valores reais enquanto os pontos em laranja representam os valores estimados.

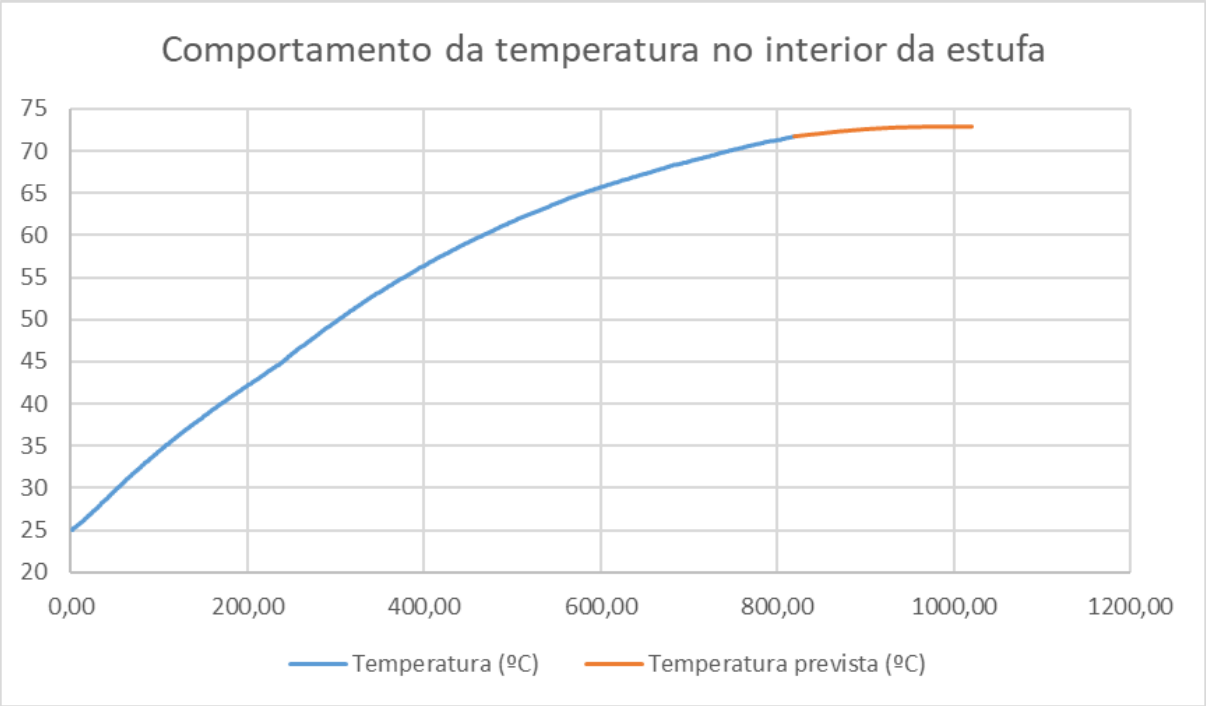
Figura 25 - Aproximação dos valores futuro.



Fonte: autoria própria (2022).

Conhecendo a variação de temperatura aproximada futura, podemos prever a estabilização de temperatura no regime permanente, o que pode ser visto na figura 26.

Figura 26 - Aumento da temperatura no interior da estufa ao longo do tempo (segundos).



Fonte: autoria própria (2022).

Desta forma, baseado no banco de dados construído, obtemos que o valor da temperatura no regime permanente é de aproximadamente 72,875 ºC.

Também foram obtidos os dados referentes a variação da temperatura ao longo do tempo, considerando o desligamento da lâmpada quando a temperatura atinge 50 ºC, foi realizada a comparação com o acionamento da ventoinha e sem o acionamento da ventoinha.

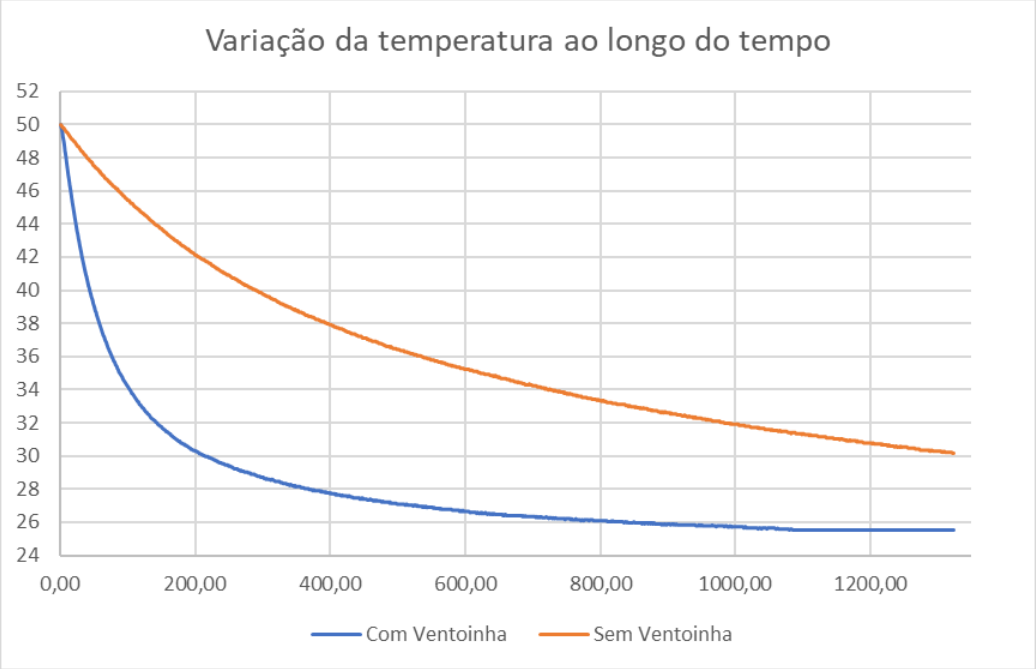
Pelo fato de a estufa ser construída de materiais que apresentam uma menor transferência de energia térmica para o ambiente, é esperado que mesmo quando a lâmpada fosse desligada o sistema mantenha a temperatura interna com uma taxa de perda bastante reduzida.

Também deve ser ressaltado que o acionamento da ventoinha não apresenta um fluxo intenso de corrente de ar de dentro para fora, visto que não há aberturas além das pequenas fissuras da caixa que permitam a entrada do ar do ambiente para substituir o que foi removido. Todavia, mesmo o fluxo sendo inferior do que o

esperado, ele é suficiente para a remoção de uma quantidade significativa de ar quente de dentro da estufa.

Tais características dos dois modos de operação para o resfriamento da estufa, pode ser notado na figura 27.

Figura 27 - Diminuição da temperatura da estufa ao longo do tempo.



Fonte: autoria própria (2022).

3.2 Função de transferência de um modelo térmico

Segundo Dorf (2001), a engenharia de controle se utiliza de determinadas metodologias para aprimorar projetos relacionados aos processos industriais.

Segundo Aguirre (2004), para aprimorar a eficiência dos processos industriais, uma das metodologias é a utilização de modelos matemáticos do processo para a construção do modelo de controle.

Esses modelos matemáticos são baseados em comportamentos físicos do sistema, de modo que podemos descrever tal comportamento através de equações matemáticas.

É preciso ressaltar que o comportamento real do sistema não é igual ao do modelo matemático descrito, visto que o modelo real apresenta ruídos que são originados através de fatores externos não controláveis.

Todavia, se a perturbação ao sistema for significativa e apresentar um comportamento físico que seja capaz de ser descrito através de equações matemáticas, então a mesma deverá ser inserida junto das equações do sistema.

A função de transferência de um sistema representado por uma equação diferencial linear invariante no tempo é definida como a relação entre a transformada de Laplace da saída e a transformada de Laplace da entrada, admitindo-se todas as condições iniciais nulas.

De acordo com Ogata (2010), podemos definir a função de transferência de um sistema físico como a representação da relação entre a transformada de Laplace entrada e a transformada de Laplace da saída através de um modelo matemático representado por uma equação diferencial linear invariante no tempo.

A função de transferência além de representar o comportamento do sistema no domínio da frequência, é utilizado para a construção de controladores que utilizam as características: Proporcional, integrativa e derivativa.

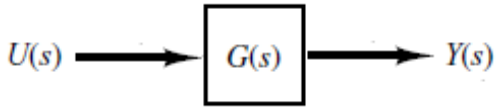
Para encontrar a função de transferência deste projeto, foi realizada a obtenção dos dados de temperatura, considerando a temperatura ambiente e inicial de 25 ºC, e acionando a lâmpada em sua potência máxima e com a tampa da estufa fechada.

Para encontrar a função de transferência utilizou-se o gráfico da figura 26, onde é possível notar que o comportamento da temperatura se assemelha a um sistema de primeira ordem, cuja forma geral de função de transferência de primeiro grau é:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{K}{\tau s + 1} \tag{13}$$

Em que Y(s) representa a saída e U(s) a entrada, como mostrado na figura 28.

Figura 28 - Função de transferência.



Fonte: Ogata (2010).

Com isso pode-se representar o K como o ganho do nosso sistema:

$$K = 72{,}875 - 25 = 47{,}875 \tag{14}$$

E τ como o tempo necessário para se atingir 63,2% da temperatura, considerando o ganho (K) somado com o valor inicial, com isso:

$$Temperatura(\tau = 1) = 47{,}875 * 0{,}632 + 25 = 55{,}16125 \tag{15}$$

De acordo com o banco de dados, o tempo necessário para se atingir a temperatura de 55,16 ºC é de 381 segundos, logo a função de transferência final da estufa, será:

$$G(s) = \frac{47{,}875}{381s + 1} \tag{16}$$

3.3 Controle de malha aberta e controle de malha fechada

Os sistemas de controle podem ser definidos por dois tipos: Malha aberta e malha fechada.

No sistema de malha aberta, o sinal da saída não é medido e consequentemente não é comparado com o valor de referência da entrada. Desta forma, sistemas que utilizam malha aberta tendem a sofrer mais com ruídos, visto que não é possível detecta-los e ameniza-los. Além de que, este tipo de sistema geralmente se baseia no tempo para a realização dos processos, assim não é possível saber se o resultado já foi obtido.

Um exemplo de um sistema de controle de malha aberta é a máquina de lavar roupas, que, por sua vez, não sabe quando a roupa já está limpa, e continua seu processo de limpeza até atingir um determinado tempo que foi configurado. O diagrama da malha aberta pode ser observado na figura 29.

Figura 29 - Malha aberta.

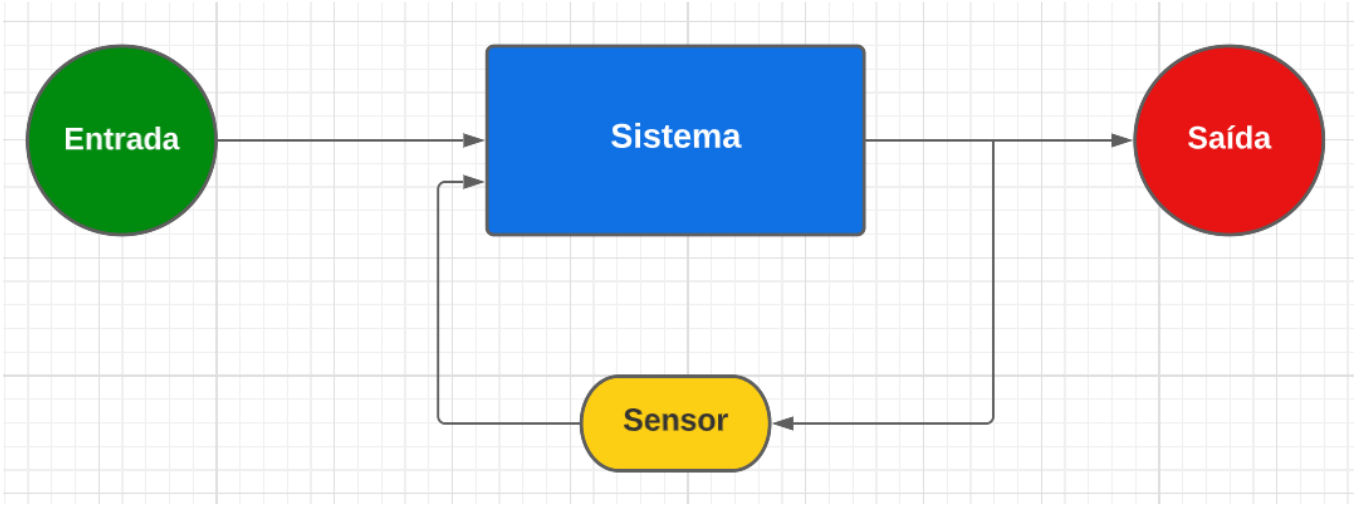
Fonte: autoria própria (2022).

Já no caso de um sistema de malha fechada, há uma realimentação do sinal de saída para o sistema, assim é possível comparar o valor de referência da entrada com o valor de saída, de modo que a diferença entre tais valores é chamada de erro. O erro é um parâmetro fundamental em um sistema de controle, visto que através dele é possível analisar se a ação de controle está reduzindo ou aumentado o erro, e com isso fazer uma variação no atuador de modo que o erro seja reduzido o suficiente de acordo com o que foi projetado.

Um exemplo de um sistema de controle de malha fechada é a própria estufa descrita neste trabalho, onde é enviado um valor de temperatura para o *Arduino*, de

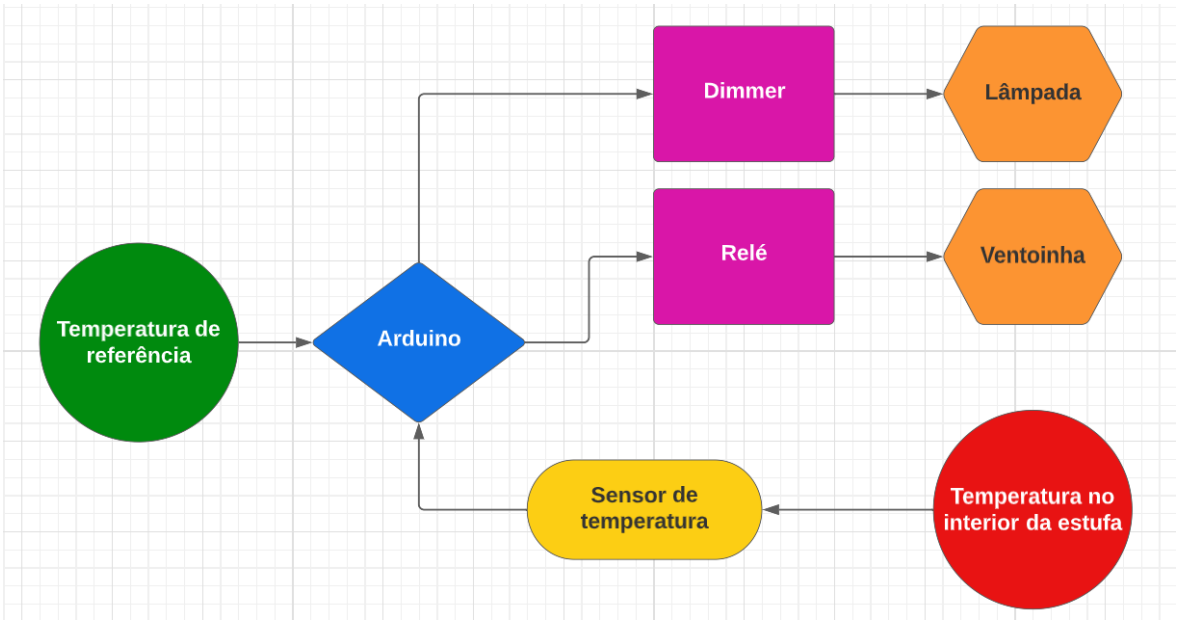
modo que a lâmpada será acionada para aquecer ou acionar a ventoinha para fazer o resfriamento do interior da estufa, e o sensor interno fará a medição da temperatura em tempo real, e enviará os valores para o *Arduino*, onde será realizado o controle dos atuadores (Lâmpada ou ventoinha), de acordo com o valor de referência solicitado e o valor real de dentro da estufa. O diagrama da malha fechada pode ser observado na figura 30, enquanto o diagrama da estufa deste projeto pode ser observado na figura 31.

Figura 30 - Malha fechada.



Fonte: autoria própria (2022).

Figura 31 - Representação em diagrama de blocos do sistema de controle da estufa.



Fonte: autoria própria (2022).

3.4 Controladores

3.4.1 Introdução sobre controladores

De acordo com Ogata (2010), um controlador automático realiza a comparação entre o sinal de saída com o sinal de entrada, de modo que reduzirá ou anulará a diferença entre os sinais. Portanto, o valor da saída se aproxima da entrada, e a forma que o controlador conduzirá este processo é denominado de ação de controle.

Existem determinados controladores que são categorizados segundo as suas ações de controle, e um dos tipos de controladores, que é amplamente utilizado na indústria, é o controlador Proporcional Integral Derivativo (PID).

3.4.2 Controlador de duas posições

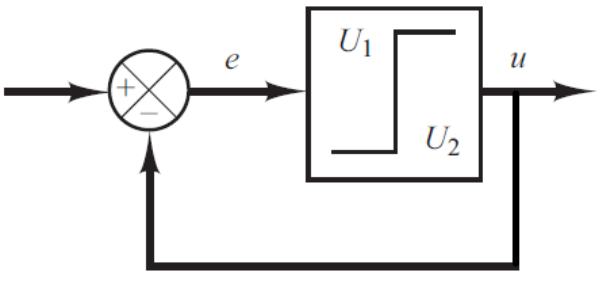
De acordo com Ogata (2010), o controlador de duas posições, também conhecido como controlador *On-Off* ou ligado-desligado, é o controlador mais simples e barato que existe sendo amplamente utilizado em sistema de controle domésticos e industriais.

Considerando que o sinal de saída do controlador seja u(t) e que o erro seja representado por e(t), podemos representar o comportamento do controlador de duas posições pela equação abaixo.

$$u(t) = \begin{cases} U_1, & para\ e(t) > 0 \\ U_2, & para\ e(t) < 0 \end{cases} \tag{17}$$

Na figura 32 é possível visualizar o diagrama de blocos de um controlador de duas posições.

Figura 32 - Diagrama de blocos de um controlador On-Off.



Fonte: OGATA (2010).

Onde U_1 e U_2 são constantes, de modo que U_2 geralmente é zero ou $-U_1$. No exemplo deste projeto, o controlador de duas posições atua de modo que quando a temperatura atual do interior da estufa for menor do que a temperatura de referência, a lâmpada é acionada e a ventoinha desligada, de modo que a temperatura aumente. Contudo, quando a temperatura for igual a temperatura de referência, a lâmpada é desligada e a ventoinha continua desligada. E quando a temperatura do interior da estufa for maior do que a temperatura de referência, apenas a ventoinha é acionada e assim diminuir a temperatura interna da estufa.

Desta forma, o comportamento do controlador de duas posições no projeto pode ser representado da seguinte forma.

$$u(t) = \begin{cases} Lâmpada = 1 \quad Ventoinha = 0, \ \ para\, T_{atual} < T_{Referência} \\ Lâmpada = 0 \quad Ventoinha = 0, \ \ para\, T_{atual} = T_{Referência} \\ Lâmpada = 0 \quad Ventoinha = 1, \ \ para\, T_{atual} > T_{Referência} \end{cases} \tag{18}$$

Onde T é a temperatura, 0 a posição de desligado e 1 a posição de ligado. A desvantagem desse tipo de controle é que a frequência de operação dos atuadores é bastante elevada para qualquer variação do valor de saída, mesmo que seja insignificante, fazendo com que haja um desgaste físico dos atuadores dependendo do tipo.

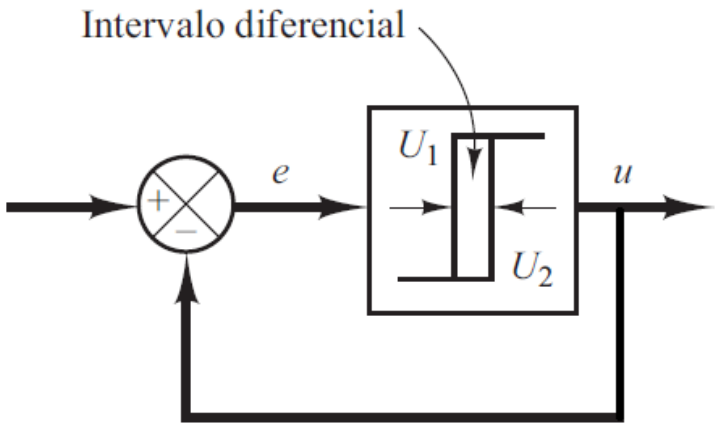
3.4.3 Controlador de duas posições com intervalo diferencial

De acordo com Ogata (2010), a faixa de valores permitidos para a variação do sinal do erro antes de haver a comutação é chamado de intervalo diferencial.

Basicamente, é um controlador de duas posições melhorado, visto que variações pequenas dos valores de saída não farão com que os atuadores sejam operados, e com isso haverá um aumento na sua vida útil.

Esse intervalo pode ser configurado de acordo com a precisão de cada processo industrial. Na figura 33, é possível observar o diagrama de blocos referente ao controlador de duas posições com intervalo diferencial.

Figura 33 - Diagrama de blocos de um controlador On-Off com intervalo diferencial.



Fonte: OGATA (2010)

3.4.4 Controlador proporcional

A característica proporcional faz com que a variável de controle seja ajustada proporcional ao erro, fazendo com que seja amplificado, assim sendo responsável pela aceleração do tempo de resposta e minimização do erro, porém nunca o elimina. A relação entre a saída do controlador u(t) e o sinal de erro atuante e(t) é:

$$u(t) = K_p e(t) \tag{19}$$

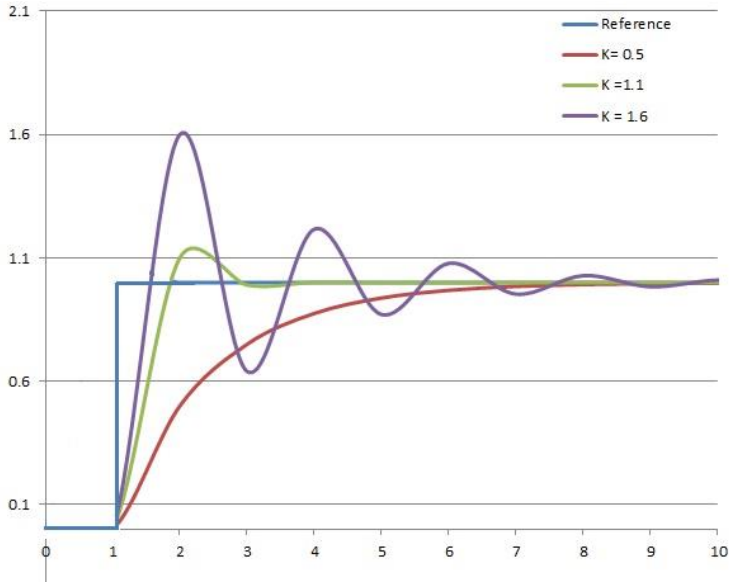
Fazendo a transformada de Laplace,

$$\frac{U(s)}{E(s)} = K_p \tag{20}$$

Onde K_p é o ganho proporcional.

Na figura 34 é possível perceber que o aumento do K_p acelera a resposta do sinal, porém seu aumento em excesso pode causar sobressalto e até oscilações no sinal, reduzindo o tempo de estabelecimento.

Figura 34 - Influência do Kp.



Fonte: Wikipedia (2022).

3.4.5 Controlador proporcional-integral

A característica integrativa produz um sinal de saída que é o produto da amplitude do erro pela sua duração, de modo que a saída se torna o erro acumulado. O controlador integral de ganho K_i elimina completamente o erro no regime permanente, contudo poderá tornar mais lenta a resposta do sistema e aumentar o sobressalto da resposta em uma resposta transiente. O valor da saída u(t) do controlador é modificado a uma taxa de variação proporcional ao sinal de erro atuante e(t).

$$\frac{du(t)}{dt} = K_i e(t) \tag{21}$$

A equação anterior também pode ser rescrita da seguinte forma

$$u(t) = K_i \int_0^t e(t)dt \tag{22}$$

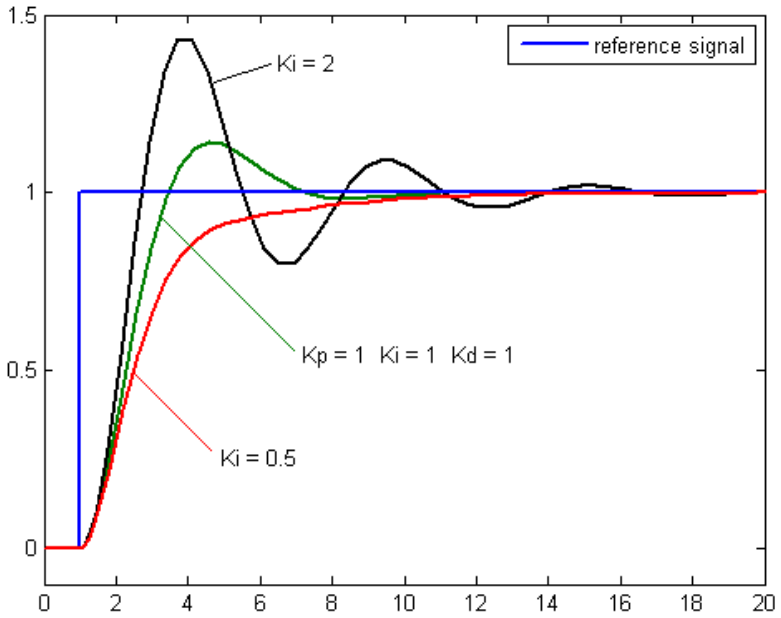
Desta forma a função de transferência de um controlador integral será:

$$\frac{U(s)}{E(s)} = \frac{K_i}{s} \tag{23}$$

Em que K_i é o ganho integrativo.

Na figura 35, é perceptível que o aumento de K_i reduz o tempo de subida, todavia há um aumento na oscilação e o no tempo de estabelecimento, fazendo com que seja necessário ajustar K_i de acordo com a resposta desejada para um determinado sistema, sendo semelhante ao K_p, porém com erro nulo no regime permanente.

Figura 35 - Influência do Ki.



Fonte: Wikipedia (2022).

É importante ressaltar que a característica integrativa de forma isolada não é um método de controle, visto que é necessário ter a ação proporcional integrada ao controlador, com isso teremos o controlador proporcional-integral (PI), que pode ser representado por:

$$u(t) = K_p e(t) + \frac{K_p}{T_i}\int_0^t e(t)dt \tag{24}$$

Logo, a função de transferência do controlador PI, será:

$$\frac{U(s)}{E(s)} = K_p\left(1 + \frac{1}{T_i s}\right) \tag{25}$$

3.4.6 Controlador proporcional-derivativo

A característica derivativa cria um sinal de saída que é proporcional à variação da velocidade de variação do erro. O controlador derivativo de ganho K_d

consegue fornecer uma correção antecipada do erro, reduzindo o tempo de resposta e o sobressalto, além de aumentar a estabilidade do sistema.

$$u(t) = K_d \frac{de(t)}{dt} \tag{26}$$

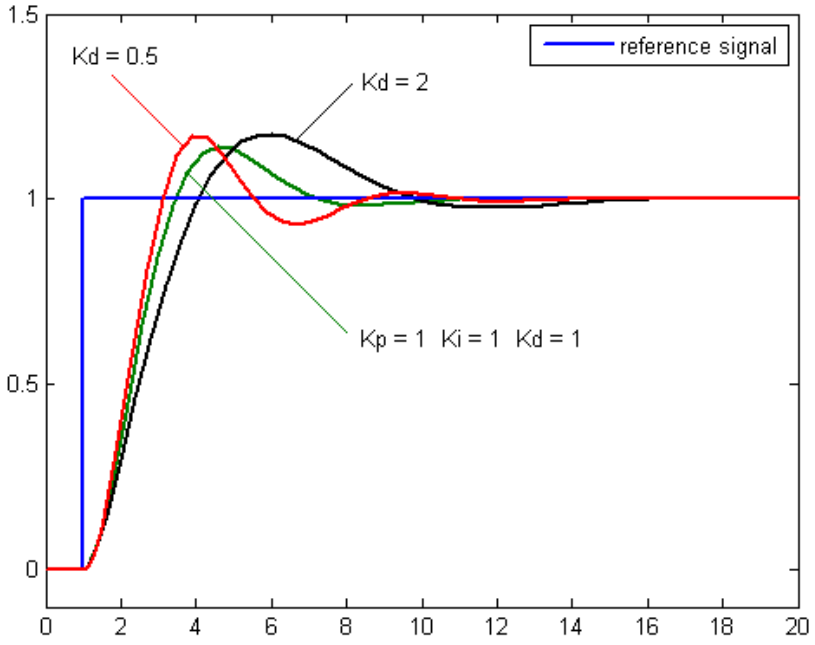
Cuja equação anterior pode ser rescrita da seguinte forma:

$$\frac{U(s)}{E(s)} = K_d s \tag{27}$$

Onde K_d é o ganho derivativo.

Na figura 36, nota-se que o aumento de K_d reduz as oscilações, todavia há um aumento no tempo de subida, tornando mais lenta a resposta do sistema.

Figura 36 - Influência do Kd



Fonte: Wikipedia (2022)

Assim como a característica integrativa, o derivativo de forma isolada não é um método de controle, visto que é necessário ter a ação proporcional integrada ao controlador, com isso teremos o controlador proporcional-derivativo (PD), que podemos representar por:

$$u(t) = K_p e(t) + K_p T_d \frac{de(t)}{dt} \tag{28}$$

Logo, a função de transferência do controlador PD, será:

$$\frac{U(s)}{E(s)} = K_p(1 + T_d s) \tag{29}$$

3.4.7 Controlador proporcional-integral-derivativo

Já o controlador proporcional-integral-derivativo (PID) (Sigla), apresenta as três características mencionadas anteriormente. Que pode ser representado por:

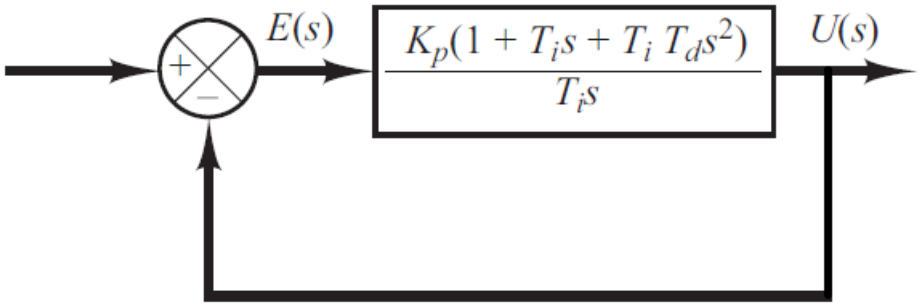
$$u(t) = K_p e(t) + \frac{K_p}{T_i}\int_0^t e(t)dt + K_p T_d \frac{de(t)}{dt} \tag{30}$$

Cuja função de transferência é:

$$\frac{U(s)}{E(s)} = K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \tag{31}$$

Em que K_p é o ganho proporcional, T_i é o tempo integrativo e T_d é o tempo derivativo. Na figura 37, pode-se ver a representação de um controlador PID através de um diagrama de blocos.

Figura 37 - Diagrama de um controlador PID.



Fonte: OGATA (2010).

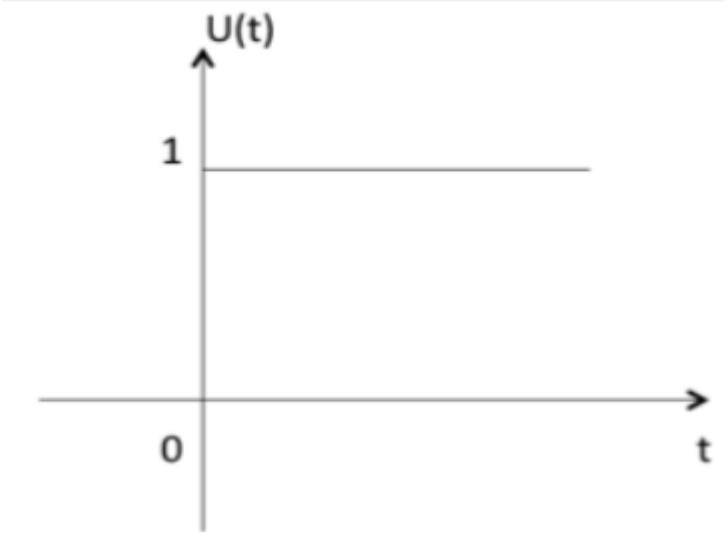
3.4.8 Aplicando o degrau unitário

A função de degrau unitário, também conhecida como função de Heaviside, é um tipo de função de entrada utilizada para testar o comportamento de uma função de transferência de uma planta.

Pode-se interpretar a função de degrau unitário como um valor de referência que se deseja no sistema, como a temperatura, fluxo de água, velocidade, ou

qualquer outra variável do nosso sistema, de modo que que o valor 1 representa o valor cem por cento da referência.

Figura 38 - Função de degrau unitário.



Fonte: TDPS (2022).

Pode-se representar matematicamente a função de Heaviside da seguinte maneira:

$$u_c(t) = \begin{cases} 1, & t \geq c \\ 0, & t < c \end{cases} \quad c \geq 0 \tag{32}$$

Em particular é possível descrever u da seguinte forma:

$$u \equiv u_0 \tag{33}$$

Desta forma:

$$u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases} \tag{34}$$

Fazendo a transformada de Laplace:

$$\mathcal{L}[u(t)] = \int_0^{\infty} e^{-st}\, u_c(t) dt \tag{35}$$

$$\mathcal{L}[u_c(t)] = \frac{e^{-sc}}{s} \tag{36}$$

Considerando:

$$u \equiv u_0 \tag{37}$$

Desta forma, pode-se reescrever a equação x, da seguinte forma:

$$\mathcal{L}[u(t)] = \frac{e^0}{s} \tag{38}$$

E com isso a transformada de Laplace da função de degrau unitário será:

$$\mathcal{L}[u(t)] = \frac{1}{s} \tag{39}$$

Desta forma, pode-se escrever matematicamente o comportamento da planta com a entrada sendo um degrau unitário, como mostrado na equação abaixo:

$$F(s) = \mathcal{L}[u(t)] * G(s) = \frac{1}{s} * \frac{47{,}875}{381s + 1} \tag{40}$$

Que também pode ser escrita da seguinte forma:

$$F(s) = \mathcal{L}[u(t)] * G(s) = \frac{47{,}875}{381s^2 + s} \tag{41}$$

4 METODOLOGIA

Neste capítulo é detalhada a metodologia que foi utilizada no desenvolvimento do projeto para a criação dos controladores além de como foram realizados os ensaios de cada controlador.

4.1 Introdução

Os controladores: Duas posições, duas posições com intervalo diferencial, PI e rastreador baseado nos métodos numéricos, foram testados de forma individual, com códigos construídos especificamente utilizando a lógica de cada controlador, e utilizando um macro do programa *Excel* para que fosse possível ser registrado em uma planilha os dados de temperatura gerados a cada segundo em tempo real. A temperatura inicial de referência utilizada foi de 25 ºC, e a temperatura desejada de referência foi de 35 ºC.

Devido a planta representar um sistema de primeira ordem e não apresentar oscilações no regime transitório, foi utilizado o controlador PI no sistema, já que a característica derivativa não teria utilidade na resposta que desejada, por isso foi utilizado o controlador PI, de modo que a característica proporcional iria diminuir o tempo de resposta, além de que a característica integrativa anularia o erro no regime estacionário.

Os controladores foram construídos com base em códigos livres disponibilizados na internet, como o código para a obtenção de dados de temperatura pelo sensor de temperatura retirado do website “FilipeFlop”, código para enviar dados para uma planilha no *Excel* retirado do website “Clube da Robótica”, código para a variação da potência da lâmpada através do módulo *Dimmer* retirado do canal do Youtube “Brincando com ideias”, de modo que todos esses códigos foram adequados para cada tipo de controlador, além da adição de novas linhas de código, que são responsáveis de fato pelo controle e acionamento dos atuadores, com base na resposta da temperatura.

4.2 Controlador de duas posições

O controlador de duas posições, também conhecido como controlador *On-Off*, é o controlador mais simples considerando a construção do código, visto que há apenas 3 casos, o primeiro em que a temperatura atual é inferior a temperatura de referência, isso faz com que apenas a lâmpada seja acionada em sua potência máxima, realizando o aquecimento do interior da estufa; o segundo caso ocorre

quando a temperatura atual for superior a temperatura de referência, isso faz com que apenas a ventoinha seja acionada fazendo com que o ar quente do interior da estufa seja trocado pelo ar do ambiente; e o terceiro caso acontece quando a temperatura atual for igual a temperatura de referência, fazendo com que a lâmpada e a ventoinha fiquem desligadas.

4.3 Controlador de duas posições com intervalo diferencial

Utilizando a própria característica da estufa de armazenar energia térmica, utilizou-se a ideia de construir duas faixas de temperatura, uma superior e outra inferior, cada faixa tem uma diferença de 0,5 ºC em relação a temperatura de referência (35 ºC), logo a faixa superior é de 35,5 ºC e a inferior de 34,5 ºC, de modo que se a temperatura atual for menor do que a faixa inferior, a lâmpada será acionada em sua potência máxima até se igualar a temperatura de referência, e a ventoinha só será ligada caso a temperatura atual seja maior do que a faixa superior e será desligada quando a temperatura for igual a temperatura de referência.

4.4 Controlador PI

Como já foi mencionado anteriormente, a função de transferência da estufa é descrita por.

$$G(s) = \frac{47{,}875}{381s + 1} \tag{42}$$

Que pode ser representada por uma função genérica, da seguinte forma:

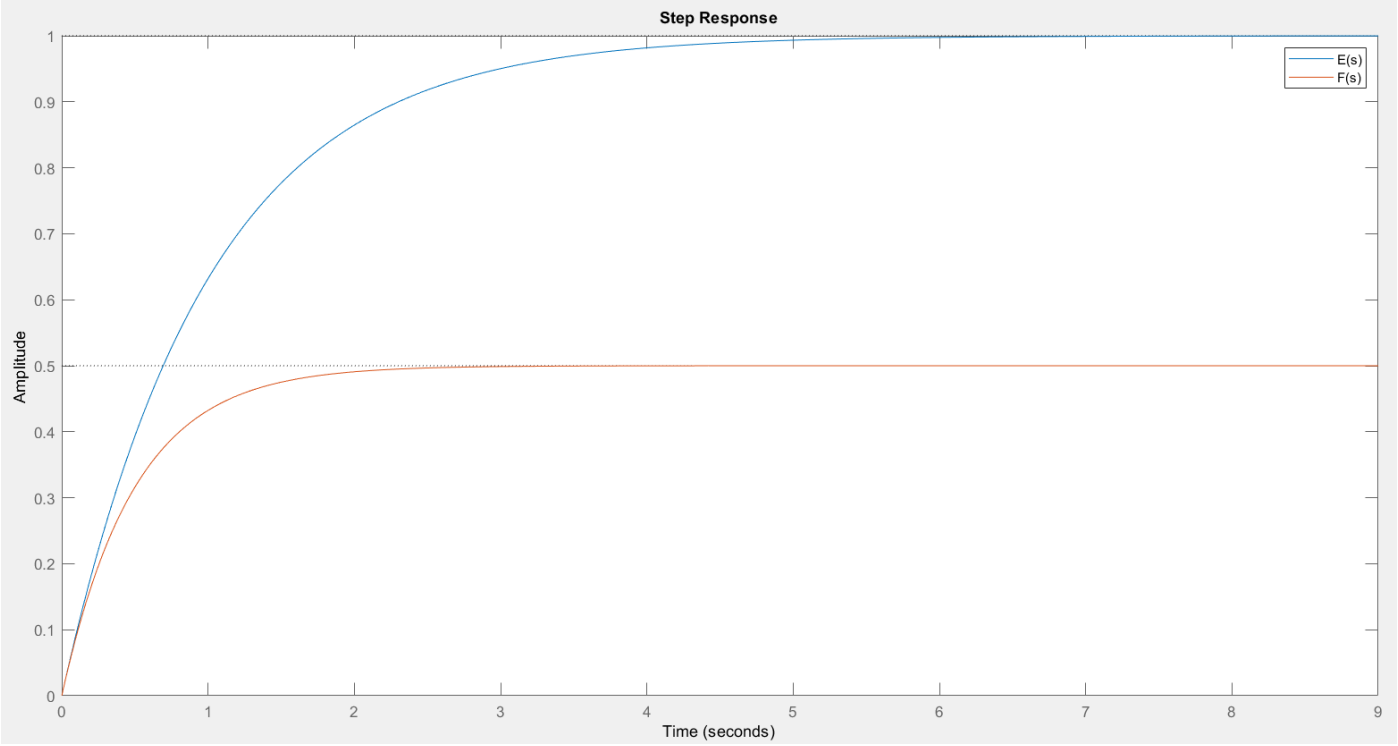
$$F(s) = \frac{1}{s + 1} \tag{43}$$

Outra função de transferência importante para as análises desta pesquisa, por possuir um polo em 0, é:

$$E(s) = \frac{1}{s} \tag{44}$$

Utilizando a plataforma *MATLAB*, pode-se verificar o comportamento das FTs considerando que estejam inseridas em um sistema de realimentação unitária, visto que o modelo da estufa apresenta tal estrutura, com isso na figura 39, nota-se o comportamento de F(s) em laranja e o comportamento de E(s) em azul.

Figura 39 - Comportamento das FTs.



Fonte: autoria própria (2022).

A amplitude representa o percentual que a FT irá atingir no regime estacionário em relação ao valor de referência, também conhecido como *SetPoint*, desta forma é nítido que E(s) alcança 100% do valor esperado, enquanto F(s) alcança apenas 50% do valor esperado. Com isso, é necessário adicionar um controlador que modifique os polos da função de transferência da planta que é representada por F(s), para E(s) para assim atingir o valor de referência desejado no regime permanente.

Sabendo que a resolução do conversor analógico-digital do *Arduino UNO* é de 256 (8 bits) para a modulação por largura de pulso ou do inglês *Pulse Width Modulation* (PWM), então é necessário dividir a resolução pelo ganho da função de transferência, desta forma:

$$G(s) = \frac{47{,}875}{381s + 1} * \frac{1}{256} = \frac{0{,}187}{381s + 1} \tag{45}$$

E com isso, para G(s) ser igual a E(s), o controlador C(s) é descrito da seguinte forma.

$$H(s) = C(s) * G(s) = \frac{0{,}187}{s} \tag{46}$$

Reescrevendo a equação.

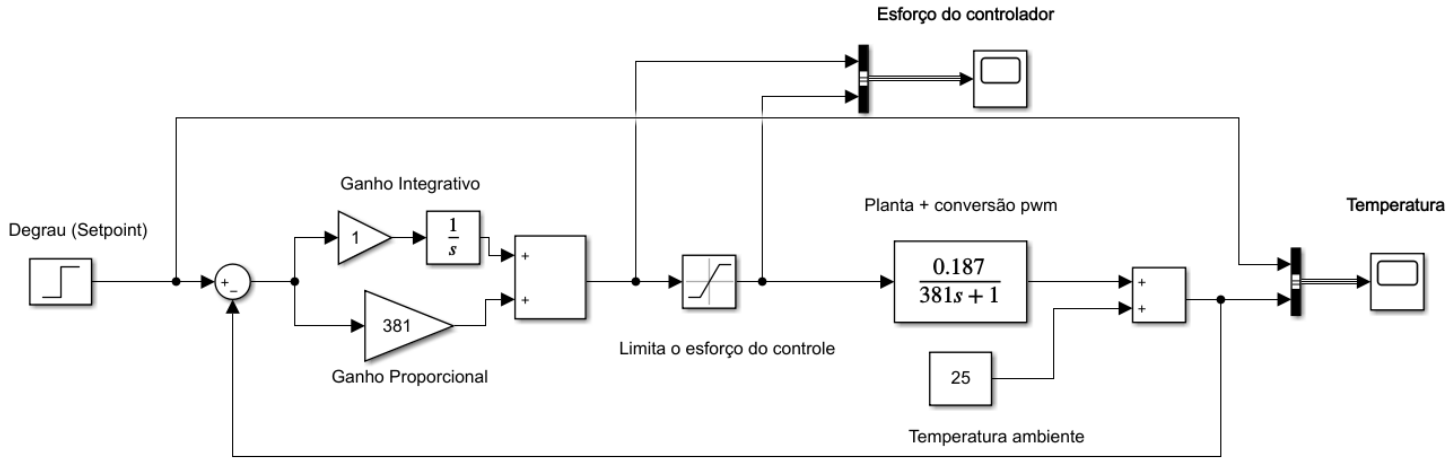
$$C(s) * \frac{0{,}187}{381s + 1} = \frac{0{,}187}{s} \tag{47}$$

$$C(s) = \frac{381s + 1}{s} \tag{48}$$

Com isso o controlador C(s) altera os polos da FT da planta, de modo que FT do sistema apresente uma convergência da resposta para o valor desejado. As equações do controlador e da planta são descritas no tempo contínuo.

Construído o sistema utilizando o SIMULINK, ferramenta do *MATLAB* utilizada principalmente para a simulação de sistemas descritos por equações matemáticas, e considerado o valor de referência igual a 35 ºC e o valor da temperatura ambiente igual a 25 ºC, tem-se o diagrama mostrado na figura 40.

Figura 40 - Diagrama da controlador e da planta no SIMULINK.

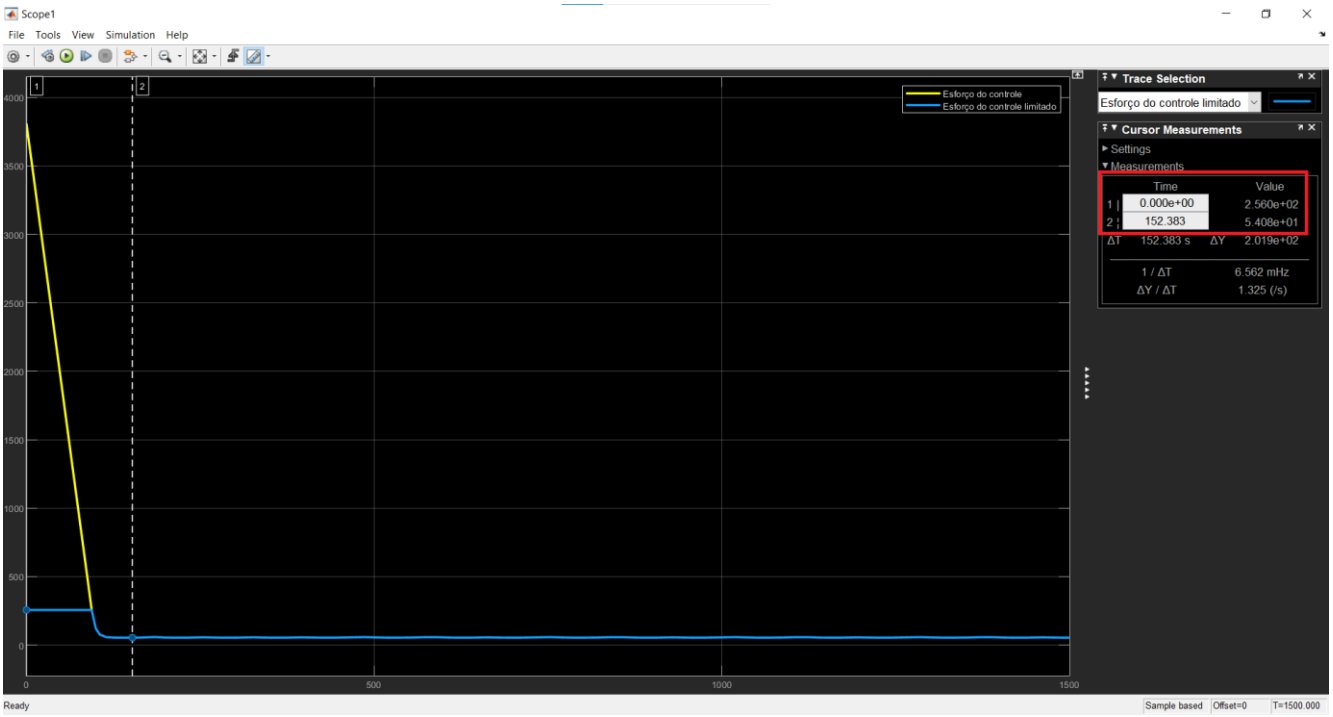


Fonte: autoria própria (2022).

Analisado no osciloscópio o esforço do controlador, observa-se duas curvas que representam o percentual de potência em que o atuador deve trabalhar, a curva em amarelo representa o real esforço teórico, enquanto a curva em azul representa o esforço real, considerando as limitações do *Arduino* e dos atuadores.

Na figura 41, nota-se as curvas que representam a ação de controle, onde a curva amarela representa a ação de controle teórica, enquanto a curva azul representa a ação de controle real, considerando a limitação do sistema.

Figura 41 – Comportamento da ação de controle teórico (Amarelo) e real (Azul).



Fonte: autoria própria (2022).

Especificamente, as curvas representam o percentual da tensão necessária em que a lâmpada deveria estar submetida para transformar a energia elétrica em energia térmica necessária para manter o sistema em determinada temperatura. Sabendo que a resolução do conversor analógico-digital do *Arduino UNO* é de 256 (8 bits) para a modulação PWM, foi utilizado os dados da curva em azul no momento inicial e no regime permanente para se obter a tensão e consequentemente a potência necessária para mantermos a estufa em 35 ºC, é aplicado as equações:

Tensão inicial, enquanto a temperatura atual for inferior a 35ºC. Na figura 41, nota-se que o valor inicial é 256, desta forma tem-se que.

$$V_{Inicial} = (220\ V)\left(\frac{256}{256} * 100\%\right) = 220 * 1 = 220\ V \quad (49)$$

Potência inicial, enquanto a temperatura atual for inferior a 35ºC.

$$P_{Inicial} = \frac{V^2}{R_{Lâmpada}} = \frac{(220)^2}{484} = 100\ W \quad (50)$$

Tensão final, quando a temperatura atual for igual ou superior a 35ºC. Na figura 41, nota-se que o valor final é 54, desta forma tem-se que.

$$V_{Final} = (220\ V)\left(\frac{54}{256} * 100\%\right) = 220 * 0{,}211 = 46{,}4\ V \quad (51)$$

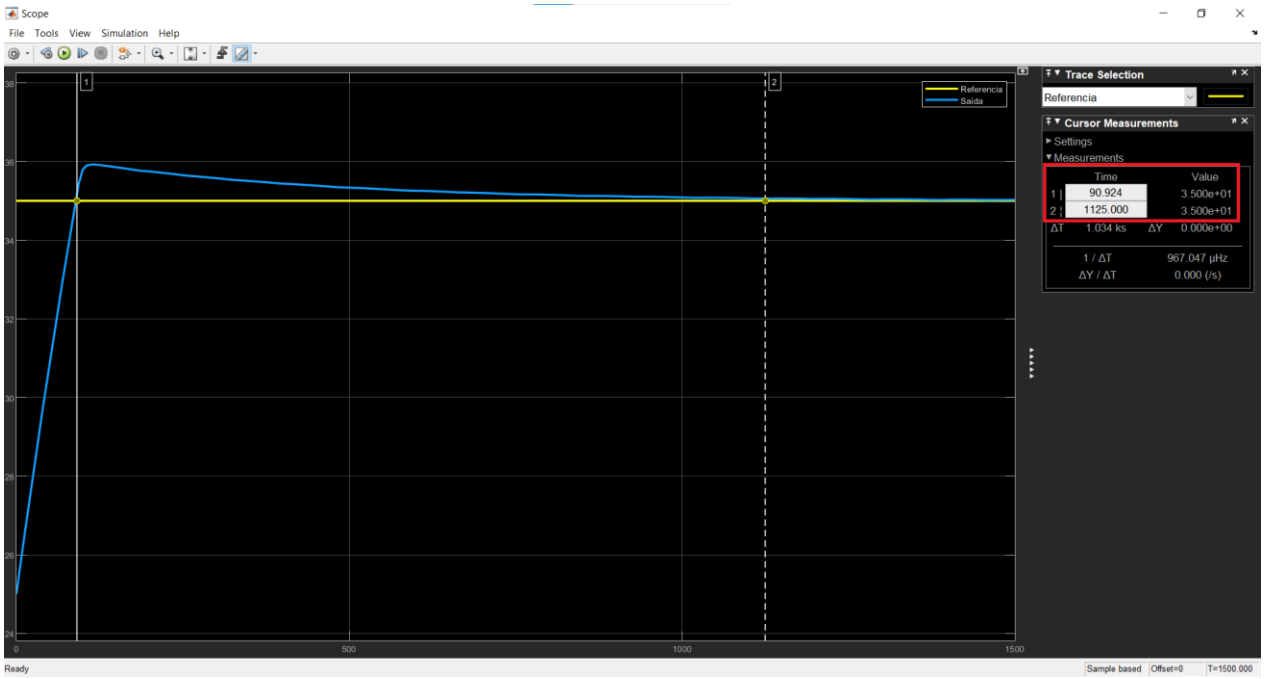
Potência final, quando a temperatura atual for igual ou superior a 35ºC.

$$P_{Final} = \frac{V^2}{R_{Lâmpada}} = \frac{(46{,}4)^2}{484} = 4{,}45\ W \quad (52)$$

Desta forma, foi possível observar a tensão nos terminais da lâmpada que é de 220 V para que a potência de 100 W, e que após 110 segundos, a lâmpada deve ter em seus terminais uma tensão de 46,4 V para gerar uma potência de 4,45 W, e assim manter a temperatura de 35 ºC.

Ainda utilizando o osciloscópio do SIMULINK para observar o comportamento da temperatura no regime transiente e estacionário, pode ser observado na figura 42.

Figura 42 – Comportamento da temperatura teórica em relação ao tempo (Azul) e a temperatura de referência (Amarelo).



Fonte: autoria própria (2022).

É importante ressaltar que o comportamento da temperatura visto na figura 42, aconteceria em um cenário onde não há a limitação no *Arduino* e nos atuadores, desta forma na realidade a temperatura vai demorar mais tempo para atingir 35 ºC, contudo, percebe-se que no regime permanente é possível ter o erro nulo.

Na tabela 2, pode-se notar os dados que foram apresentados anteriormente sobre a relação do ângulo de disparo com seu respectivo valor real da tensão nos terminais da lâmpada através do módulo *Dimmer* do *Arduino*. Através desses dados, e com algumas técnicas de métodos numéricos, é obtido um valor aproximado,

porém bastante próximo do ângulo necessário para que haja uma tensão de 46,4 V nos terminais da lâmpada.

De acordo com Chapra (2008), os métodos numéricos utilizam operações aritméticas simples para resolver problemas matemáticos complexos, de modo que essas técnicas necessitam de uma série de cálculos de modo a se aproximar do resultado final, tal repetição é chamada de interação.

Tabela 2 - Relação do ângulo de disparo com seu respectivo valor real da tensão nos terminais da lâmpada.

Ângulo (º)	**Valor real (V_{RMS})**
170	26,8
160	44,2
150	63,8
140	84,3
130	108,5
120	128,3
110	145,8
100	160,9
90	172,5
80	187,6
70	198,4
60	205,4
50	208,7
40	211,5
30	215,3
20	216,2

Fonte: autoria própria (2022).

Assim, é encontrado um valor aproximado da tensão para ângulos que não há os valores de tensão. Para isso, realizado a média do valor mínimo e o valor máximo mais próximos do resultado desejado e aumentando o número de iterações, mais próximo é o valor obtido do do valor real. As equações a seguir mostram o procedimento:

1º iteração.

$$V_{\hat{A}ngulo(155)} = \frac{V_{160} + V_{150}}{2} = \frac{44{,}2 + 63{,}8}{2} = 54\ V \tag{53}$$

2º iteração.

$$V_{\hat{A}ngulo(157{,}5)} = \frac{V_{160} + V_{155}}{2} = \frac{44{,}2 + 54}{2} = 49{,}1\ V \tag{54}$$

3º iteração.

$$V_{\hat{A}ngulo(158,75)} = \frac{V_{160} + V_{157,5}}{2} = \frac{44,2 + 49,2}{2} = 46,7\ V \quad (55)$$

No anexo E, é mostrado o código construído na linguagem Python, para a realização de tais cálculos.

Desta forma, para construir o controlador PI, foi utilizada a lógica de acionar a lâmpada em sua potência total durante a elevação da temperatura, e quando a temperatura é igual a 35 ºC o módulo Dimmer irá operar com um ângulo de disparo de 159ºC, de modo a manter a potência da lâmpada em 4,45 W no regime estacionário.

4.5 Controlador rastreador baseado nos métodos numéricos

O controlador rastreador desenvolvido neste projeto tem o princípio de adequar o acionamento dos atuadores com base nos dados fornecidos pelo sensor através de métodos numéricos, de modo que é possível fazer a sua implementação em sistemas que tem ou não uma função de transferência conhecida.

A lógica utilizada neste projeto se baseia em uma série de condições que busca ajustar a potência da lâmpada de forma dinâmica, através do comportamento da temperatura.

Os ângulos utilizados nos testes variam de 20º (216,2 V e 96,57 W) até 170º (26,8 V e 1,48 W), visto que colocar ângulo inferiores a 20º e superiores a 170º causam instabilidade na lâmpada, fazendo com que ela cause oscilações de potência. Considerando que u(t) seja o sinal de saída no regime permanente, tem-se.

AMa = Ângulo Máximo = 170º

AMi = Ângulo Mínimo = 20º

AA = Ângulo Atual = 20º

$$u(t) = \begin{cases} AA = \dfrac{AMa + AMi}{2} \quad e \quad AMi = AA, \ \ para\ T_{atual} > T_{Referência} \\ AA = \dfrac{AMa + AMi}{2} \quad e \quad AMa = AA, \ \ para\ T_{atual} < T_{Referência} \end{cases} \quad (56)$$

Desta forma, considerando manter a temperatura em 35 ºC, é necessária uma potência de 4,45 W, fazendo com que o ângulo de disparo seja aproximadamente

159 ºC. Então o controlador rastreador resultará nos seguinte ângulos a cada iteração.

1º iteração - Como 20º resulta em uma potência maior do que o ângulo 159º então a temperatura será maior do que 35 ºC.

$$\hat{A}ngulo\ atual = \frac{170 + 20}{2} = 95º \quad (57)$$

2º iteração - Como 95º resulta em uma potência maior do que o ângulo 159º então a temperatura será maior do que 35 ºC.

$$\hat{A}ngulo\ atual = \frac{170 + 95}{2} = 132{,}5º \quad (58)$$

3º iteração - Como 132,5º resulta em uma potência maior do que o ângulo 159º então a temperatura será maior do que 35 ºC.

$$\hat{A}ngulo\ atual = \frac{170 + 132{,}5}{2} = 151{,}25º \quad (59)$$

4º iteração - Como 151,25º resulta em uma potência maior do que o ângulo 159º então a temperatura será maior do que 35 ºC.

$$\hat{A}ngulo\ atual = \frac{170 + 151{,}25}{2} = 160{,}625º \quad (60)$$

5º iteração - Como 160,625º resulta em uma potência menor do que o ângulo 159º então a temperatura será menor do que 35 ºC.

$$\hat{A}ngulo\ atual = \frac{160{,}625 + 151{,}25}{2} = 156{,}125º \quad (61)$$

6º iteração - Como 156,125º resulta em uma potência maior do que o ângulo 159º então a temperatura será maior do que 35 ºC.

$$\hat{A}ngulo\ atual = \frac{160{,}625 + 156{,}125}{2} = 158{,}375º \quad (62)$$

Desta forma é possível notar que o valor do ângulo de disparo tende a se ajustar automaticamente ao ângulo necessário para manter a temperatura atual igual a temperatura de referência, além de que quanto mais iterações o sistema realizar, menor será o erro do ângulo de disparo atual com o ângulo de disparo necessário para se manter a temperatura em 35 ºC.

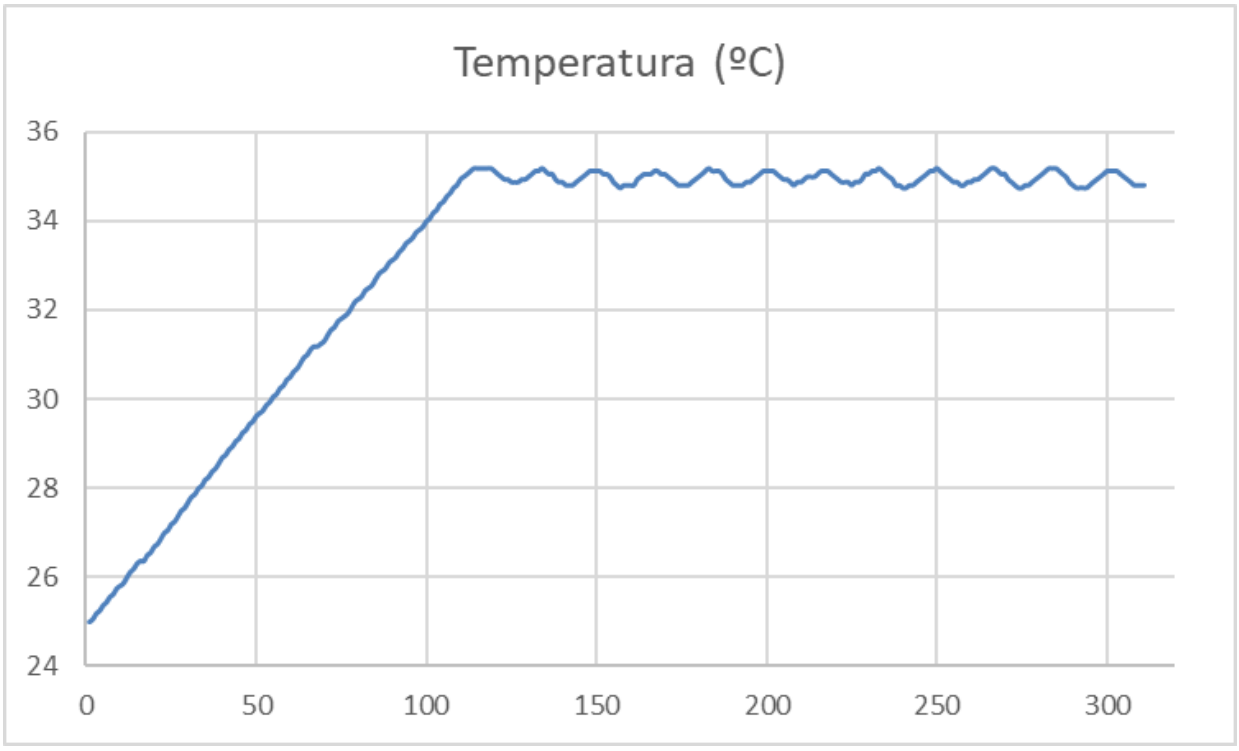
5 RESULTADOS E ANÁLISE

Neste capítulo são apresentados os resultados obtidos com cada controlador, analisado os dados obtidos e feita a comparação entre eles.

5.1 Controlador de duas posições

Na figura 43 é possível visualizar o comportamento da temperatura ao longo do tempo (segundos), utilizando o controlador de duas posições. É nítido que há uma oscilação semelhante à forma senoidal, próximo da temperatura de referência.

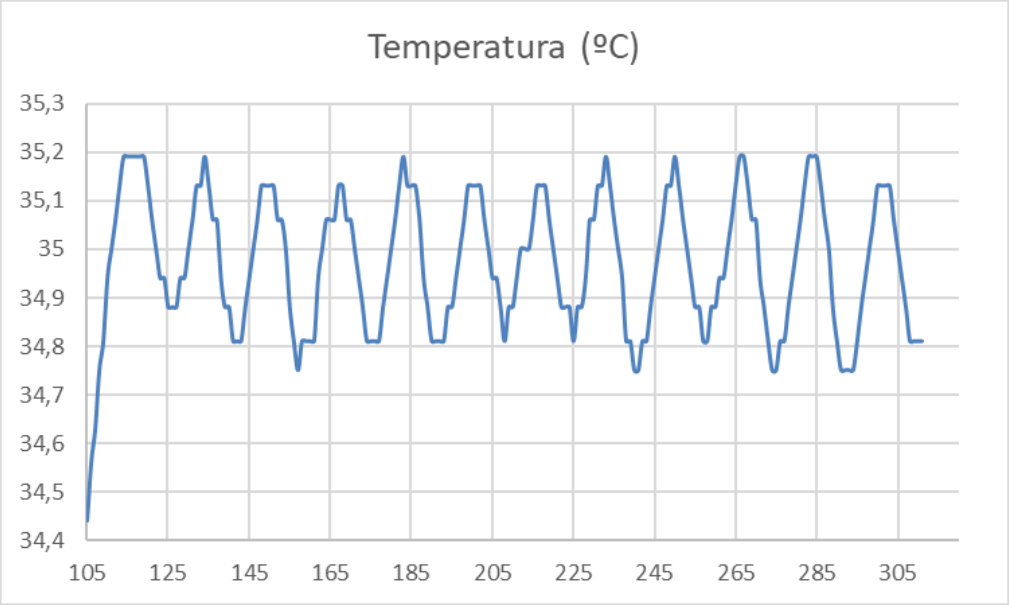
Figura 43 - Comportamento da temperatura utilizando o controlador de duas posições.



Fonte: autoria própria (2022).

Na figura 44 é possível analisar o comportamento da temperatura no regime permanente.

Figura 44 - Comportamento da temperatura utilizando o controlador de duas posições no regime permanente.



Fonte: autoria própria (2022).

Foi calculado o tempo médio que a ventoinha e a lâmpada ficaram ligadas, baseado nos dados de temperatura do controlador de duas posições, registrados a cada segundo no *Excel*, foi obtido o ciclo dos atuadores, como pode ser visto na tabela 3.

Tabela 3 - Ciclo dos atuadores utilizando o controlador de duas posições.

Estado	Tempo (s)
Tudo ligado	1
Ventoinha ligada	9
Tudo desligado	1
Lâmpada ligada	7

Fonte: autoria própria (2022).

Utilizando os dados da tabela 3, podemos calcular o percentual de tempo de ventoinha e lâmpada quando ligadas e a potência consumida. Sabendo que o período do ciclo é a soma do tempo de todos os estados, vemos que o período apresenta uma duração de 18 segundos.

$$T_{Ventoinha} = \frac{9}{18} * 100\% = 50\% \quad (63)$$

$$P_{Ventoinha} = (50\%)(3\ W) = 1{,}5\ W \quad (64)$$

$$T_{Lâmpada} = \frac{7}{18} * 100\% = 38{,}89\% \quad (65)$$

$$P_{Lâmpada} = (38{,}89\%)(100\ W) = 38{,}89\ W \quad (66)$$

$$P_{Total} = P_{Ventoinha} + P_{Lâmpada} = 1{,}5 + 38{,}89 = 40{,}39\ W \quad (67)$$

Onde:

$T_{Ventoinha}$ = Percentual do ciclo do controlador onde a ventoinha fica ativa

$P_{Ventoinha}$ = Potência consumida pela ventoinha durante o ciclo

$T_{Lâmpada}$ = Percentual do ciclo do controlador onde a lâmpada fica ativa

$P_{Lâmpada}$ = Potência consumida pela lâmpada durante o ciclo

P_{Total} = Potência consumida pela lâmpada e ventoinha durante o ciclo

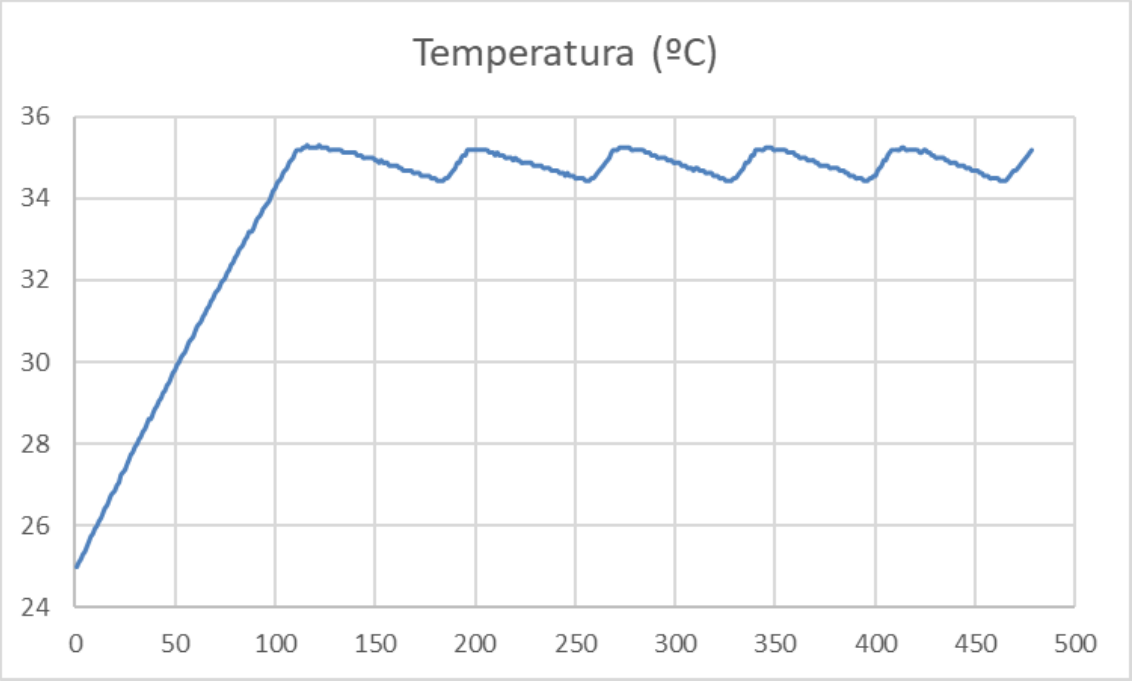
Desta forma conclui-se que a potência necessária para manter a temperatura em 35 ºC no regime permanente é de 40,39 W, além de que a temperatura máxima é de 35,2 ºC e a temperatura mínima é de 34,75 ºC.

Como pode ser notado na tabela 3 e nas equações 63 e 65, o problema do controlador de duas posições, é que os atuadores são acionados com uma frequência muito alta, fazendo com que haja um gasto energético elevado, além de que dependendo do atuador, haverá uma desgaste mecânico e uma redução da sua vida útil, como por exemplo, o relé que aciona a ventoinha.

5.2 Controlador de duas posições com intervalo diferencial

Na figura 45 é possível visualizar o comportamento da temperatura ao longo do tempo (segundos), utilizando o controlador de duas posições com intervalo diferencial. É perceptível que há uma oscilação semelhante à forma dente de serra, próximo da temperatura de referência. Nota-se que o período é maior quando comparado com o controlador de duas posições, como consequência, os atuadores serão acionados com uma frequência menor, reduzindo o desgaste mecânico e aumentando a vida útil.

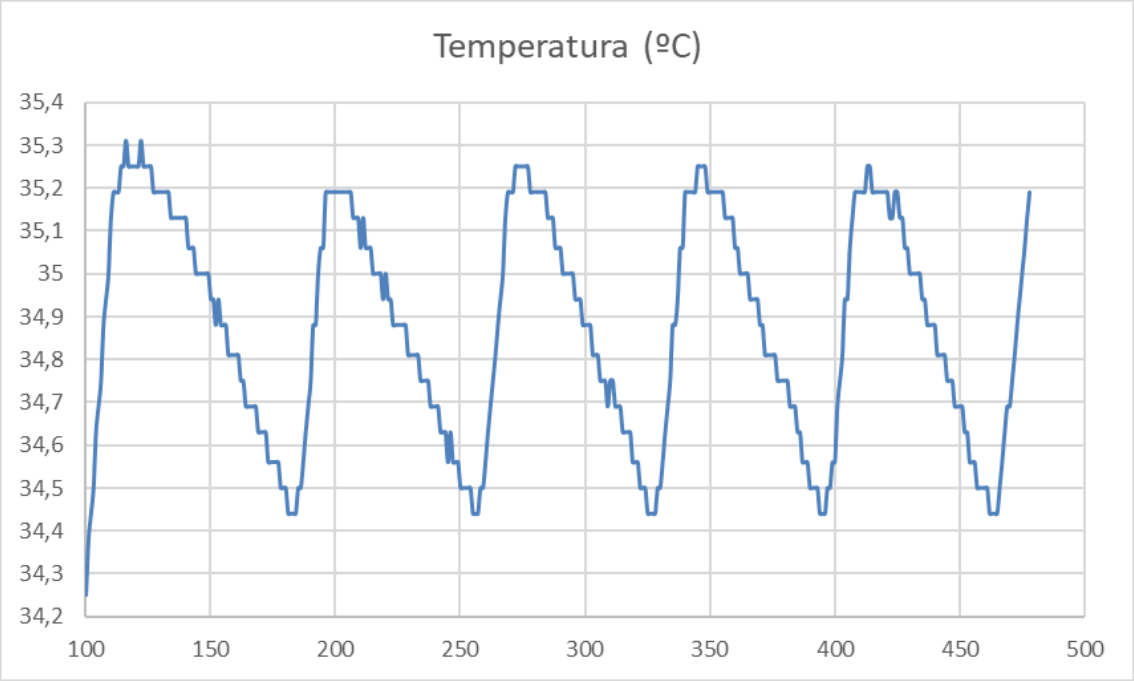
Figura 45 - Comportamento da temperatura utilizando o controlador de duas posições com intervalo diferencial.



Fonte: autoria própria (2022).

Na figura 46, é possível analisar o comportamento da temperatura no regime permanente.

Figura 46 - Comportamento da temperatura utilizando o controlador de duas posições com intervalo diferencial no regime permanente.



Fonte: autoria própria (2022).

Realizado o mesmo procedimento do controlador de duas posições, foi registrado o tempo médio de acionamento dos atuadores, como a temperatura máxima em regime permanente era de 35,3 ºC e a mínima de 34,45 ºC, a ventoinha não foi acionada já que só é ligada caso ultrapasse a temperatura de 35,5 ºC.

Tabela 4 - Ciclo dos lâmpada utilizando o controlador de duas posições com intervalo diferencial

Estado	Tempo (s)
Lâmpada ligada	10
Lâmpada desligada	60

Fonte: autoria própria (2022)

Assim o percentual de tempo em que a lâmpada é acionada durante um período é:

$$T_{Lâmpada} = \frac{10}{70} * 100\% = 14{,}29\% \quad (68)$$

Já a potência da lâmpada necessária para manter a temperatura no interior da estufa em 35ºC no regime permanente é.

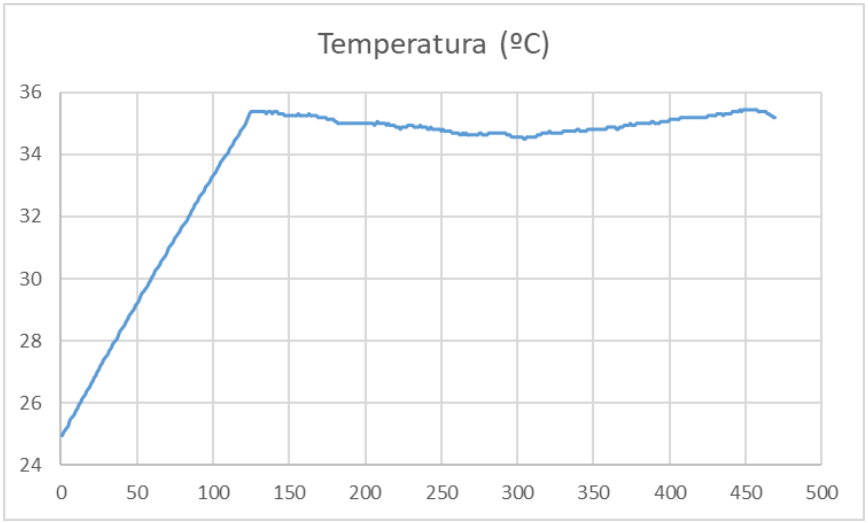
$$P_{Lâmpada} = (14{,}29\%)(100\ W) = 14{,}29\ W \quad (69)$$

Portanto, há um aumento significativo do ponto de vista da eficiência energética do controlador de duas posições para o controlador de duas posições com intervalo diferencial, visto que o primeiro controlador apresenta um gasto energético de 2,83 vezes maior do que o segundo controlador. Todavia ambos os controladores apresentam uma oscilação de temperatura em relação a temperatura de referência, que apesar de ser baixa e dependendo da aplicação de um controlador, seja necessário uma maior estabilidade e um menor erro.

5.3 Controlador PI

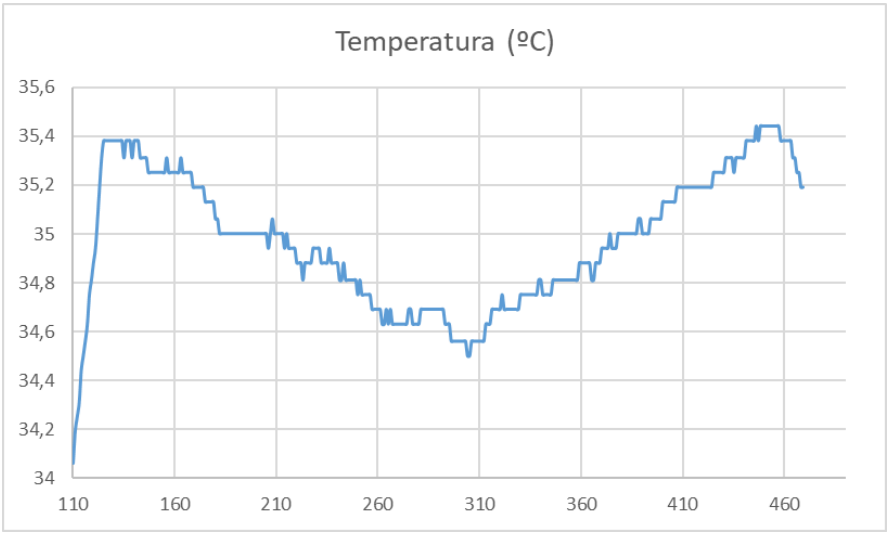
Nas figuras 47 e 48 é possível visualizar o comportamento da temperatura ao longo do tempo (segundos) utilizando o controlador PI.

Figura 47 - Comportamento da temperatura utilizando o controlador PI.



Fonte: autoria própria (2022).

Figura 48 - Comportamento da temperatura utilizando o controlador PI no regime permanente.



Fonte: autoria própria (2022).

É possível notar de forma clara através da figura 48, que a temperatura apresentou uma oscilação significativa, apesar da mesma se manter em uma faixa de temperatura aceitável, a razão deste problema se deve ao fato de que algumas linhas de código atrasam o processamento do *Arduino*.

O *Arduino* processa o código linha por linha, sendo que essa execução é feita em uma escala de tempo bastante reduzida, entre microssegundos e milissegundos, contudo existem certas linhas de códigos que demoram um pouco mais de tempo para ser processado. Como já foi visto, o módulo Dimmer do *Arduino* aciona a lâmpada através de um tempo configurado baseado no ângulo de disparo, sendo que esse tempo vai de 0 até 8,33 milissegundos, ou seja, qualquer atraso no tempo

de execução do código é extremamente sensível fazendo com que a lâmpada não seja acionada.

As principais linhas de código responsável por este atraso, são as linhas que utilizam a função própria do *Arduino* chamada *Serial.print()*. Esta função é a responsável pela apresentação dos dados obtidos pelo *Arduino*, além de escrever esses dados na planilha *Excel*.

Ressalta-se que a precisão do sensor de temperatura que pode apresentar ruídos em sua medição, além das aproximações que foram utilizadas para a construção da função de transferência da planta.

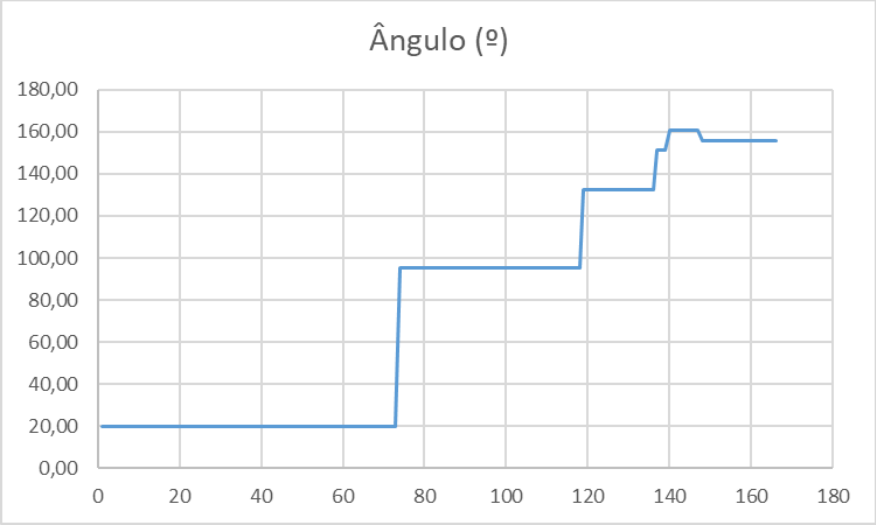
Ainda assim é nítido que foi possível ter a temperatura em uma faixa muito próxima do valor desejado com uma potência bastante baixa, de modo que o controlador de duas posições com intervalo diferencial tem um gasto energético 3,22 vezes maior do que o controlador PI.

Apesar do controlador PI apresentar uma resposta rápida, alcançando a temperatura de referência aproximadamente em um minuto e trinta segundos, além de uma eficiência energética alta, é necessário ter a função de transferência do sistema, o que pode acabar sendo um obstáculo em sua implementação.

5.4 Controlador rastreador baseado nos métodos numéricos

Os dados registrados mostram a concordância das iterações teóricas com as iterações reais, na figura 49 é possível notar que o ângulo atual se aproxima cada vez mais ao ângulo de 159º ao longo do tempo.

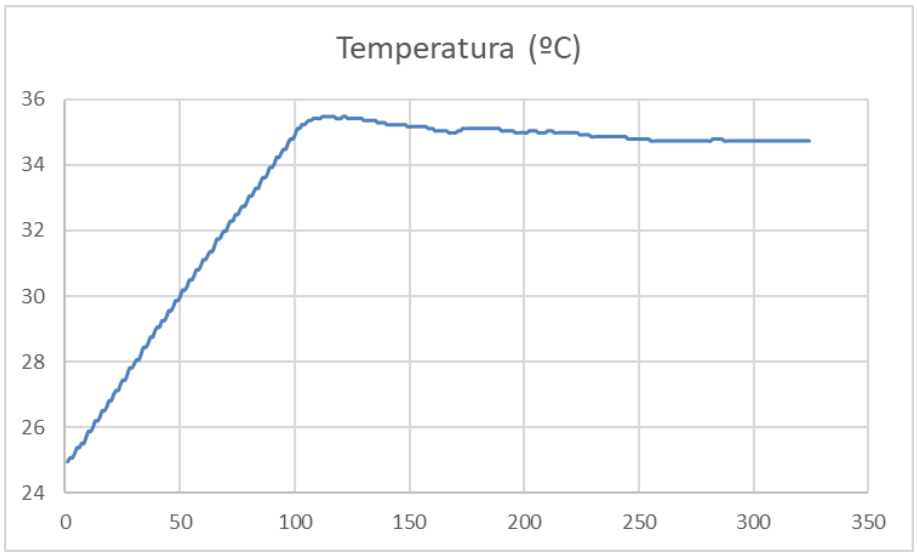
Figura 49 - Auto ajustamento do ângulo em relação ao tempo (s).



Fonte: autoria própria (2022).

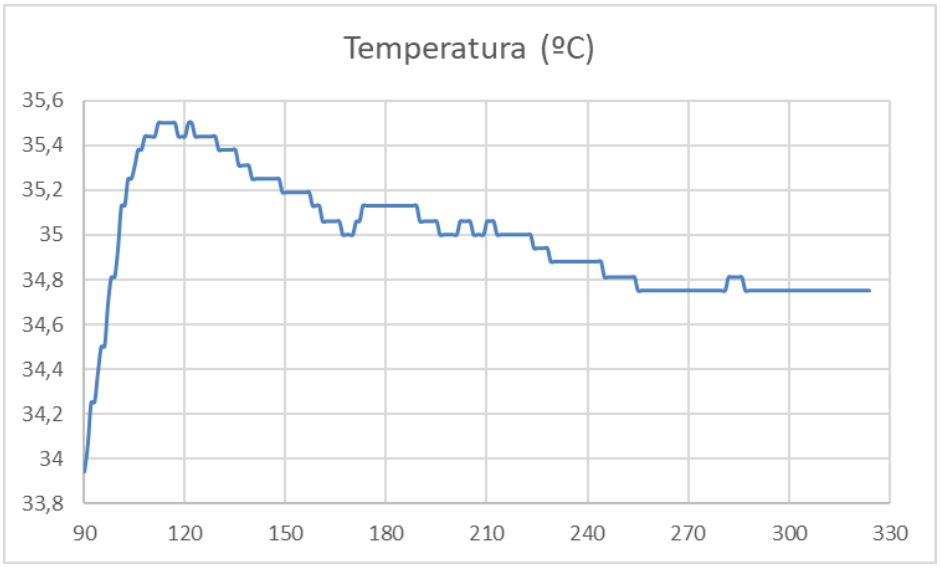
Nas figura 50 e 51, pode-se notar o comportamento da temperatura ao longo do tempo (segundos).

Figura 50 - Comportamento da temperatura utilizando o controlador rastreador em relação ao tempo (s).



Fonte: autoria própria (2022).

Figura 51 - Comportamento da temperatura utilizando o controlador rastreador no regime permanente em relação ao tempo (s).



Fonte: autoria própria (2022).

Da mesma forma que o controlador PI sofre interferências na variação da potência da lâmpada devido a linhas de códigos que atrasavam o acionamento do TRIAC, como a função *Serial.print()*, o código do controlador rastreador também

sofre com esse problema, visto que é preciso fazer o registro dos dados no *Excel* com essa função, o que fez com que houvesse oscilações na potência da lâmpada.

Apesar da instabilidade da lâmpada, e dos ruídos que o sensor de temperatura apresentava assim como qualquer outro sensor, é nítido através da figura 51, que a temperatura conseguiu se adequar e se manteve bastante próxima da temperatura de referência, além de que também conseguiu manter a temperatura com uma potência de 4,45 W no regime permanente assim como o controlador PI, contudo não necessitou de uma função de transferência e nem de um controlador, o que promove a possibilidade da utilização deste controlador em qualquer sistema que apresente ou não uma função de transferência.

5.5 Comparação dos controladores

As respostas dos controladores no regime transitório são aproximadamente iguais, visto que todos iniciam na mesma temperatura ambiente e utilizam a potência máxima e de forma contínua até atingir a temperatura de referência.

As temperaturas que são registradas no regime permanente são aceitáveis, de modo que apesar de haver oscilação nela, o erro não é significativo. As principais características podem ser vistas na tabela 5.

Tabela 5 - Comparação dos principais parâmetros dos controladores no regime permanente (240 segundos)

Controlador	Potência (35 ºC)	Temperatura Máxima	Temperatura Mínima	Erro Máximo absoluto	Erro Máximo percentual
Duas posições	40,39 W	35,2 ºC	34,75 ºC	0,25 ºC	0,714%
Duas posições com intervalo diferencial	14,29 W	35,25 ºC	34,45 ºC	0,55 ºC	1,571%
PI	4,45 W	35,41 ºC	34,45 ºC	0,55 ºC	1,571%
Rastreador	4,45 W	34,85 ºC	34,75 ºC	0,25 ºC	0,714%

Fonte: autoria própria (2022).

O controlador de duas posições é o controlador mais simples de ser implementado em linhas de código, todavia há um custo energético elevado além da utilização frequente dos atuadores, que dependendo do dispositivo resultará em uma redução na vida útil através de desgastes mecânicos, como por exemplo o relé que aciona a ventoinha na estufa.

O controlador de duas posições com intervalo diferencial é uma melhoria do controlador de duas posições, visto que neste caso há uma tolerância maior para a variação da temperatura antes que os atuadores sejam acionados, o que faz com que haja um gasto energético reduzido além do aumento da vida útil dos atuadores,

contudo houve um aumento de duas vezes no erro, mas ainda assim continua um erro bastante baixo.

O controlador PI neste projeto apresentou uma eficiência maior quando comparado com o controlador de duas posições e de duas posições com intervalo diferencial, visto que tendo a função de transferência do sistema, tem o comportamento do sistema baseado em uma equação matemática, o que permite projetar o comportamento do sinal de saída como: Tempo de subida, tempo de acomodação, Sobressalto (*Overshoot*) e erro. Todavia, como já foi mencionado, é necessário se ter a função de transferência da planta além da projeção do controlador, o que acaba se tornando um empecilho em seu uso, visto que há sistemas onde não é possível descrever matematicamente o comportamento dele, e nos casos onde é possível obter a função de transferência, é necessário garantir que os ruídos também sejam integrados a FT, que são fatores externos que estão presentes na maioria das plantas, caso contrário haverá uma imprecisão durante a aplicação prática.

O controlador rastreador busca se adaptar de forma gradual, com base na temperatura atual e na temperatura de referência, o controlador consegue ajustar a potência da lâmpada de modo que possa anular o erro utilizando a menor potência possível, além de estabilizar a temperatura sem apresentar oscilações. Sua principal vantagem sobre o controlador PI, se deve ao não uso de uma função de transferência além da sua capacidade em se adaptar sob ruídos.

Os códigos utilizados no *Arduino* para cada controlador podem ser vistos nos anexos: A, B, C e D.

6 CONSIDERAÇÕES FINAIS

Neste capítulo é apresentada a conclusão do trabalho e sugestões para trabalhos futuros.

6.1 Conclusão

Foi obtido êxito na execução dos objetivos específicos, visto que, o trabalho apresentou a importância do controle no ambiente industrial, além de detalhar na prática a construção e utilização dos principais controladores clássicos.

O trabalho foi construído em uma estufa de baixo custo, com componentes eletrônicos acessíveis, de modo que é possível a ampliação do estudo na área de controle por qualquer pessoa, também é possível replicar o sistema, utilizando este trabalho como guia.

A obtenção dos dados de temperatura de cada controlado, permitiu a comparação dos resultados, evidenciando a eficiência energética do controlador rastreador, além da estabilidade na resposta no regime permanente.

Também deve ser ressaltado a demonstração de uma alternativa aos sistemas de controle, demonstrando a possibilidade de construção de um controlador capaz de rastrear o comportamento dos atuadores de modo a ter uma resposta e gasto energético igual ao controlador PI, porém sem a necessidade de se ter a função de transferência do sistema.

6.2 Sugestões para trabalhos futuros

Realizar a aplicação prática da estufa, como a utilização em uma chocadeira ou qualquer outro sistema necessite trabalhar com temperaturas constantes e inferiores a 70 ºC.

Também pode ser indicado a utilização do controlador rastreador em outros tipos de sistemas que também necessitam de um controlador, como o enchimento ou esvaziamento de um líquido de um reservatório, variação na velocidade ou força de um motor, ou qualquer planta onde se tem um sensor e um atuador.

REFERÊNCIAS

AGUIRRE, L. A. **Controle e automação**. v. 3, 2007.

AGUIRRE, L. A. **Introdução à Identificação de Sistemas – Técnicas Lineares e Não-Lineares Aplicadas a Sistemas Reais**. 3. ed. Belo Horizonte: Editora UFMG, 2007.

AHMED, A. **Eletrônica de Potência**. 1. ed. Pearson, 2000.

BANZI, M.; SHILOH, S. **Primeiros Passos com o Arduino:** A plataforma de prototipagem eletrônica open source. 2. ed. [S.l.]: Novatec, 2015.

Baú da Eletrônica. Disponivel em: <https://www.baudaeletronica.com.br/triac-bta16-600b.html>. Acesso em: 18 ago. 2022.

BRAGA, N. C. **Relés:** Circuitos e aplicações. [S.l.]: NCB, 2017.

BRINCANDO COM IDEIAS. DIMMER com Arduino | Controle de Brilho da Lâmpada | Chocadeira Automática. YouTube, 11 de fevereiro de 2020. Disponível em: < https://www.youtube.com/watch?v=_vMG9QmPtNE >. Acesso em: 25 de agosto de 2022.

CAMPOS, A. **BR-Arduino.org**, 2016. Disponivel em: <https://br-arduino.org/2016/03/arduino-triac-dimmer.html>. Acesso em: 17 ago. 2022.

CHAPRA, S. C.; CANALE, R. P. **Métodos numéricos para engenharia**. 5. ed., 2008.

Clube de Robótica. Disponivel em: <https://cluberobotica.wordpress.com/2019/04/29/plx-daq/#:~:text=O%20PLX%2DDAQ%20%C3%A9%20uma,microcontroladores%20Parallax%20para%20Microsoft%20Excel.&text=%E2%80%9CDAQ%20%E2%80%93%20Significa%20aquisi%C3%A7%C3%A3o%20de%20dados,o%20uso%20de%20um%20computa>. Acesso em: 06 ago. 2022.

docs.arduino. Disponivel em: <https://docs.arduino.cc/hardware/uno-rev3>. Acesso em: 02 ago. 2022.

DORF, R. C.; BISHOP, R. H. **Sistemas de Controle Moderno**. 8. ed., Addison Wesley Longman, 2001.

FUENTES, R. C. **e-tec Brasil**, 2011. Disponivel em: <http://redeetec.mec.gov.br/images/stories/pdf/eixo_ctrl_proc_indust/tec_autom_ind/eletronica/161012_eletronica.pdf>. Acesso em: 13 ago. 2022.

HART, Daniel W. **Eletrônica de Potência Análise e Projetos de Circuitos**. 1. ed., Editora Mcgraw-hill, 2012.

INEE. **INSTITUTO NACIONAL DE EFICIÊNCIA ENERGÉTICA**. O que é eficiência energética? Por que se desperdiça energia? Disponível em:

<www.inee.org.br/eficiencia_o_que_eh.asp?Cat=eficiencia>. Acesso em: 26 de dez. 2022.

JUNIOR, Ervaldo G. **Introdução a Sistemas de Supervisão, Controle e Aquisição de Dados: SCADA**. Alta Books, 2019.

KITTEL, C. **Física térmica**. Espanha: Editorial Reverté.

KNIGHT, R. D. **Física - V1:** Uma Abordagem Estratégica - Mecânica Newtoniana, Gravitação, Oscilações e Ondas. 2. ed. [S.l.]: Bookman, 2009.

LOCATELLI, C. **Curto Circuito**, 2021. Disponivel em: <https://curtocircuito.com.br/blog/Categoria%20Arduino/como-utilizar-o-ds18b20>. Acesso em: 02 ago. 2022.

MCROBERTS, M. **Arduino Básico**. Novatec Editora, 2018.

MONK, S. **Programação com Arduino:** Começando com Sketches. 2. ed. [S.l.]: Bookman, 2016.

OGATA, K. **Engenharia de Controle Moderno**. 4. ed., Prentice-Hall, 2004.

PEREIRA, A. A. K. **Laboratório de Garagem**, 2014. Disponivel em: <https://labdegaragem.com/profiles/blogs/tutorial-dimmer-shield>. Acesso em: 06 ago. 2022.

PETRUZELLA, F. D. **Controladores Lógicos Programáveis**. 4. ed. [S.l.]: Bookman, 2014.

RIVA, A. **Alessandro Riva elettronica**, 2013. Disponivel em: <http://alessandro-riva-elettronica.blogspot.com/2013/06/>. Acesso em: 20 ago. 2022.

TDPS – Estratégias de controle para processos industriais. Disponivel em: <https://apcmode.com/?page_id=322>. Acesso em: 15 ago. 2022.

THOMSEN, A. **FILIPEFLOP**, 2015. Disponivel em: <https://www.filipeflop.com/blog/sensor-de-temperatura-ds18b20-arduino/>. Acesso em: 02 ago. 2022.

Wikipédia - A enciclopédia livre. Disponivel em: <https://pt.wikipedia.org/wiki/Controlador_proporcional_integral_derivativo>. Acesso em: 28 ago. 2022.

APÊNDICE A – CÓDIGO DO CONTROLADOR DE DUAS POSIÇÕES

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Pino do sensor DS18B20
# define ONE_WIRE_BUS 7

OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);
DeviceAddress sensor1;

// ===============================================

# define pino_ventoinha 12
# define pino_lampada 3

float temperatura_referencia = 35.0;
float faixa_superior = temperatura_referencia + 0.5;
float faixa_inferior = temperatura_referencia - 0.5;

// ===============================================
// CÓDIGO INICIAL DO MACRO DE EXCEL

int linha = 0;        // variavel que se refere as linhas do excel
int LABEL = 1;

// ===============================================

void setup(void)
{
```

```
// ================================================
// SETUP DO CÓDIGO DE TEMPERATURA

  Serial.begin(9600);
  sensors.begin();
  Serial.print(sensors.getDeviceCount(), DEC);
  if (!sensors.getAddress(sensor1, 0))
     Serial.println("Sensores nao encontrados!");

  mostra_endereco_sensor(sensor1);

  pinMode(pino_ventoinha, OUTPUT);
  pinMode(pino_lampada, OUTPUT);

// ================================================
// SETUP DO CÓDIGO DO EXCEL

  Serial.println("CLEARDATA");
  Serial.println("LABEL,Hora,Linha,Valor");

}

void mostra_endereco_sensor(DeviceAddress deviceAddress)
{
  for (uint8_t i = 0; i < 8; i++)
  {
    if (deviceAddress[i] < 16) Serial.print("0");
    Serial.print(deviceAddress[i], HEX);
  }
}

void loop()
{
```

```
// ================================================
// LOOP DO CÓDIGO DE TEMPERATURA

  sensors.requestTemperatures();
  float tempC = sensors.getTempC(sensor1);

// ================================================

  linha++;

  Serial.print("DATA,TIME,");
  Serial.print(linha);
  Serial.print(",");
  Serial.println(tempC);

  if (tempC < temperatura_referencia) // Se a temperatura atual for menor do que a de referência
  {
  digitalWrite(pino_lampada, HIGH); // Liga a Lâmpada
  digitalWrite(pino_ventoinha, HIGH); // Desiga a ventoinha (Inverso)
  }

  if (tempC > temperatura_referencia) // Se a temperatura atual for maior do que a de referência
  {
  digitalWrite(pino_lampada, LOW); // Desliga a lâmpada
  digitalWrite(pino_ventoinha, LOW); // Liga a ventoinha (Inverso)
  }

  if (tempC == temperatura_referencia) // Se a temperatura atual for igual do que a de referência
  {
  digitalWrite(pino_lampada, LOW); // Desliga a lâmpada
```

```
  digitalWrite(pino_ventoinha, HIGH); // Desiga a ventoinha (Inverso)
  }

delay(500);  // espera 500 milisegundos (0,5 segundo)

// ================================================

}
```

APÊNDICE B – CÓDIGO DO CONTROLADOR DE DUAS POSIÇÕES COM INTERVALO DIFERENCIAL

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 7

OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);
DeviceAddress sensor1;

//===========================================

# define pino_ventoinha 12
# define pino_lampada 3

float temperatura_referencia = 35.0;
float faixa_superior = temperatura_referencia + 0.5;
float faixa_inferior = temperatura_referencia - 0.5;
```

```
int estado1 = 0;
int estado2 = 0;

//============================================
// CÓDIGO INICIAL DO MACRO DE EXCEL

int linha = 0;
int LABEL = 1;

//============================================

void setup(void)
{

//============================================
// SETUP DO CÓDIGO DE TEMPERATURA

  Serial.begin(9600);
  sensors.begin();

  Serial.print(sensors.getDeviceCount(), DEC);
  if (!sensors.getAddress(sensor1, 0))
     Serial.println("Sensores nao encontrados !");

  mostra_endereco_sensor(sensor1);

  pinMode(pino_ventoinha, OUTPUT);
  pinMode(pino_lampada, OUTPUT);


//============================================
// SETUP DO CÓDIGO DO EXCEL
```

```
  Serial.println("CLEARDATA");
  Serial.println("LABEL,Hora,Linha,Valor");

//============================================
}

void mostra_endereco_sensor(DeviceAddress deviceAddress)
{
  for (uint8_t i = 0; i < 8; i++)
  {
    // Adiciona zeros se necessário
    if (deviceAddress[i] < 16) Serial.print("0");
    Serial.print(deviceAddress[i], HEX);
  }
}

void loop()
{

//============================================
// LOOP DO CÓDIGO DE TEMPERATURA

  sensors.requestTemperatures();
  float tempC = sensors.getTempC(sensor1);

//============================================

  linha++;

  Serial.print("DATA,TIME,");
  Serial.print(linha);
  Serial.print(",");
  Serial.println(tempC);
```

```
if (tempC < faixa_inferior)
{
  estado1 = 1;
}

if (tempC > faixa_superior)
{
  estado2 = 1;
}

if (estado1 == 1)
{
digitalWrite(pino_lampada, HIGH); // Liga a lâmpada
digitalWrite(pino_ventoinha, HIGH); // Desiga a ventoinha (Inverso)

if (tempC >= temperatura_referencia)
{
  estado1 = 0;
}

}

if (estado2 == 1)
{
digitalWrite(pino_lampada, LOW); // Desliga a lâmpada
digitalWrite(pino_ventoinha, LOW); // Liga a ventoinha (Inverso)

if (tempC >= temperatura_referencia)
{
  estado2 = 0;
}
```

```
  }


  if ((estado1 == 0) && (estado2 == 0)) // Se a temperatura atual for igual do que a de referência
  {
  digitalWrite(pino_lampada, LOW); // Desliga a lâmpada
  digitalWrite(pino_ventoinha, HIGH); // Desiga a ventoinha (Inverso)
  }


delay(500);  // espera 500 milisegundos (0,5 segundo)

//============================================

}
```

APÊNDICE C – CÓDIGO DO CONTROLADOR PI

```
int contador = -1;
int contador2 = 0;
int segundos = 0;
int minutos = 0;
int horas = 0;
int x = 0;
int tempo_segundos = 15;
float contador_vetor[16]; // Ex.: 10 segundos -> 11     120 segundos -> 121



#include <OneWire.h>
```

```
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 7

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
DeviceAddress sensor1;



#define pinZC  2 // Pino que recebe o pulso, dizendo que está em 0 volts
#define pinDIM 3 // Pino que envia o pulso para o acionamento do TRIAC
float alpha = 159.0;
float TempoDelayMicro = 0.0;



#define pino_ventoinha 12



float temperatura_referencia = 35;
float faixa_superior = temperatura_referencia + 0.5;
float faixa_inferior = temperatura_referencia - 0.5;



int alpha_ativo = 0;

//=========================================

void setup(void)
{
```

```
  Serial.begin(9600);



  sensors.begin();
  if (!sensors.getAddress(sensor1, 0));



  pinMode(pinDIM, OUTPUT);
  pinMode(pinZC, INPUT);
  attachInterrupt(digitalPinToInterrupt(pinZC), sinalZC, RISING);



  pinMode(pino_ventoinha, OUTPUT);
  digitalWrite(pino_ventoinha, HIGH);
}

//============================================

void loop()
{
  sensors.requestTemperatures();
  float tempC = sensors.getTempC(sensor1);



  contador = contador + 1;
  contador_vetor[contador] = tempC;


//============================================
```

```
  TempoDelayMicro = alpha*46.29629;

  if (tempC < temperatura_referencia)
  {
    digitalWrite(pinDIM, HIGH);
  }

  if (tempC >= temperatura_referencia)
  {
    alpha_ativo = 1;
  }

//============================================

  delay(430);
}

//============================================

void sinalZC()
{
  if (alpha_ativo == 1)
  {
  delayMicroseconds(TempoDelayMicro);
  digitalWrite(pinDIM, HIGH);
  delayMicroseconds(6);
  digitalWrite(pinDIM, LOW);

  }

}
```

APÊNDICE D – CÓDIGO DO CONTROLADOR RASTREADOR BASEADO NOS MÉTODOS NUMÉRICOS

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 7

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
DeviceAddress sensor1;



float temp_desejada = 35.0;
float temp_1 = 0.0;
float temp_2 = 0.0;
float temp_3 = 0.0;

float aux1 = 0.0;
float aux2 = 0.0;

int estado = 0; // 1 = Subindo      2 = Descendo

float angulo_max = 170.0;
float angulo_min = 20.0;

float espera1 = 1;
float espera2 = 0;

//==============================
// CONTROLADOR DO ÂNGULO DE CORTE
```

```
#define pinZC  2
#define pinDIM 3

float alpha = 20.0; // Mínimo 20        Máximo 165 - 170
float TempoDelayMicro = 0.0;

//================================
// VENTOINHA
# define pino_Ventoinha 12
int estado_ventoinha = 0;

//================================

// CÓDIGO INICIAL DO MACRO DE EXCEL

int linha = 0;
int LABEL = 1;

void setup(void)
{
//================================
// SENSOR DE TEMPERATURA

  Serial.begin(9600);
  sensors.begin();
  if (!sensors.getAddress(sensor1, 0));

//================================
// CONTROLADOR DO ÂNGULO DE CORTE

  pinMode(pinDIM, OUTPUT);
  pinMode(pinZC, INPUT);
```

```
  attachInterrupt(digitalPinToInterrupt(pinZC), sinalZC, RISING);

//================================
// VENTOINHA

  //pinMode(pino_Ventoinha, OUTPUT);

//================================
// SETUP DO CÓDIGO DO EXCEL

  Serial.println("CLEARDATA");
  Serial.println("LABEL,Hora,Angulo,Valor");
}

void loop()
{
//================================
// SENSOR DE TEMPERATURA

  sensors.requestTemperatures();
  float tempC = sensors.getTempC(sensor1);

//================================
// Pega os 3 últimos valores de temperatura, onde os 3 são diferentes entre si
  aux2 = aux1;
  aux1 = temp_1;

  temp_1 = tempC;

  if (temp_1 != aux1)
  {
    temp_2 = aux1;
  }
```

```
  if((temp_2 != aux2) and (temp_1 != aux2))
  {
    temp_3 = aux2;
  }

//==================================
// Define a tendência da temperatura

  if ((temp_1 > temp_2) and (temp_2 > temp_3))
  {
    estado = 1; // Temperatura aumentando
  }

  else if ((temp_1 < temp_2) and (temp_2 < temp_3))
  {
    estado = 2; // Temperatura caindo
  }

  else
  {
    estado = 0; // Temperatura estável
  }

//================================
// Define o ângulo de corte necessário

if ((tempC > temp_desejada) and (espera1 == 1)) // and (estado == 1)
{
  angulo_min = alpha;
  alpha = (angulo_max + angulo_min)/2;
  espera1 = 0;
}
```

```
if ((tempC < (temp_desejada - 0.15)) and (espera2 == 1)) // and (estado == 2)
{
  angulo_max = alpha;
  alpha = (angulo_max + angulo_min)/2;
  espera2 = 0;
}

if (tempC <= temp_desejada)
{
  espera1 = 1;
}

if (tempC > temp_desejada)
{
  espera2 = 1;
}

// DESLIGA A LÂMPADA
if (tempC > temp_desejada)
{
  estado_ventoinha = 1;
}

// LIGA A LÂMPADA
if (tempC <= temp_desejada)
{
  estado_ventoinha = 0;
}

//================================
// ADICIONA OS DADOS NO EXCEL
```

```
  linha++; // incrementa a linha do excel para que a leitura pule de linha em linha

  Serial.print("DATA,TIME,");
  Serial.print(alpha);
  Serial.print(",");
  Serial.println(tempC);

//================================
// CONTROLADOR DO ÂNGULO DE CORTE

  TempoDelayMicro = alpha*46.29629;
  delay(1000);
}

void sinalZC()
{
  if (estado_ventoinha == 0)
  {
  delayMicroseconds(TempoDelayMicro);
  digitalWrite(pinDIM, HIGH);
  delayMicroseconds(6);
  digitalWrite(pinDIM, LOW);
  }
}
```

APÊNDICE E – CÓDIGO PARA A CONVERSÃO DE ÂNGULO PARA POTÊNCIA (PYTHON)

```
# Do 20 ao 170, com passo de 10

valor_real1 = [96.58,95.77,92.42,89.99,87.17,81.33]
valor_real2 = [72.71,61.48,53.49,43.92,34.01,24.32]
```

```
valor_real3 = [14.68,8.41,4.04,1.48]

valor_real = []
valor_real.extend(valor_real1)
valor_real.extend(valor_real2)
valor_real.extend(valor_real3)

c1 = 0
c2 = 20

d = 0

t = 0

while t == 0:

   angulo = int(input("Digite o ângulo maior ou igual que 20 e menor ou igual a 170 para saber a potência: "))

   if (angulo < 20) or (angulo > 170):

      while True:

         if (angulo < 20) or (angulo > 170):
            angulo = int(input("O ângulo digitado não pode ser utilizado, digite um outro ângulo: "))

         else:
            break

#=====================================================
# Nessa parte eu já tenho o valor que está dentro do alcance permitido (maior ou igual a 20 / menor ou igual a 70)
# E verifico se o ângulo já está previamente na lista -> 20,30,40,50,...,170
```

```
    while c2 <= 170:

        if angulo == c2:
            potencia = valor_real[c1]
            d = 10 # Faz com que a próxima condição não aconteça
            break

        c1 = c1 + 1
        c2 = 20 + c1*10

#=====================================================
# Se o ângulo não já estiver previamente na lista -> 20,30,40,50,...,170     Tentaremos o resto dos ângulos

    menor_anterior = 0
    menor_atual = 20
    menor_referencia = 0

    maior_anterior = 0
    maior_atual = 20
    maior_referencia = 0

    c3 = 0
    d2 = 0
    d3 = 0

#=====================================================

    if d == 0:

        while True:

            if (menor_atual > angulo) and (menor_anterior < angulo):
```

```
            menor_referencia = menor_anterior
            d3 = 10

        if (maior_atual > angulo) and (d2 == 0):
            maior_referencia = maior_atual
            d2 = 10

        menor_anterior = menor_atual
        menor_atual = menor_atual + 10

        maior_anterior = maior_atual
        maior_atual = maior_atual + 10

        if (d2 + d3) == 20:
            break

#=====================================================

    posicao_menor = int((menor_referencia/10) - 2)
    posicao_maior = int((maior_referencia/10) - 2)

    c4 = 0
    media_valor = 0
    media_potencia = 0

    maior_potencia = valor_real[posicao_menor]
    menor_potencia = valor_real[posicao_maior]
    maior_valor = maior_referencia
    menor_valor = menor_referencia

#=====================================================

    while True:
```

```
media_valor = (maior_valor + menor_valor)/2
media_potencia = (maior_potencia + menor_potencia)/2

if media_valor > angulo:
    maior_valor = media_valor
    menor_potencia = media_potencia

if media_valor < angulo:
    menor_valor = media_valor
    maior_potencia = media_potencia

c4 = c4 + 1

if c4 == 8:
    print("%.2f" %media_potencia)
    break
```